

# التطبيقات التجارية لبرمجيات الحاسب الآلي

أعداد

دكتور أحمد عبد العظيم أحمد الدغدي

كلية التجارة - جامعة جنوب الوادي

قسم الأساليب الكمية





## مقدمة

بسم الله الرحمن الرحيم، والحمد لله رب العالمين له الحمد في الأولى والآخرة، والصلاة والسلام على سيد الخلق وخاتم الأنبياء والمرسلين، وبعد،  
يقدم هذا الكتاب الركائز الأساسية لبرمجة الحاسبات، كمحاولة لتوجيه الدارس في أولى ساعات تعلمه برمجة الحاسب التي غالبا ما تتسم بالارتباك والخط، فإذا ما تتبع الكتاب بدقة سيجده يقدم له القاعدة الأساسية التي تساعد على فهم كتب البرمجة الأكثر تقدما وتعقيدا.  
وطبقا لما سبق تم أعداد هذا الكتاب ليحقق عدد من الأهداف المأمول تحقيقها بعد الانتهاء منه والتمثلة في:

\*\* فهم واستيعاب أسس وقواعد لغات الحاسب المختلفة وكيفية الاختيار فيما بينها وبما يتناسب مع المشكلة المراد حلها.

\*\* فهم واستيعاب كيفية الاستفادة ببرمجيات الحاسب الآلي في حل المشاكل التطبيقية في مجالات التجارة والإحصاء.

\*\* ممارسة التفكير الابتكاري والإبداعي في مواجهة المواقف الادارية المختلفة.

\*\* استخدام المنهج العلمي في حل المشكلات واتخاذ القرارات في المستويات الادارية المختلفة.

\*\* اكتساب مهارات الاتصال الفعال والعمل الجماعي والابتكار والتطوير والتحسين المستمر في العمل.

وعلى هذا يحتوي هذا الكتاب على ثلاث أبواب الأول يتناول اسس ومبادئ كتابة والتعامل مع البرامج

من خلال التعرف على لغات البرمجة المختلفة، ثم كيفية التخطيط للبرنامج والثاني يتناول الملامح

الأساسية للغة البرمجة ++C وتطبيقاتها التجارية والثالث يتناول أحد البرامج الإحصائية

، برنامج SAS .

كما يحتوي الكتاب على الكثير من التطبيقات والتمارين لتكون خير وسيلة لتعميق الفهم والتعلم لدى

الدارس والاستعداد لمواجهة وحل تلك المشكلات في المستقبل.

ولا يفوتني أن أتقدم بكل الشكر والعرفان الى الأساتذة الأفاضل الذين استفدت من كتاباتهم في هذا

المجال الواسع من مجالات العلم والمعرفة، وأرجو من الله سبحانه وتعالى أن يكون العلم النافع والعمل

الصالح، إنه سميع الدعاء .

دكتور أحمد عبد العظيم الدغدي





# الباب الأول

## أساسيات البرمجة



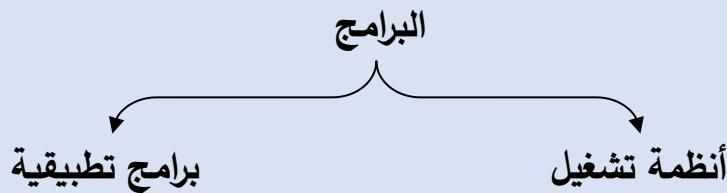
## الفصل الأول

### لغات الحاسب الآلي

يقوم الحاسب بمعالجة البيانات لتصبح معلومات ذات معنى، وتدخل البيانات لمعالجتها الحاسب ببرامجه منتجاً معلومات ، فالبيانات وحدها لا تكون مفيدة لاستخدامها في اتخاذ القرارات والمعلومات هي التي تفيد في هذه الحالة، وعلى هذا تمثل البرامج القوة المحركة للحاسبات، والبرنامج عبارة عن تعليمات لا يستطيع الحاسب أداء أي عمل بدونها، ومهمة المبرمج توجيه الحاسب لتلقى البيانات (أدخال)، وتحويلها الى معنى يفيد المستخدم (معالجة)، وعرضها امامه (إخراج)، دون شرط أن تكون له دراية بالأمور الفنية فكل ما يسعى له المستخدم هو النتائج التي يوفرها الحاسب ببرامجه، ومع التطور السريع للأعمال وزيادة عدد المستخدمين، كانت الحاجة الى تطوير البرامج، واستحداث لغات البرمجة.

وعند بداية ظهور أجهزة الحاسبات الإلكترونية كان من اهم اسباب عدم انتشارها على نطاق واسع صعوبة التعامل معها واستخدامها، بسبب اعتمادها الأساسي على كتابة البرامج لها بلغة الآلة الصعبة الاستخدام لغير المتخصصين، ولما بدأ استخدام لغات البرمجة الأسهل من لغة الآلة، أصبح في إمكان غير المتخصصين برمجة الأجهزة، إذ امتازت لغات البرمجة بقربها من اللغة المألوفة للأفراد.

وتنقسم البرامج بصفة عامة الى جزئين وهما:



ويمكن توضيح كلاً منهما فيما يلي:

### أنظمة تشغيل:

ويتكون نظام التشغيل من مجموعة من برامج تحتوي على تعليمات Instructions وأوامر Commands مكتوبة بإحدى لغات البرمجة القريبة من لغة الآلة، ووظيفتها تنظيم والتحكم في تشغيل وإدارة الحاسب الألى ومن أهمها Dos , Windows , Unix ، وتمثل هذه البرامج الأداة التي يستطيع العنصر البشرى من خلالها التعامل مع الحاسب بكل مكوناته المادية والبرامج بمختلف أنواعها، وبدون نظام التشغيل يعتبر الحاسب مجموعة من قطع الحديد والبلاستيك.

### برامج تطبيقية:

وهي برامج تساعد المستخدم على ادخال البيانات أو معالجتها بالصورة المناسبة لإخراج النتائج المطلوبة حيث يتم استخدام البرنامج المناسب للعمل المناسب ومن أهمها: Office Word , Excel , وبقية برامج Office وتنقسم البرامج التطبيقية الى عدة أنواع:

- برامج تطبيقية عامة مثل ( Word-Excel ).
- برامج تطبيقية خاصة ذات برمجة لأغراض محددة مثل حساب فواتير الكهرباء.

يعتبر للحاسب الآلى كيان مستقل يجب على الإنسان أن يجد طريقة مناسبة للتعامل معه، وكان عليه إما أن يتعامل مع الحاسب باللغة التي يستخدمها الإنسان أو بواسطة اللغة التي يتعامل بها الحاسب نفسه، ومن هنا ظهرت فكرة التوفيق ما بين لغة الإنسان ولغة الآلة (الحاسب)، وبالتالي فلغات البرمجة هي لغات خاصة بالتعامل مع الحاسب الآلى.

وتعتبر لغات البرمجة هي الوسيلة الوحيدة للتعامل مع الحاسب الآلي، وهي عبارة عن مجموعة من التعليمات والأوامر التي توجه الحاسب لتأدية عمليات معينة، ويوجد الآن ما يزيد عن 200 لغة من لغات البرمجة، وأي لغة من اللغات يجب أن تكون قادرة على تمثيل الحروف والأرقام والحروف الخاصة.

والجدير بالذكر أن لغات البرمجة قد مرت بعدة مراحل مختلفة وهي:

1- مرحلة لغة الآلة.

2- مرحلة اللغات الرمزية.

3- مرحلة اللغات ذات المستوى الرفيع.

ولقد ارتبطت كل مرحلة من هذه المراحل ارتباطا وثيقا بتطور استخدام وتصنيع الحاسبات نفسها، فالمرحلة الأولى من استخدام الحاسبات كانت اللغة الأساسية للتعامل مع الحاسب هي لغة الآلة. ومع تطور استخدام الحاسبات ولصعوبة التعامل مع الحاسبات بلغة الآلة ظهرت الحاجة إلى لغة أخرى، ومن هنا ظهرت اللغات الرمزية التي سهلت إلى حد ما التعامل مع الحاسبات ولكن انتشار الحاسبات ظل محدودا.

ومع استمرار التطور العلمي وتطور المعلومات والحاجة إلى استخدام الحاسبات في تطبيقات كثيرة في مختلف المجالات التجارية والاقتصادية والاجتماعية إلى جانب المجالات العلمية والهندسية والرياضية، ظهرت اللغات ذات المستوى الرفيع التي ساعدت إلى حد كبير في تسهيل التعامل مع الحاسبات الآلية مما أدى إلى انتشار الحاسبات، والجدير بالذكر أن اللغات ذات المستوى الرفيع هي لغات قريبة الشبة بلغة الإنسان، وبذلك انتقل العبء إلى الحاسبات التي تقوم بدورها بتحويل البرامج والأوامر المكتوبة باللغات الرمزية أو اللغات ذات المستوى الرفيع إلى لغة الآلة التي يتقبلها الحاسب، وذلك بواسطة برامج خاصة تعدها الشركات المنتجة للحاسبات، ويسمى البرنامج الذي يقوم بتحويل اللغات الرمزية واللغات ذات المستوى الرفيع

إلى لغة الآلة ( الحاسب ) بالبرنامج المترجم ، وبظهور هذا البرنامج انتشرت اللغات ذات المستوى الرفيع مما كان له أكبر الأثر في انتشار الحاسبات الآلية.

ويمكن إيجاز مراحل تطور لغات البرمجة كالتالي:

## 1- لغة الآلة Machine Language

لغة الآلة هي اللغة التي تستخدم فيها الأرقام الثنائية Binary (0 ، 1) في التعبير عن الأوامر والتعليمات المختلفة التي يتكون منها البرنامج، وكذلك البيانات اللازمة، وقد صاحبت هذه اللغة ظهور الحاسبات الآلية، وكان مصممو هذه الحاسبات هم الذين يقومون بتصميم البرامج مما أدى إلى صعوبة فهم تلك اللغة وبالتالي عدم انتشار الحاسبات التي صنعت أساساً لحل المشاكل المعقدة التي يصعب على الإنسان حلها بالطرق اليدوية.

ومن مزايا هذه اللغة:

في الواقع أن لهذه اللغة ميزة واحدة فقط ألا وهي أنها لا تحتاج إلى ترجمة، حيث أنها مكتوبة باللغة الطبيعية التي يستطيع الحاسب أن يتعامل معها مباشرة.

ومن عيوب هذه اللغة:

\* جميع الأوامر والتعليمات والبيانات مكتوبة بواسطة أرقام ثنائية (0، 1)، وهي طريقة مرهقة جداً في كتابة البرامج واحتمال الخطأ فيها كبير.

\* صعوبة فهم البرامج المكتوبة بلغة الآلة بواسطة الأشخاص العاديين الذين يرغبون في قراءة أي برنامج، وتكون شبة مستحيلة لأن البرامج تكون عبارة عن أرقام ثنائية (0، 1) ولا تحتوي على أية حروف أو رموز.

\* البرنامج المكتوب بلغة الآلة هو برنامج خاص لنوع معين من الحاسبات وبالتالي يحتاج مخطط البرامج أن يتعرف على إمكانيات وتفصيل الحاسب قبل كتابة البرامج إلى الحاسب.

\* البرنامج المكتوب بلغة الآلة لنوع معين من الحاسبات لا يمكن تشغيله على نوع آخر من الحاسبات بدون تعديلات جوهرية.

\* مخطط البرامج هو الوحيد الذي يقوم بعملية ترتيب الأوامر منطقيا وفقا لما يجب تنفيذه بواسطة الحاسب.

## 2- اللغة الرمزية Assembly Language

نتيجة للصعوبات التي نتجت عن استخدام لغة الآلة، فقد قامت الشركات المنتجة للحاسبات بالتوصل للغات رمزية تسهила لهذه الصعوبات، حتى تساعد على انتشار الحاسبات وتعتبر هذه اللغات مرحلة وسطى بين لغة الآلة واللغات ذات المستوى الرفيع. وتستخدم هذه اللغة خليطا من بعض الأرقام والرموز والعلامات، وذلك عن طريق إعطاء أسماء للأوامر والتعليمات المختلفة للآلة وأسماء لأماكن التخزين الرئيسية، والجدير بالذكر أن الرموز المستخدمة تختلف باختلاف الشركات المنتجة للحاسبات وطرازها.

وتحتاج هذه اللغات إلى مترجم لترجمتها إلى لغة الآلة التي يتعامل معها الحاسب وذلك بواسطة برنامج خاص بالترجمة.

ومن مزايا هذه اللغات:

- \* تخفيض نسبة الأخطاء وسهولة تصحيحها.
- \* سهولة متابعة البرنامج وسهولة تعديله منطقيا إذا لزم الأمر ذلك.
- \* سهولة كتابة البرامج حيث انه يستخدم في كتابتها بعض الحروف والرموز بدلا من الأرقام.
- \* تخفيض الوقت اللازم لكتابة برنامج.

ومن عيوب هذه اللغات:

- \* أن لغات خاصة بالحاسبات التي وضعت من أجلها، حيث لا يمكن تشغيلها على أنواع مختلفة من الحاسبات، بل يجب أن تعمل على نوع واحد فقط من الحاسبات.
- \* تحتاج إلى برنامج للترجمة لتحويل اللغات الرمزية إلى لغة الآلة.
- \* صعوبة تعلمها لأنها تحتاج إلى مهارات وخبرات ومعرفة بالحاسب الذي سيتم تشغيل البرنامج عليه.

### 3- اللغات ذات المستوى الرفيع High Level Languages

لتلافى صعوبة البرمجة في اللغات الرمزية فقد قام مجموعة من المتخصصين في علوم الحاسب بتطوير تلك اللغات لجعلها أكثر سهولة في التعامل مع الحاسب، والغرض من هذا التطوير هو لأن تتم كتابة البرامج للحاسب ببعض اللغات العالية المستوى التي تتفق مجموعة الأوامر الخاصة بها مع لغات وأفكار الإنسان وتؤدي بالتالي إلى تخفيض الوقت اللازم لكتابة البرامج، كما يمكن استخدام البرامج المكتوبة بأحد هذه اللغات ذات المستوى الرفيع لحل مشاكل معينة على حاسبات عديدة حيث لا ترتبط هذه اللغات بنوع الحاسب.

ومن أهم اللغات عالية المستوى:

#### • لغة كوبول Cobol Language

يأتي اسم هذه اللغة اختصاراً لجملة لغة موجهة لإدارة الأعمال

#### COMMON BUSINESS ORIENTED LANGUAGE

وينتشر استخدام لغة كوبول على نطاق واسع عالمياً حيث تستخدم في البنوك وفي المنظمات الحكومية، وتستخدم على حاسبات كبيرة أو على حاسبات شخصية. وتتميز لغة كوبول بقدرتها على التعامل مع الملفات، لذا اشتهرت بأنها لغة أعمال.



## • لغة فورتران Fortran Language

يأتي اسم هذه اللغة اختصارا لجملة FORMULA TRANSLATIO ، وتستخدم في حل المسائل العلمية والرياضية بشكل عام، وبشكل خاص في برمجة حلول المعادلات المعقدة مثل معادلات التحليل الاقتصادي القياسي ومعالجة المسائل الهندسية.

## • لغة بيسك Basic Language

هي اختصار للعبارة BEGINNERS ALL- PURPOSE SYMBOLIC INSTRUCCION CODE أي اللغة متعددة الأغراض للمبتدئين، فهذه اللغة ونظرا للبساطة تعليماتها ومحدوديتها فأنها تعد لغة مناسبة للتعلم من قبل المبتدئين في عالم الحاسب الألى والبرمجة، وتستخدم هذه اللغة في معظم الحاسبات الشخصية، مما يدل على الانتشار الواسع لها، وتستخدم لغة الحديثة في قطاع واسع، إذ تستخدم في مجال الاعمال لقدرتها على التعامل مع الملفات، وكذا في العمليات الرياضية من قبل العلماء والمهندسين لامتلاكها كثيرا من الوظائف للقيام بمثل تلك العمليات المعقدة.

## • لغة C:

وتستخدم لكتابة برامج النظم التشغيلية، وتشبه الى حد كبير لغة التجميع وتمتاز بسرعتها الكبيرة، كما تملك مجموعة جيدة من التعليمات، كما أنها لغة قابلة للنقل من جهاز الى اخر لصغر الجزء الواجب نقله منها، وتعد لغة من اللغات التركيبية، وقد تطورت هذه اللغة فظهرت

نسخ محسنة مثل C+,C++,C#

ومن مزايا هذه اللغات:

1- سهولة التعلم وسهولة كتابة البرامج، حيث أنها تستخدم كلمات مشابهة لتلك التي تستخدم في الحياة اليومية للإنسان.

2- تستخدم كلمات وتعابير مشابهة للكلمات والتعابير التي يستخدمها الإنسان.

- 3- عدم ارتباط البرامج المكتوبة بأحد هذه اللغات بآلة معينة.
- 4- لا تحتاج عملية تغيير الحاسب بحاسب آخر تغيير كبير في البرامج.
- 5- سهولة اكتشاف الأخطاء وسهولة تعديلها وتصحيحها.
- 6- توفير الوقت اللازم لكتابة البرامج والوقت اللازم لتشغيلها على الحاسب.
- 7- إن الأمر الواحد في لغة عالية المستوى يكون مساويا لعدة أوامر من لغة الآلة، وبالتالي توفير الجهد والوقت الذي كان يقوم به محلي ومخططي البرامج في أثناء كتابتهم للبرامج بلغة الآلة أو باللغة الرمزية.

والجدير بالذكر أن البرنامج المكتوب بلغة عالية المستوى يجب أن يترجم إلى لغة الآلة قبل أن يتم تنفيذه، وتتم عملية الترجمة أوتوماتيكيا بداخل الحاسب، وفي الحقيقة أن المبرمج قليل الخبرة ربما لا يعي أن هذا الإجراء يتم داخل الحاسب، حيث انه يرى فقط البرنامج الأصلي وبيانات الإدخال والنتائج وبيانات الإخراج.

والمترجم ما هو إلا برنامج معد للحاسب يقبل أي برنامج مكتوب بلغة عالية المستوى كبيانات إدخال وينتج عنه برنامج مناظر مكتوب بلغة الآلة كمخرج - وتبعاً لذلك - فإن البرنامج الأصلي المكتوب بلغة عالية المستوى يسمى (البرنامج المصدري)، والبرنامج الناتج المكتوب بلغة الآلة يسمى (البرنامج الهدف)، ويجب أن يكون لكل حاسب مترجم لكل لغة من اللغات العالية المستوى الخاص بها، والجدير بالذكر أن استخدام المترجمات هي التي تمكننا من الحصول على التناسق والاستقلال عن الآلة مع لغة عالية المستوى.

هناك نوعان من برامج الترجمة بين لغات الحاسب، وتحويل تعليمات اللغة الى لغة الآلة، وهذان النوعان هما المترجم **Compiler** ، والمفسر **Interpreters** ، فكلاهما يأخذ تعليمات لغة البرمجة، ويترجمها الى نموذج مقروء عن طريق الحاسب، لكن يأخذ كل منهما طريقاً مختلفاً وان كانت نتائجهما النهائية واحدة، فكلاهما يأخذ البرنامج المصدر ويحوّله الى لغة الآلة.

يقوم كل من المفسر **Interpreter** ، والمترجم **Compiler** بترجمة شفرة المصدر **Source Code** للبرنامج المكتوب بلغة من لغات البرمجة، ويحولها الى لغة آلة ذات مستوى

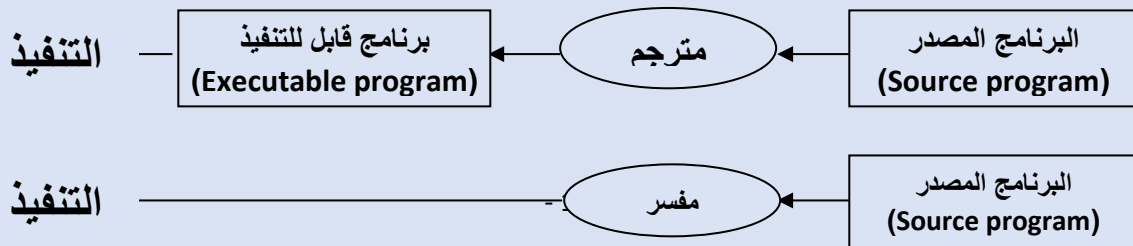
بسيط حتى يستطيع الحاسب ان يفهمها. تترجم بعض لغات البرمجة عن طريق المفسر مثل لغة **APL** ، ولغة **Basic** ، ويقوم المفسر بترجمة سطر، ثم يقوم بتنفيذه بعد ترجمته.

تتسم عملية التفسير بالبطء عن عملية الترجمة، فعملية تفسير البرمجة تفسر سطرا واحدا ففي كل مرة، ولما كان الجهاز عادة يقوم بأداء مهام متكررة مثل طباعة المئات من بيانات الموظفين فأن المهام المتكررة تعمل بصورة بطيئة، ففي كل مرة يكر سطرا لابد ان يقوم بتفسيره اولا، وبالرغم من ذلك فان الوقت المحسوب منذ تشغيل البرنامج حتى رؤية نتائجه يعتبر اقل ففي حالة استخدام المفسر منه ففي حالة المترجم.

يقوم عمل المترجم على اساس ترجمة **Translate** لترجمة البرنامج كله مرة واحدة، وعلى الرغم من طول فترة الاعداد الا انه يمكن الرجوع الى النسخة المترجمة في أي وقت عند الحاجة، ولن تكون هناك حاجة الى المترجم بعد انتهاء ترجمة البرنامج.

تأخذ ترجمة البرنامج خطوة إضافية عن خطوات تفسير البرنامج فيجب على المترجم أن يترجم البرنامج ثم ينتظر النتائج، ولن يعمل البرنامج الا بعد انتهاء كل الترجمة، وتفضل معظم الاعمال لغات البرمجة المترجمة لأنها بمجرد ترجمتها يتم تشغيل برامجها بصورة أسرع من تفسيرها، كما تمتاز البرامج المترجمة عن المفسرة بعامل الامان، فبمجرد انتهاء الترجمة لا يتم تغيير البرنامج بسهولة، وعند شراء برنامج مكتوب مترجم يمكن تشغيله دون رؤية برنامج المصدر نفسه.

كما ان الحاسب أي ما كان نوعه وحجمه لا يعمل بدون برامج، ويقوم المبرمجون بإمداده بتلك البرامج، ولا تنتهي مهمة المبرمج عند كتابة البرامج، فالبرامج قد تكتب مرة واحدة لكنه يعدل عدة مرات، وكلما تغيرت الاعمال، وتقدمت بيئة الحاسب لابد ان يتبعها تطور في البرامج ايضا، كما أن الهدف الأساسي لوجود البرامج هو الوصول الى الخطوات المنطقية لحل مشكلة أو مسألة معينة.



خطوات حل مشكله باستخدام الحاسب:

#### 1- تحديد المشكلة موضع الدراسة.

حيث يتم في تلك المرحلة الدقيقة تعريف المشكلة وتحديد أبعادها المختلفة، ويدخل ضمن نطاق هذه المرحلة عملية تحديد الهدف الذي يفترض الرغبة في تحقيق المستوي الأمثل لحل تلك المشكلة موضع الدراسة.

#### 2- جمع البيانات عن المشكلة موضع الدراسة.

ويتم في تلك المرحلة جمع كافة البيانات المتعلقة بالمشكلة محل الدراسة ويمكن الاستعانة بكافة البيانات المتعلقة بالمشكلة المخزنة ويدخل ضمن نطاق هذه المرحلة عملية تحديد الوقت اللازم لجمع البيانات وكذلك تصنيفها بالطرق الإحصائية التي تساعد في عملية معالجة البيانات والاستفادة منها فيما يلي من خطوات.

#### 3- تحديد البدائل الممكنة من حلول للمشكلة موضع الدراسة.

حيث يتم اجراء عملية حصر للحلول والبدائل الممكنة للمشكلة موضع الدراسة طبقا للبيانات التي يتم جمعها سابقا ويدخل في نطاق هذه المرحلة عملية تصفيه بدائل الحل واستبعاد البدائل غير المتنافسة ومن ثم حصر عملية الاختيار في عدد محدود من البدائل المتنافسة لحل المشكلة موضع الدراسة.

#### 4- متابعه تنفيذ القرار.


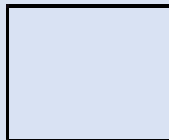

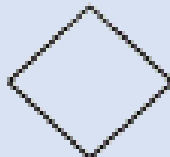
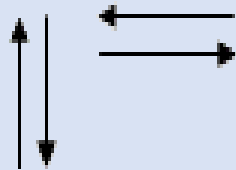
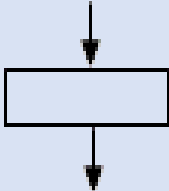

حيث أن عملية صناعه القرار لا تنتهي حتى باختيار البديل الأمثل بل تمتد لتهيئة الظروف والعوامل المناسبة اللازمة لتطبيق البديل الأمثل وذلك لبدء عملية التنفيذ ثم متابعة التنفيذ واكتشاف أي خلل أو انحرافات وعلاجها باستمرار. وتعتبر الخطوات السابقة بمثابة المنهج العلمي لصناعه القرار الأمثل سواء تم ذلك بطريقه بشريه أو باستخدام الحاسب الالي.

## الفصل الثانى

### خرائط التدفق

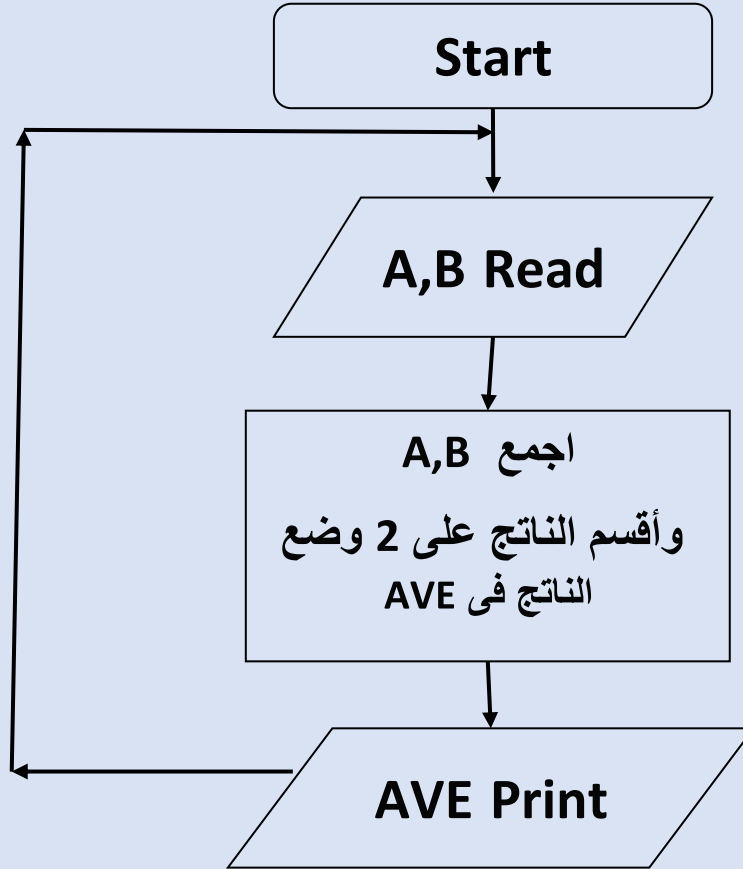
### Flow Chart

خرائط التدفق تمثل أحد الخطوات الأساسية التي يقوم بها مخطط البرامج لأعداد برنامج لهذه العملية، كما يسبق عمل خريطة التدفق لعملية ما تحليل وتفصيل (Break down) العملية الى مكوناتها الأساسية، وعادة ما يقوم محلل النظم بهذه الخطوة ثم يقوم مخطط البرامج بوضع خريطة التدفق لتوضيح التسلسل المنطقي لسير العمليات داخل العملية، أما عملية ترجمة خريطة التدفق في برنامج بلغة من لغات الحاسب فما هي الا عملية شبة الية، وفى معظم الأحوال يكون التأكد من سلامة منطق خريطة التدفق أهم بكثير من التأكد من سلامة الترجمة الى لغة الحاسب، ولسهولة تتبع خرائط التدفق عادة ما تكون خريطة التدفق هي اللغة المشتركة التي يتباحث بها كل من محلل النظم ومخطط البرامج قبل البدء الفعلي فى عملية ترجمة أية خرائط تدفق الى البرنامج بلغة معينة، كذلك فان تحديد وتصحيح الاخطاء يكون أكثر سهولة فى خرائط التدفق عن اجراء ذلك فى البر امج المكتوبة بلغة معينة، وكأى لغة تخاطب بين الأفراد فلخرائط التدفق أحرفها الأبجدية التي تتكون من بعض الأشكال والموضحة فيما يلى:

الرمز	الحدث الذي يمثله	مثال
	حدث طرفي Terminal لبيان بدء (Start) أو انتهاء (Stop) خريطة سير العمليات	<p>START</p> <p>STOP</p>
	عملية حسابية (Process)	<p>LET</p> <p>X+Y</p>
	إدخال / إخراج INPUT \ OUTPUT لبيان إدخال / إخراج معلومات من / إلى الحاسب	<p>PRINT</p> <p>Z</p> <p>INPUT</p> <p>X, Y</p>
	اتخاذ قرار Decision	<p>NO</p> <p>X=Y</p> <p>YES</p>
	اتجاه تدفق (سريان) Flow line	
	تكرار أو دوران Loop	<p>FOR</p> <p>I= 1 to 10</p>

والأمثلة التالية توضح إستخدام وكيفية أعداد خرائط التدفق.

مثال: إرسم خريطة تدفق Flowchart لعمل برنامج لإيجاد متوسط عددين A,B.



ونلاحظ في الخريطة السابقة تحويلة من آخر الخريطة الى خطوة الإدخال، وهذا يعنى أنه كلما أدخلت A,B فى الخطوة الثانية يتم تقدير المتوسط فى الخطوة التالية ثم يطبع المتوسط فى الخطوة رقم 4. ولا يتوقف البرنامج بل يتوجه الى الخطوة الثانية (الإدخال) ليسأل عن قيمة A,B الجديدة، ويطلق على هذا البرنامج برنامج غير محدود Unlimited Program لا يتوقف الا بانقطاع التيار عن الكمبيوتر وهو ما يشبه الى حد كبير برنامج ماكينة الصرافة ATM فى البنوك حيث تكون هذه الماكينات دائما فى وضع استقبال البيانات لتنفيذ عمليات السحب والإيداع المختلفة.

مثال: أرسم خريطة تدفق لحساب الفائدة المركبة على أموال المودعين تبعا للزمن المعطى  
قرين كل أيداع وبسعر فائدة 9%.

الحل:

قبل ان نبدأ فى اعداد الخريطة يجب ان نسأل أنفسنا:

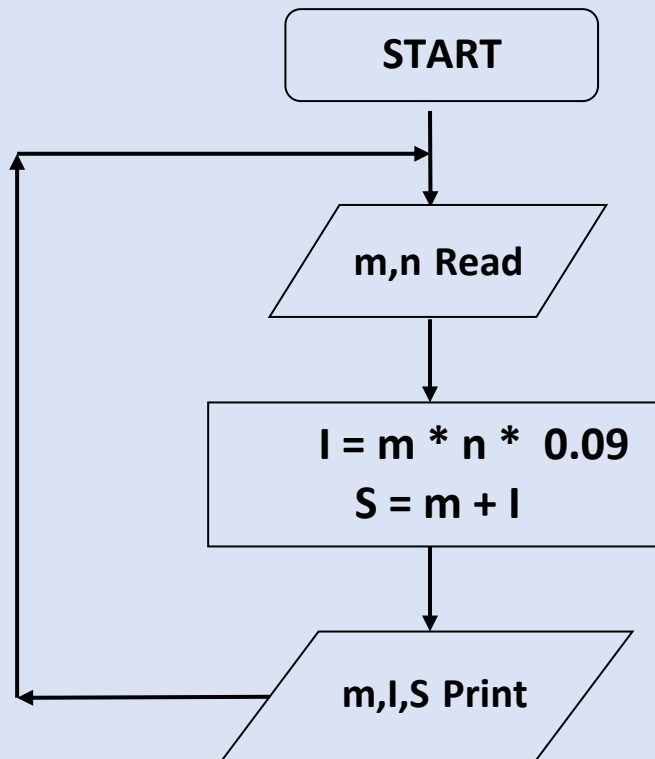
- ما هى المدخلات، والأجابة بالطبع هى مبلغ الايداع  $m$  والزمن  $n$ .
- ما هى العمليات التى سوف تجرى، والاجابة هى ايجاد قيمة الفائدة  $I$  من المعادلة التالية:

$$I = m * n * 0.09$$

ثم حساب الجملة من المعادلة التالية:

$$S = m + I$$

- ما هى المخرجات، والاجابة هى المبلغ  $m$  والفائدة  $I$  والجملة  $S$ .





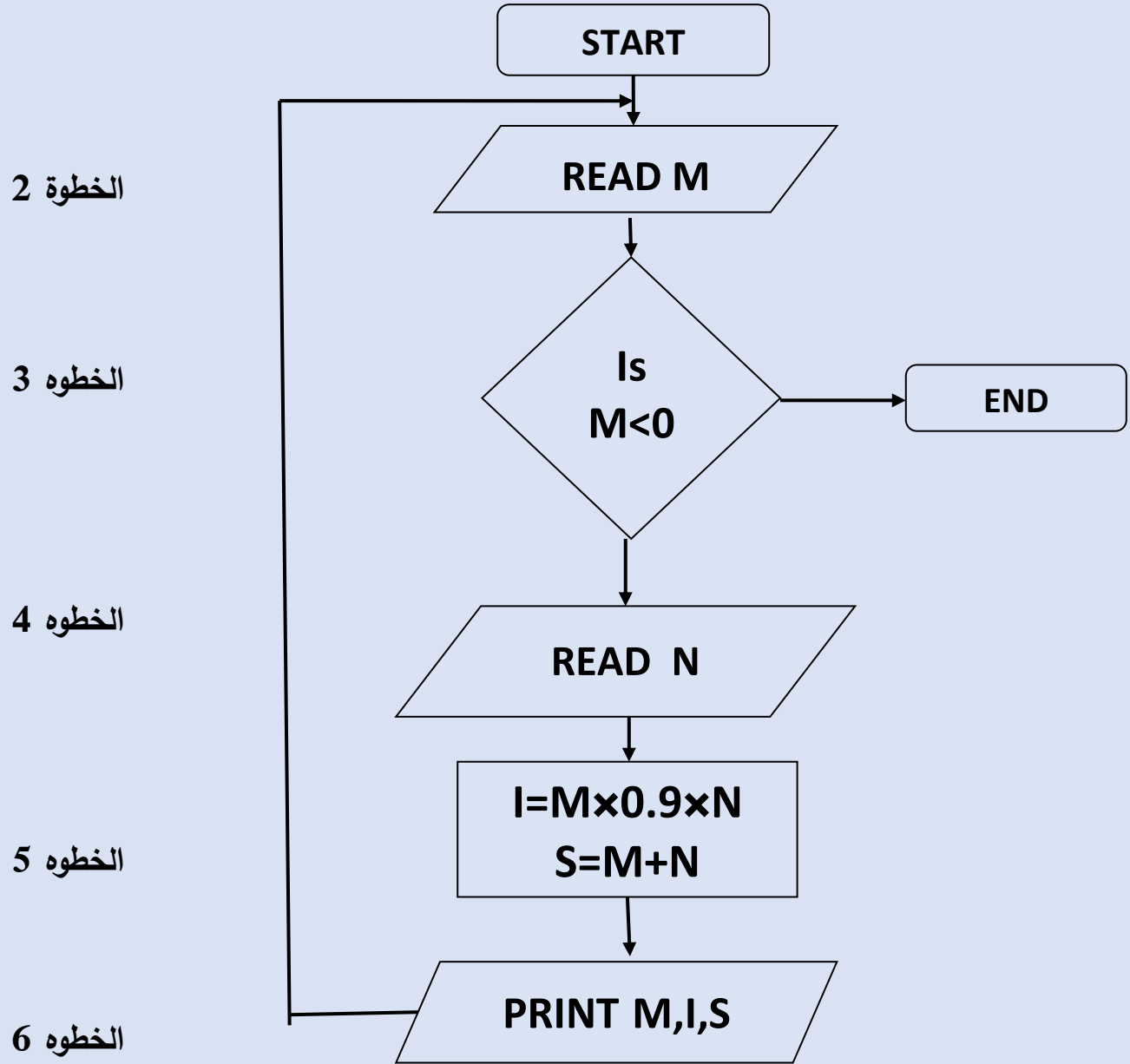
وبالنظر الى الخريطة نجدها كالسابقة تعطى برنامجا غير محدود، وهناك ثلاث طرق لا يقاف العمليات وهي:

### اولا: طريقة آخر بيان Last Record Technique

في المثالين السابقين بعد ظهور المخرجات في كل تشغيل يعود البرنامج للسؤال عن المدخلات الجديدة، فعند هذه النقطة نعطي الحاسب مدخل أخير (Last Record) ذو صفة مميزة ونجعل الحاسب يختبر هذه الصفة في كل بيان (مدخل) ففي حالة عدم وجودها يكمل العمليات ويتوقف في حالة وجودها، وهناك بعض الشروط الواجب توافرها في آخر بيان وهي:

1. ان يكون ذو قيمة منطقية من وجهة نظر الحاسب، فاذا كانت البيانات المدخلة لفظية فيمكن ان يكون اخر بيان لفظي وان كانت عددية فيجب ان يكون اخر بيان عددي.
2. ان تكون قيمة اخر بيان غير منطقي من وجهة نظر الواقع، ففي حالة المثال الحالي لو أردنا اختيار اخر بيان فلا بد ان تكون قيمة عددية حتى يقبلها الحاسب وان تكون سالبة فتكون غير منطقية لأنه لا يوجد ايداع بالسالب.

ويكون التعديل الواجب اجراءه على المثال السابق لاستخدام طريقة اخر بيان لإيقاف البرنامج في أي وقت هي:



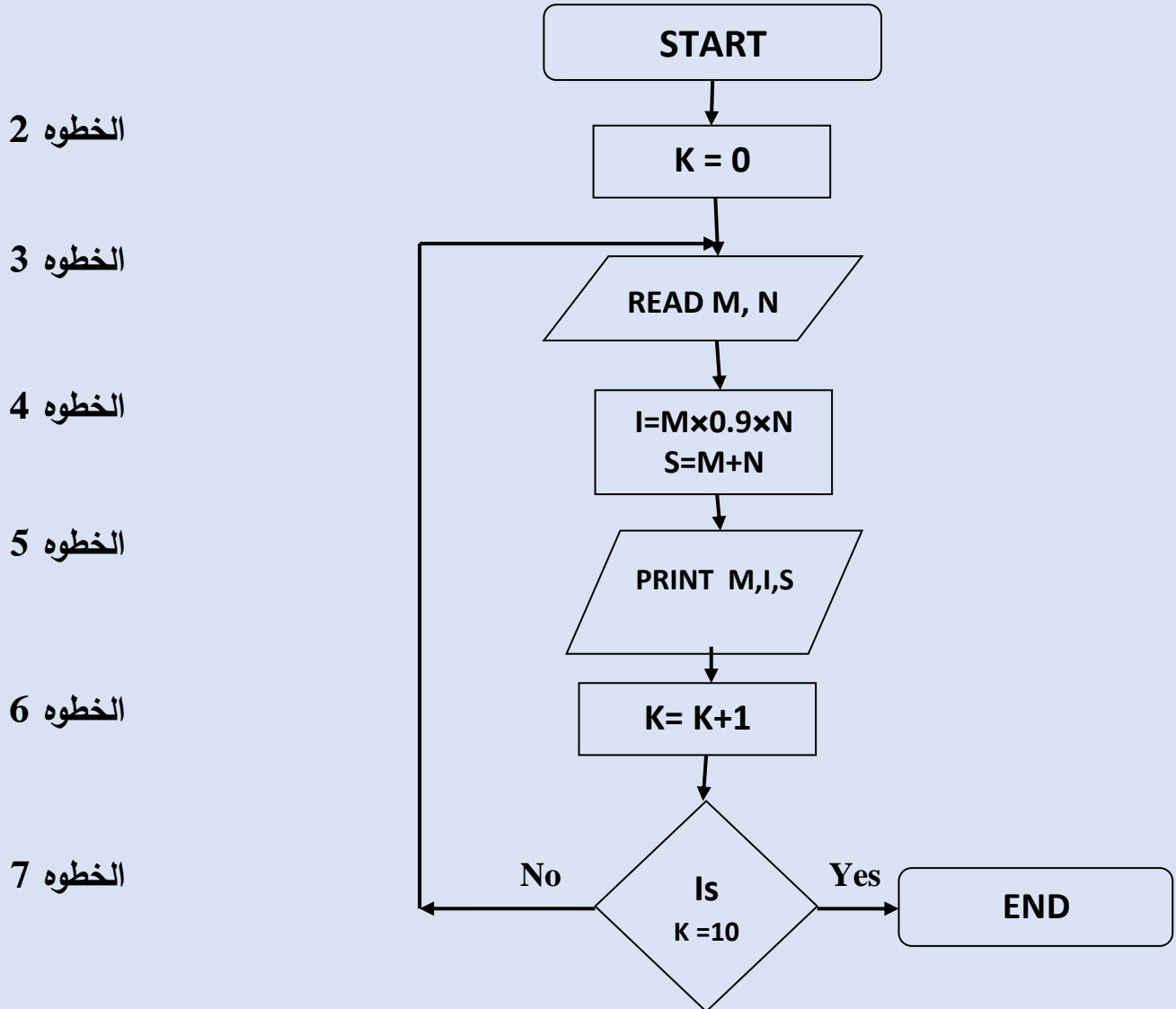
في الخطوة الثانية من الخريطة السابقة يطلب ادخال المبلغ  $m$ ، في الخطوة (3) تختبر  $m$  فان كانت سالبة يتجه الى الخطوة (8) مباشرة، حيث ينتهي البرنامج وان كانت خلاف ذلك فان البرنامج يتجه الى الخطوة رقم (4) حيث يطلب المدخل الثاني وهو الزمن  $n$  ثم يجرى المعالجة في الخطوة رقم (5) ثم في الخطوة رقم (6) تتم طباعة المخرجات اما في الخطوة السابعة فيتم تحويل التدفق الى الخطوة الثانية ..... وهكذا.

## ثانيا: طريقة العدادات

الشرط الأساسي لهذه الطريقة هو معرفة عدد العمليات مسبقا ففي المثال السابق يجب ان يحدد مسبقا عدد الايداعات المطلوب اجراء العمليات عليها، وعلى ان تتم الخطوات التالية:

1. تصفير العداد أي وضع القيمة صفر في العداد.
2. بعد اتمام العمليات يضاف (واحد) في العداد.
3. اختبار العداد هل وصل الى نهاية العدد.

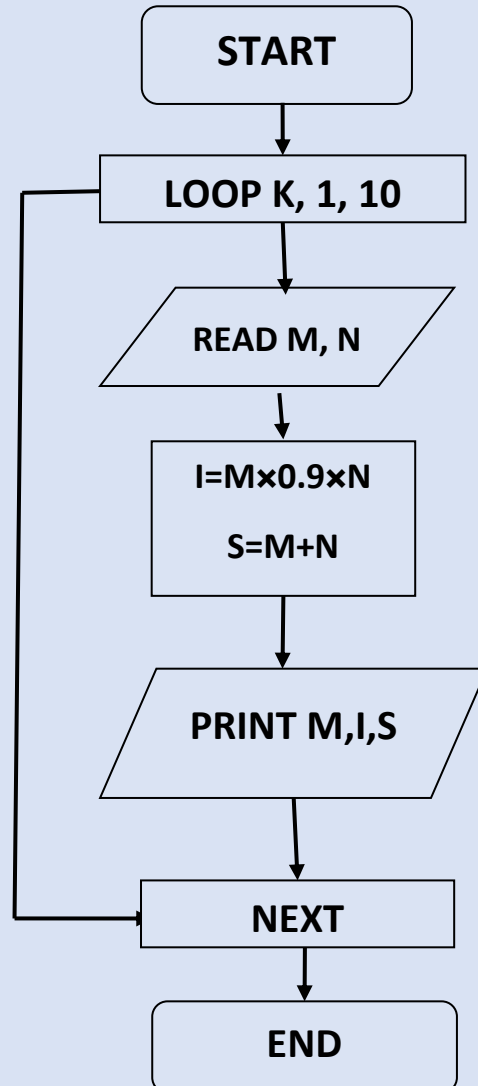
فلو فرضنا في المثال لسابق ان عدد الايداعات المطلوب عملها كان (10) فيكون التعديل الواجب اجراءه على الخريطة كما يلي:



ففي الشكل وضع صفر في المتغير K وذلك في الخطوه رقم 2 ويسير البرنامج من الخطوه 3 و4 حتى تتم طباعه المخرجات في الخطوه رقم 5. ويتم إضافة واحد الي العداد في الخطوه رقم 6 وهي الخطوه التي يليها اختبار العداد لتحديد مسار الخريطه.

### ثالثا: طريقة الدورات الانقلابية Loops

وهذه الطريقة تشبهه الى حد كبير طريقة العدادات، ففي هذه الطريقة يجب ان يكون عدد العمليات معلوم مسبقا، والشكل التالي لخريطة التدفق يبين التعديل الواجب عمله في الخريطة السابقة في حالة ما استخدمنا طريقة الدورات الانقلابية.



ونلاحظ في الشكل السابق وفي الخطوه رقم 2 تصميم LOOP والمتغير التابع لها يسمى K وقيمتها الابتدائية واحد وقيمتها النهائية 10 (القيمة المعلومة لعدد العمليات المطلوب تكرارها) أما الخطوه رقم 6 فهي المكملة للدورة الانقلابية. ويمكن شرح الدورة الانقلابية في المثال السابق بأن الخطوات المحصورة بين طرفي الدورة الانقلابية سوف تتكرر عشر مرات.

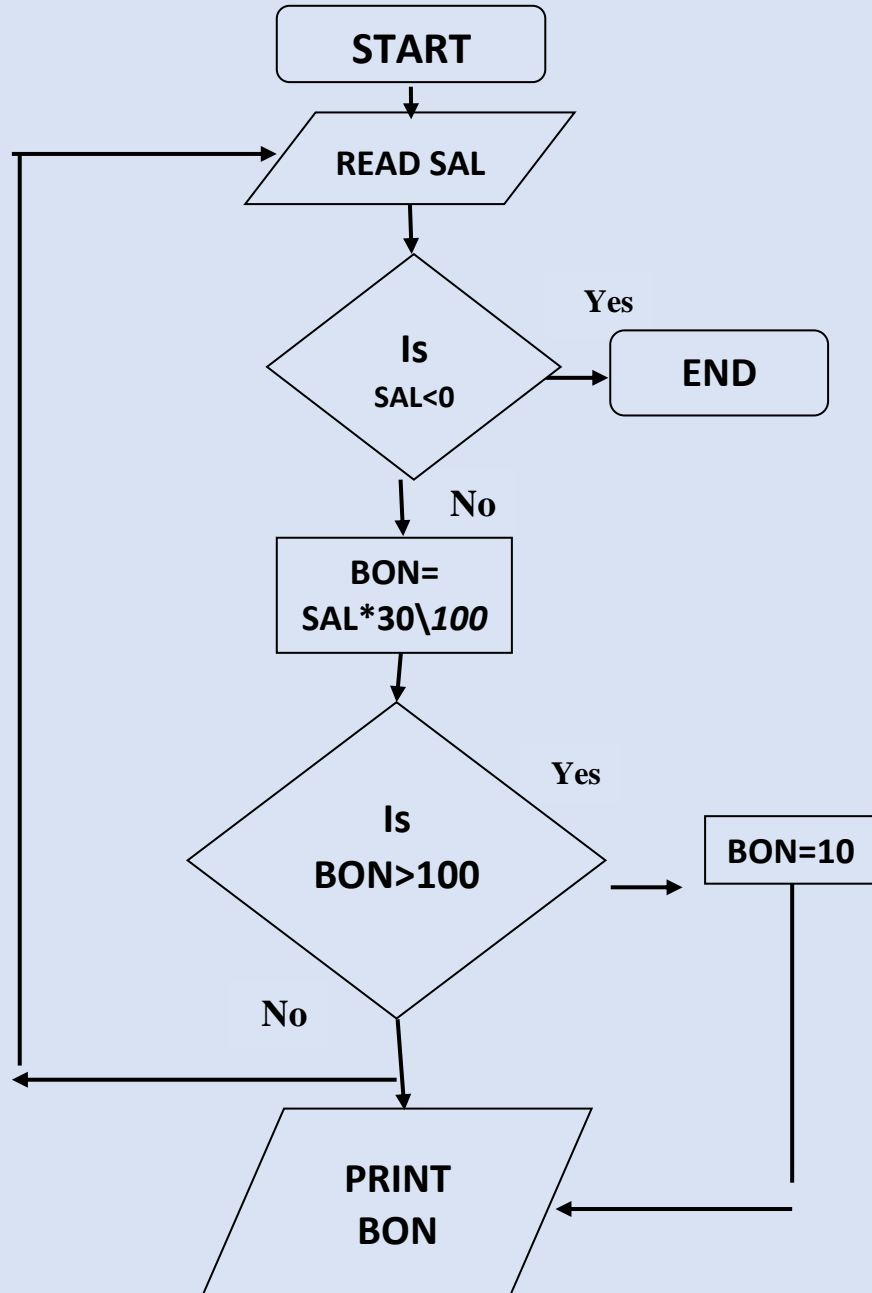
مثال: ارسـم خريـطة تدفق لعمـل برنامـج لتقديـر منـحة العمـال في شـهر ماـيو بواقـع 30% من المـرتب وبـحد اقـصى 100 جـنيه.

الحل:

نلاحظ هنا ان المرتبات SAL تمثل المدخلات والعمليات تتمثل في عمليات منطقية وهي اختبار إذا زادت المنحة عن 100 جنيه تخفض الى 100 جنيه، وتحسب المنحة BON كما يلي:

$$BON = SAL * 30/100$$

والمخرجات هي المنحة BON، وتكون خريطة التدفق كما يلي:

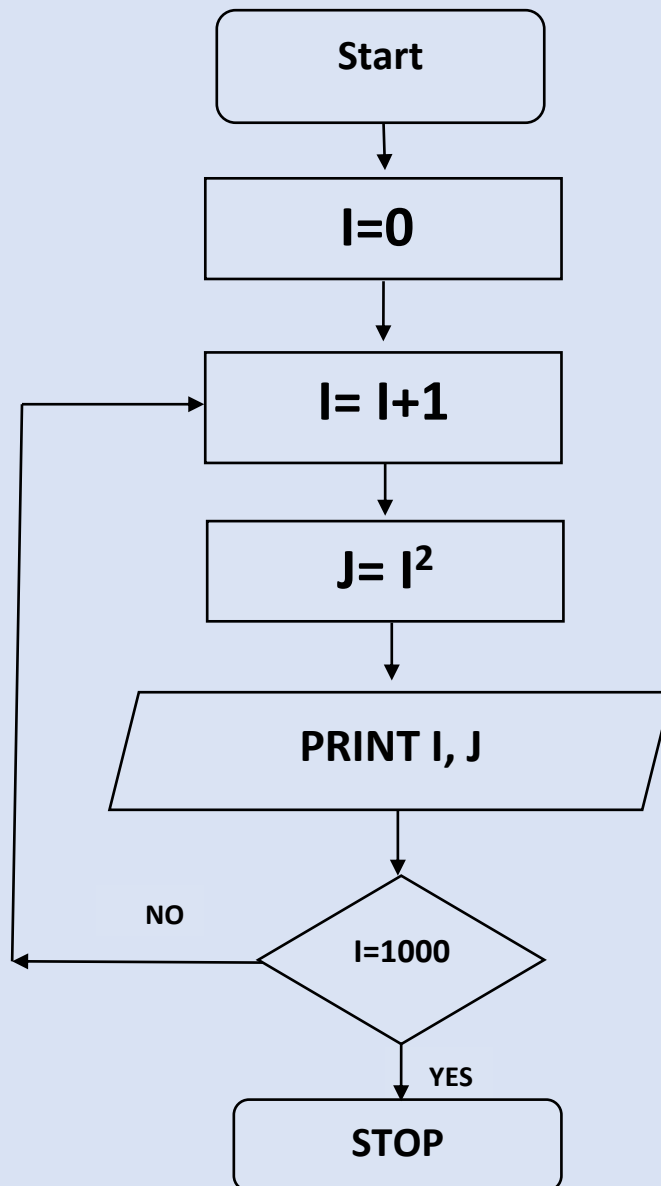


مثال: قم بإعادة حل المثال السابق باستخدام طريقة العدادات والدورات الانقلابية.

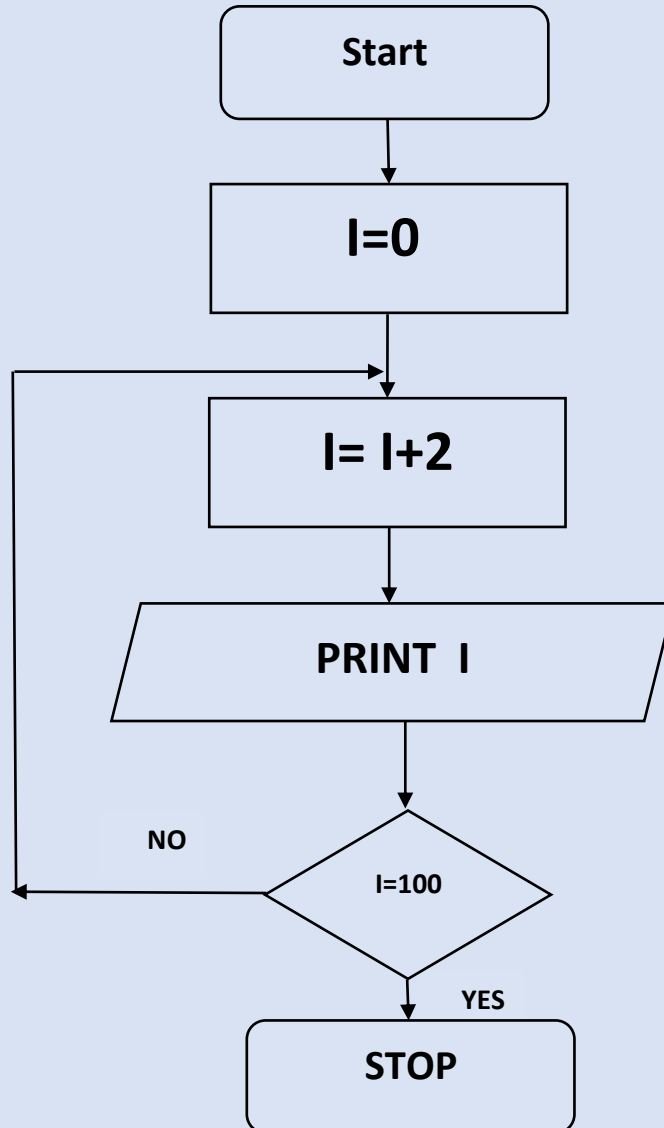
الحل متروك للطالب.

مثال: إرسم خريطة التدفق لبرنامج يوضح كيفية طباعه الأعداد الصحيحة ومربعاتها من 1

الي 1000



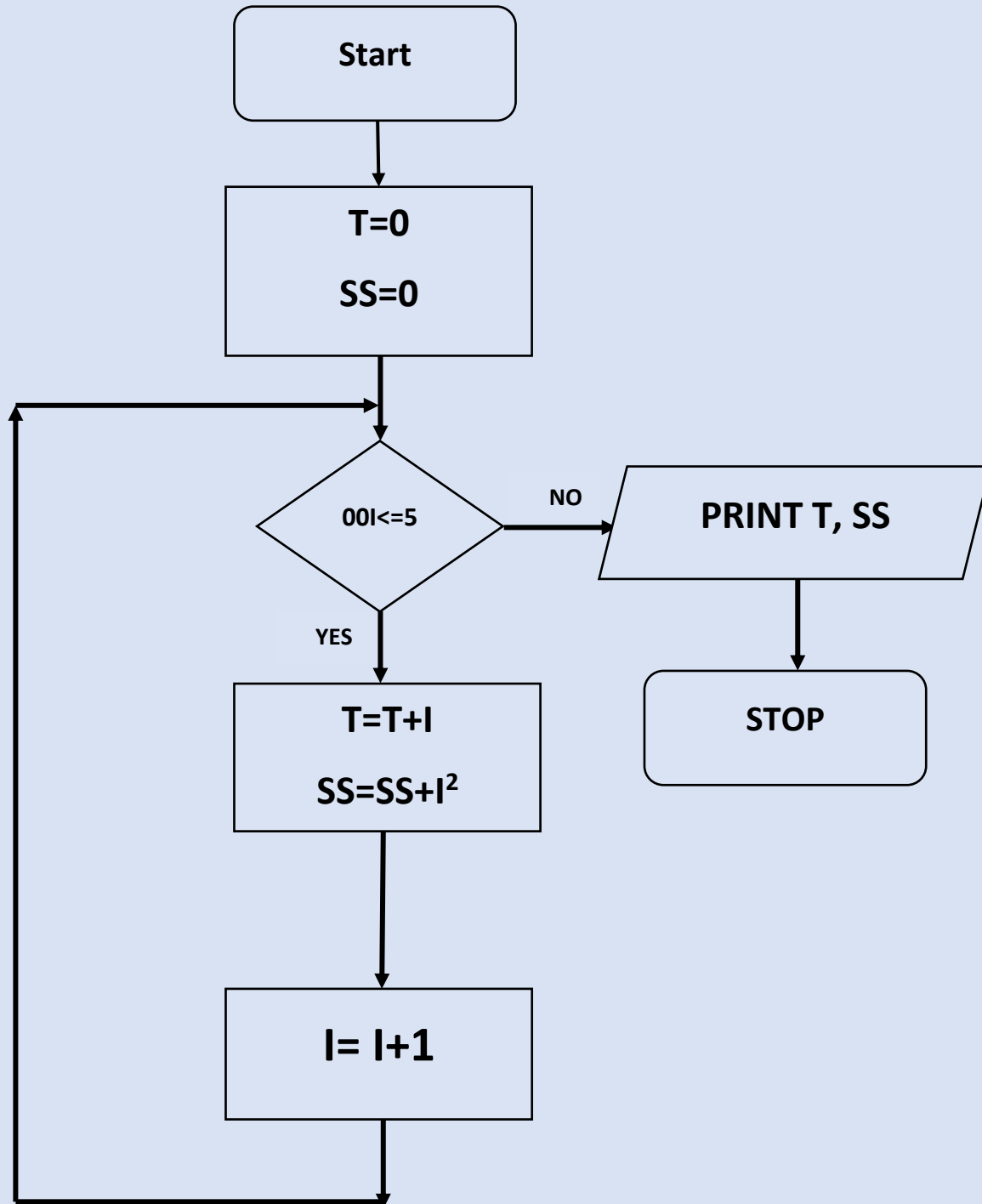
مثال: إرسم خريطة التدفق لبرنامج يوضح كيفية طباعه الأعداد الصحيحة الزوجية من 1 الي 100 باستخدام طريقه العدادات.



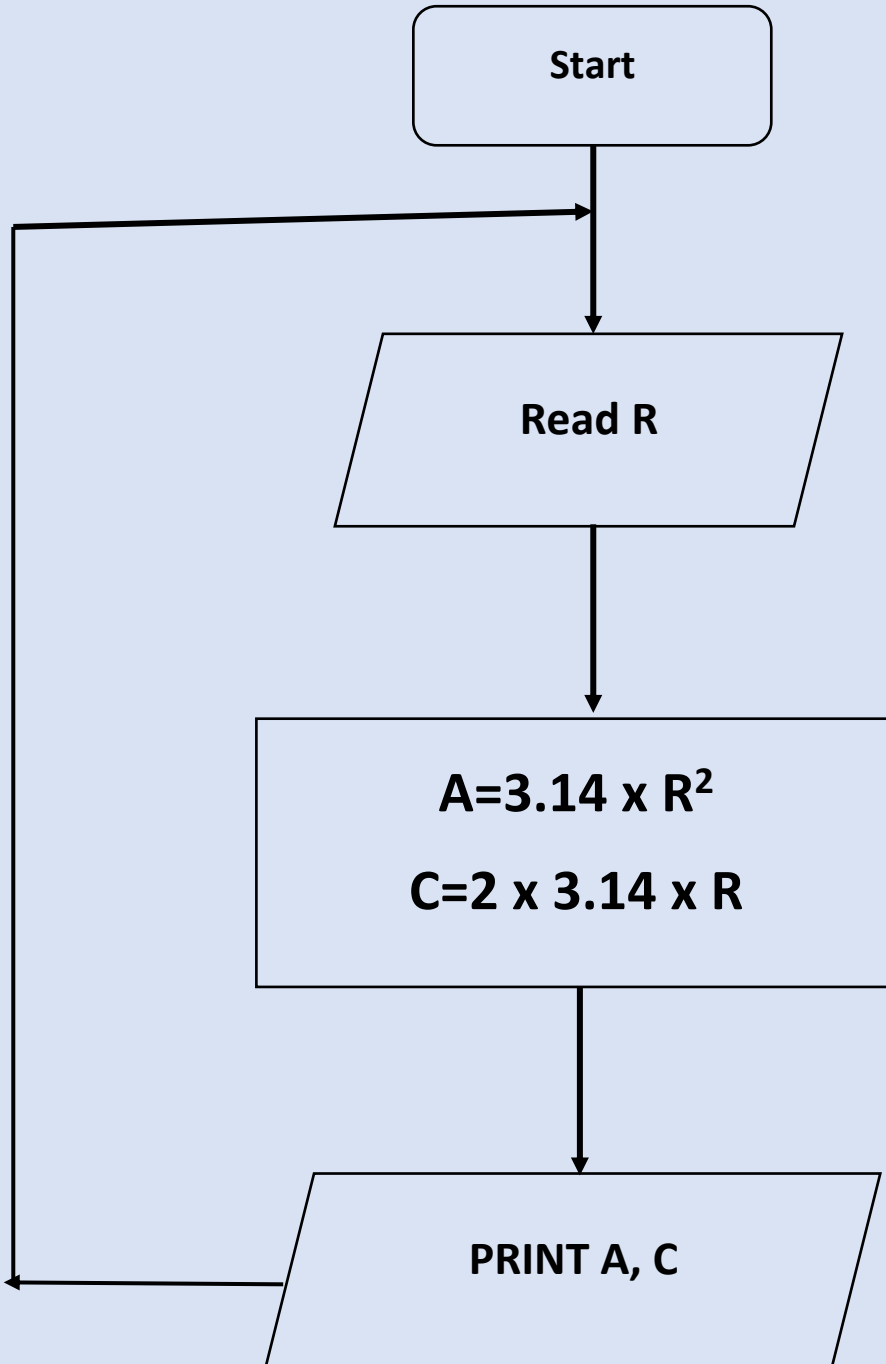


مثال: إرسـم خريـطه التدفق لبرنامج يوضح إيجاد مجموع الأعداد الصحيحة من

1 الي 500.



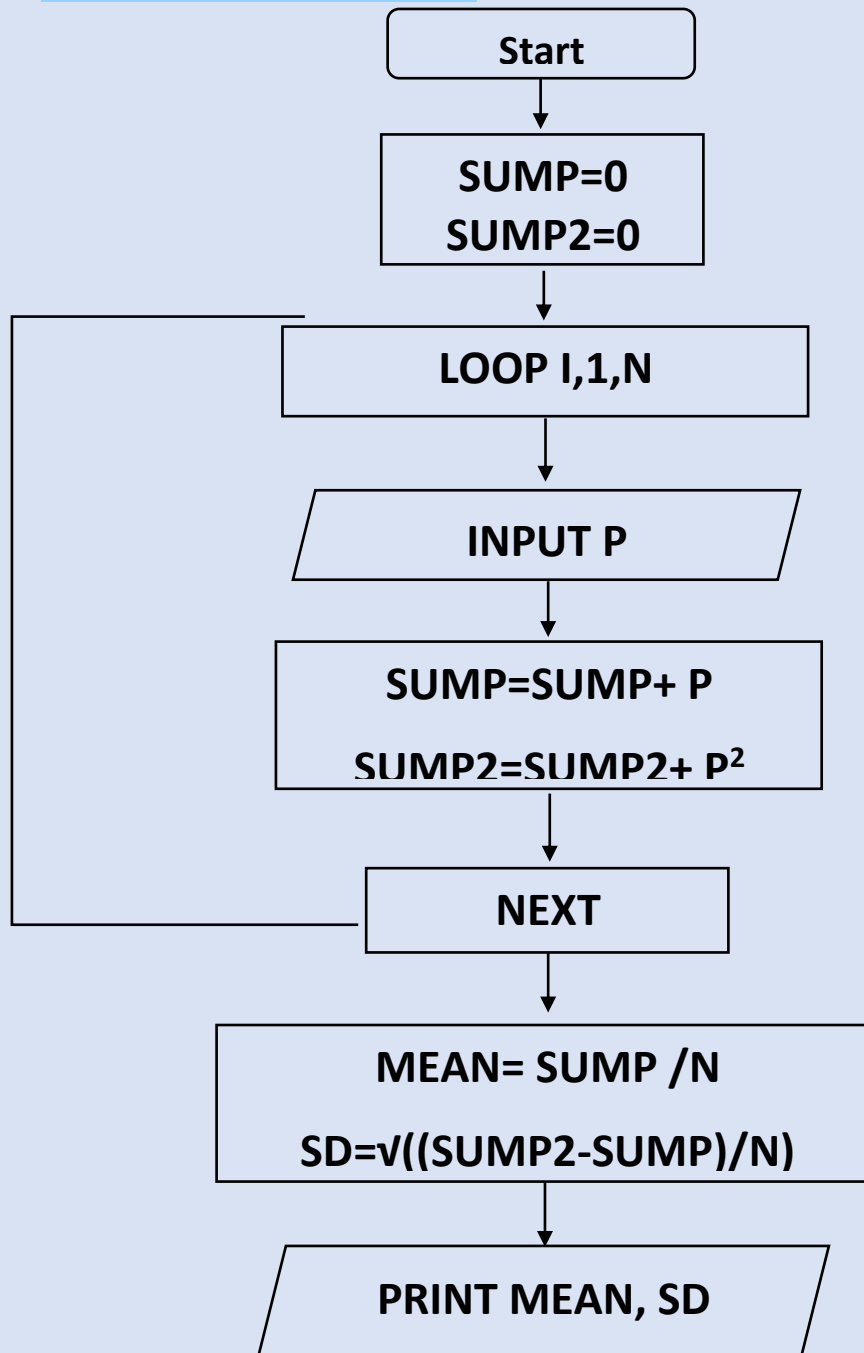
مثال: إرسم خريطة التدفق لبرنامج يوضح كيفية حساب مساحة ومحيط الدائرة نصف قطرها R.



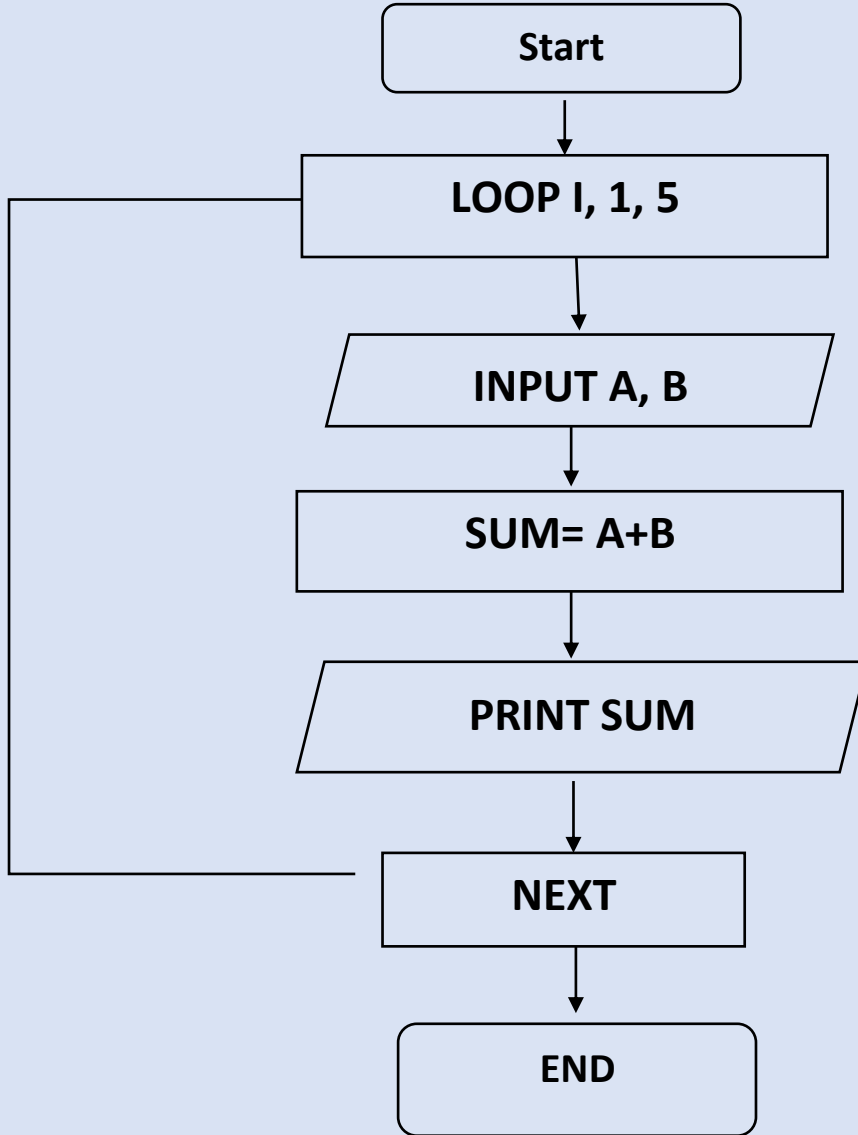
مثال: إرسـم خريـطه التدفق لبرنامج يوضح كيفية حساب الوسط الحسابي والانحراف المعياري لمجموعه من الأرقام عددها N.

فيما يلي معادلات الوسط الحسابي والانحراف المعياري كما درستها في الإحصاء.

$$\text{Avg} = \frac{\sum x_i}{N} \quad SD = \sqrt{\frac{\sum (x - \bar{x})^2}{N}}$$



مثال: إرسم خريطة التدفق لبرنامج يوضح كيفية حساب حاصل جمع خمسة أزواج من الأرقام باستخدام الحلقات التكرارية.



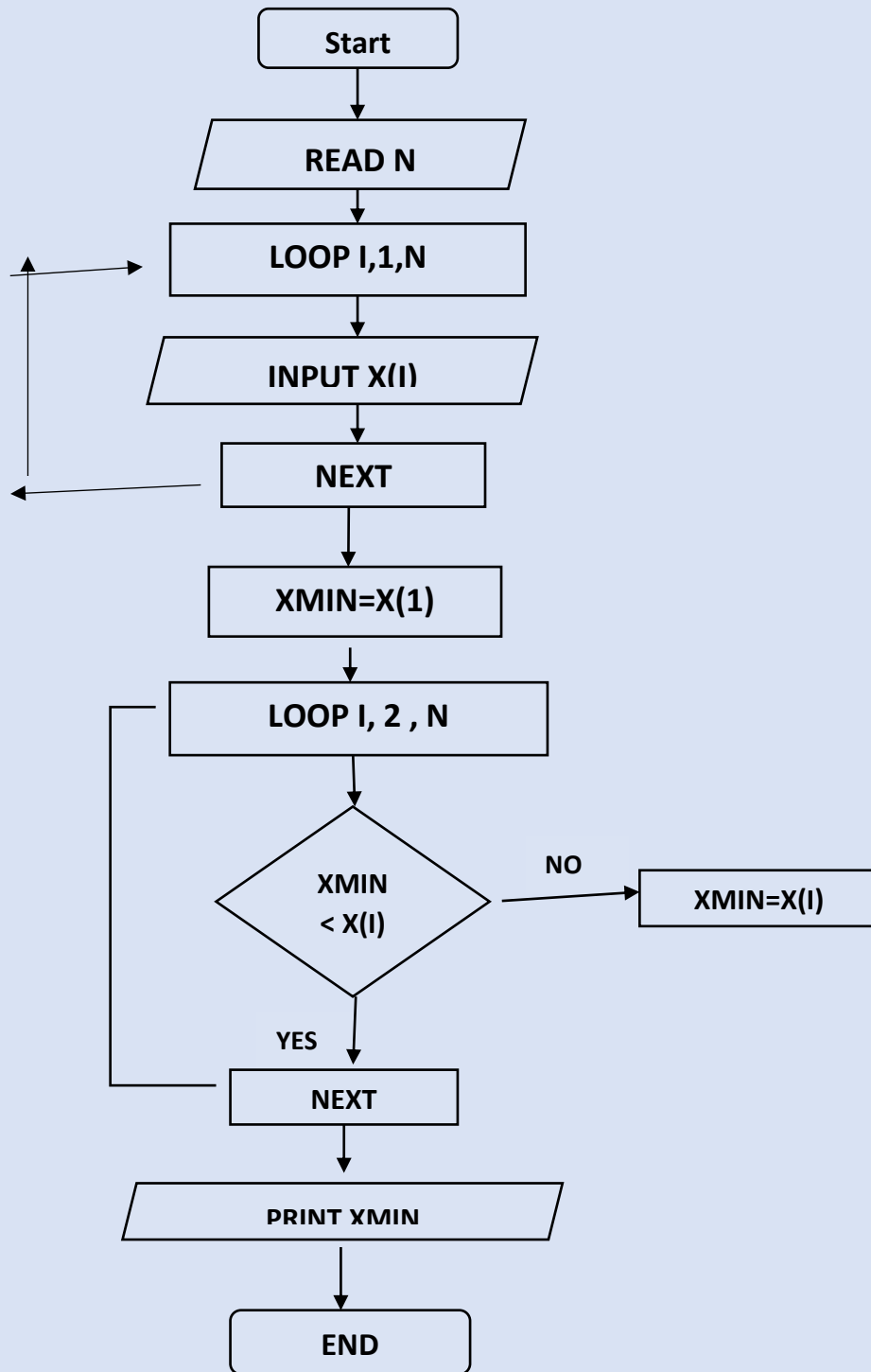
مثال: قم بحل المثال السابق باستخدام طريقة العدادات.

مثال: قم بحل المثال السابق باستخدام طريقة اخر بيان، هل من الممكن ذلك؟

مثال: ترغب إحدى الشركات فى شراء عدد من أجهزة الكمبيوتر ولذلك قامت إدارة المشتريات بالشركة بإجراء مناقصه تقدم لها عدد من موردي اجهزه الكمبيوتر. والمطلوب أن ترسم خريطه التدفق لتحديد أقل الأسعار التي تقدم بها أحد الموردين.

الحل:

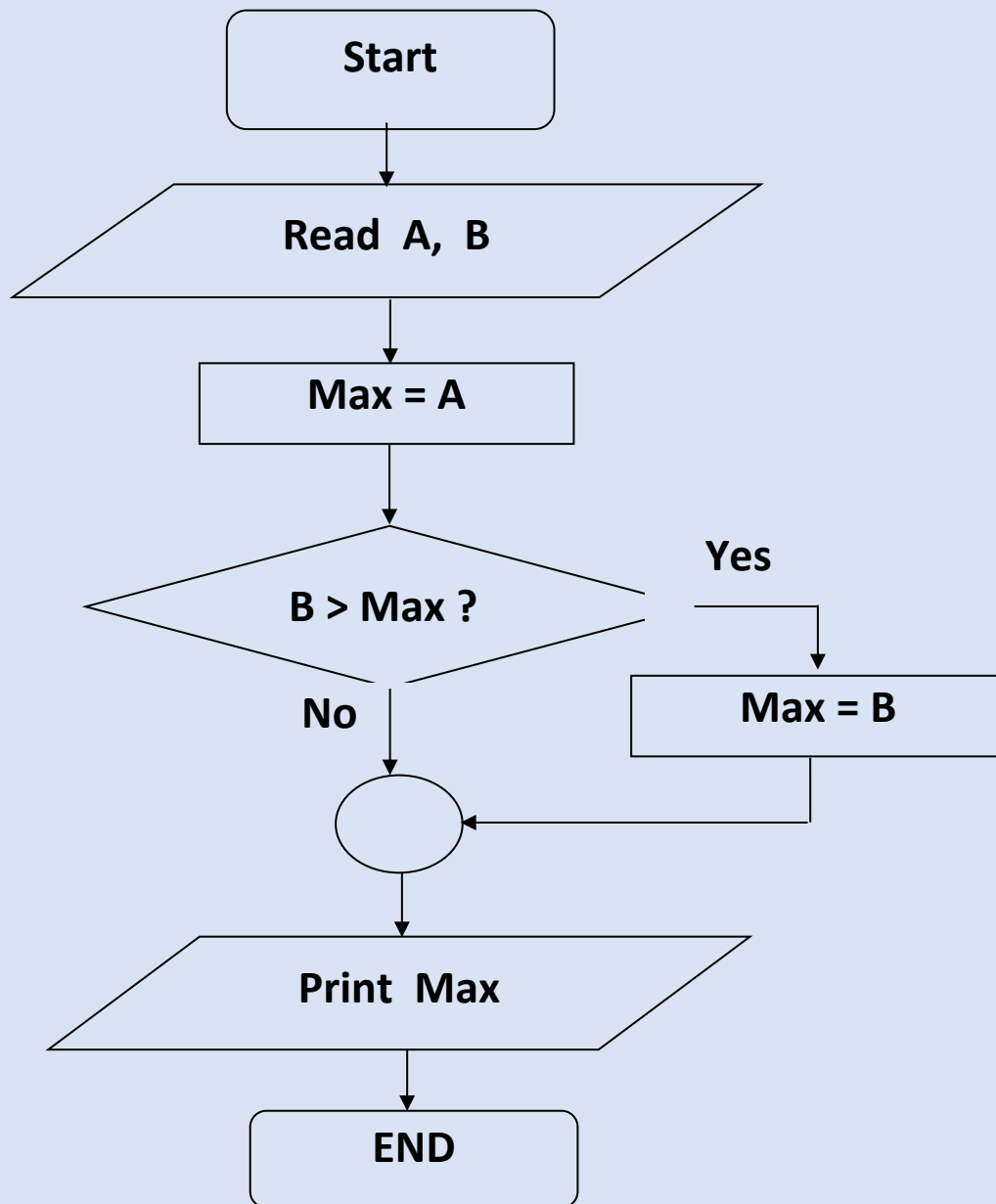
نفترض أن عدد الموردين الذين تقدموا للمناقصة هو  $N$  وأن الأسعار التي تقدموا بها كالاتي:  
 $X(1), X(2), X(3), X(4), X(N)$   
ونفترض أن أقل الأسعار هو  $X_{MIN}$  من بين كل الأسعار السابقة ولتحديد الشخص الذي سترسو عليه المناقصة (صاحب السعر  $X_{MIN}$ ) تكون الخريطه كالتالي:

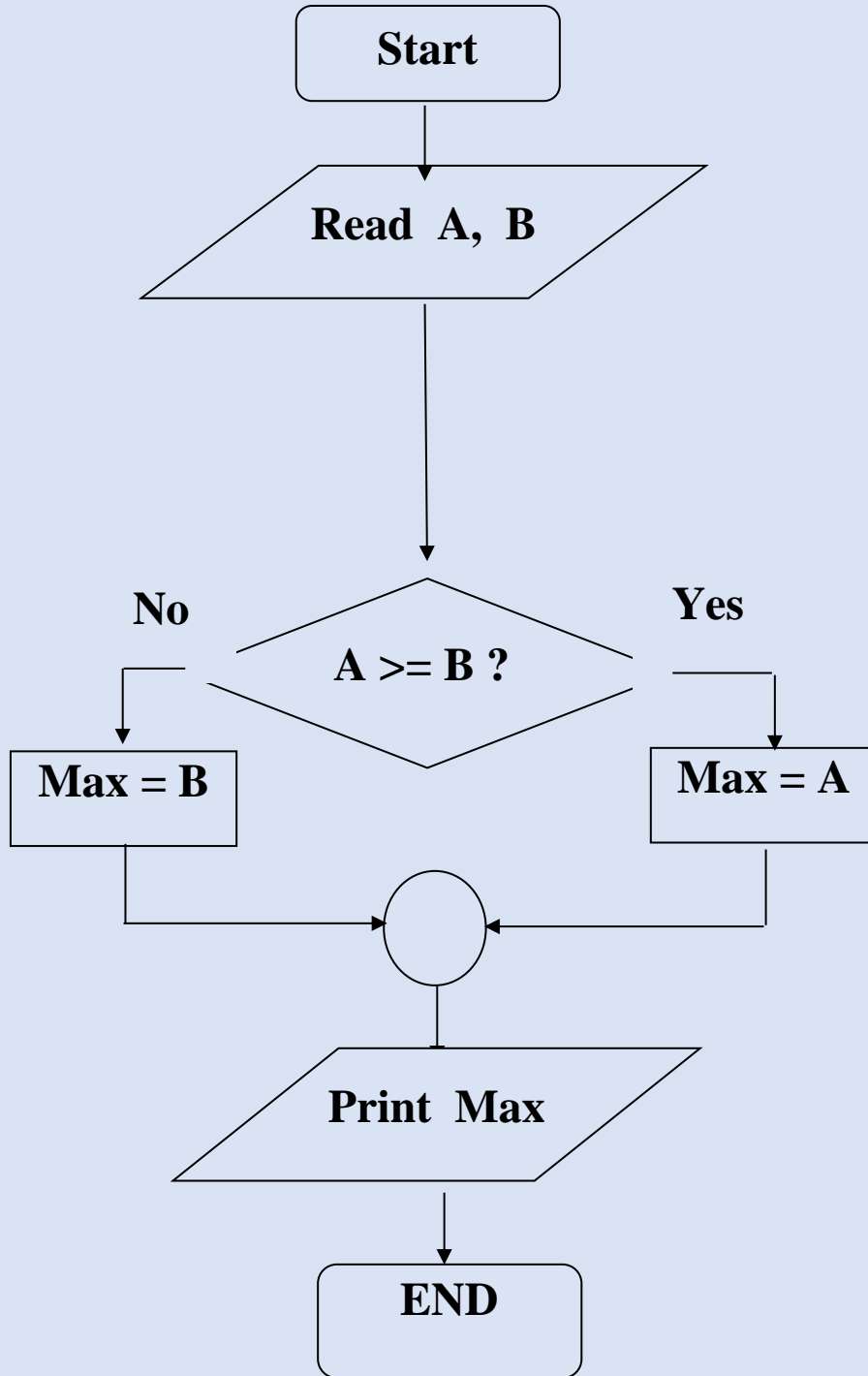


مثال: إرسم خريطه التدفق التي توضح سير العمليات لبرنامج يقوم بطباعه القيمة الأكبر

من بين العددين الصحيحين A, B.

الحل: الطريقة الأولى



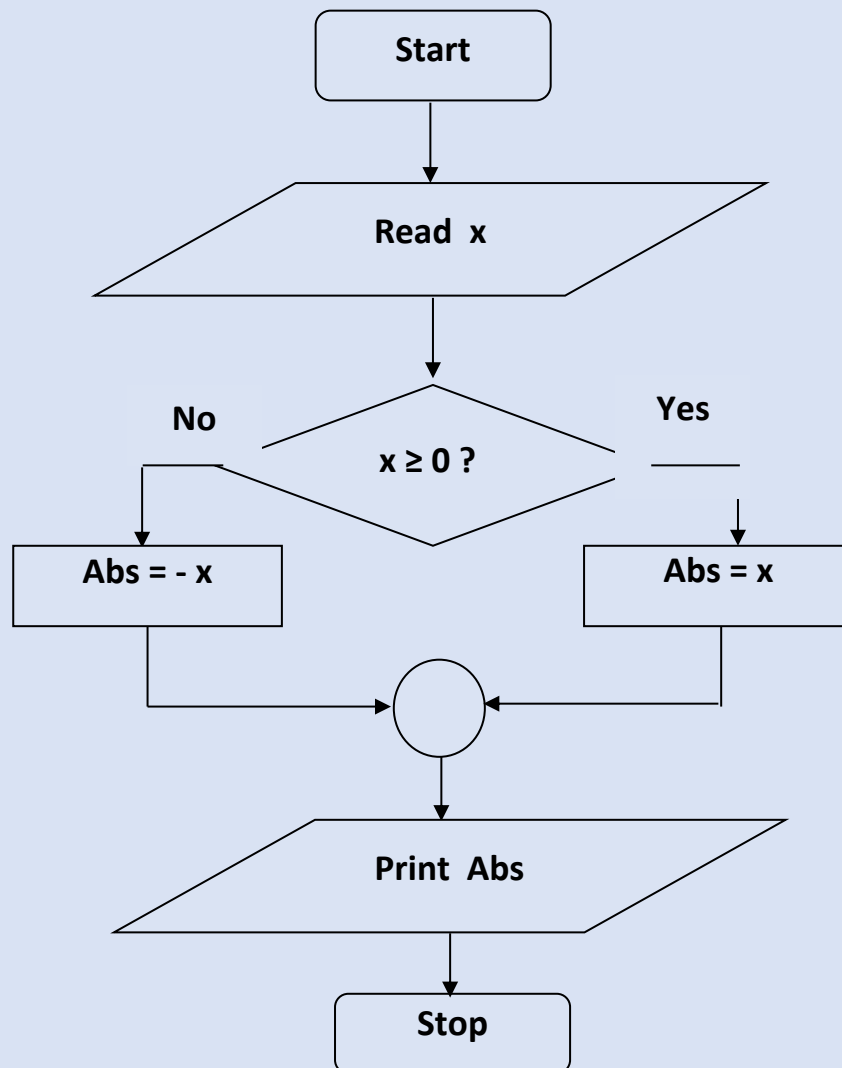




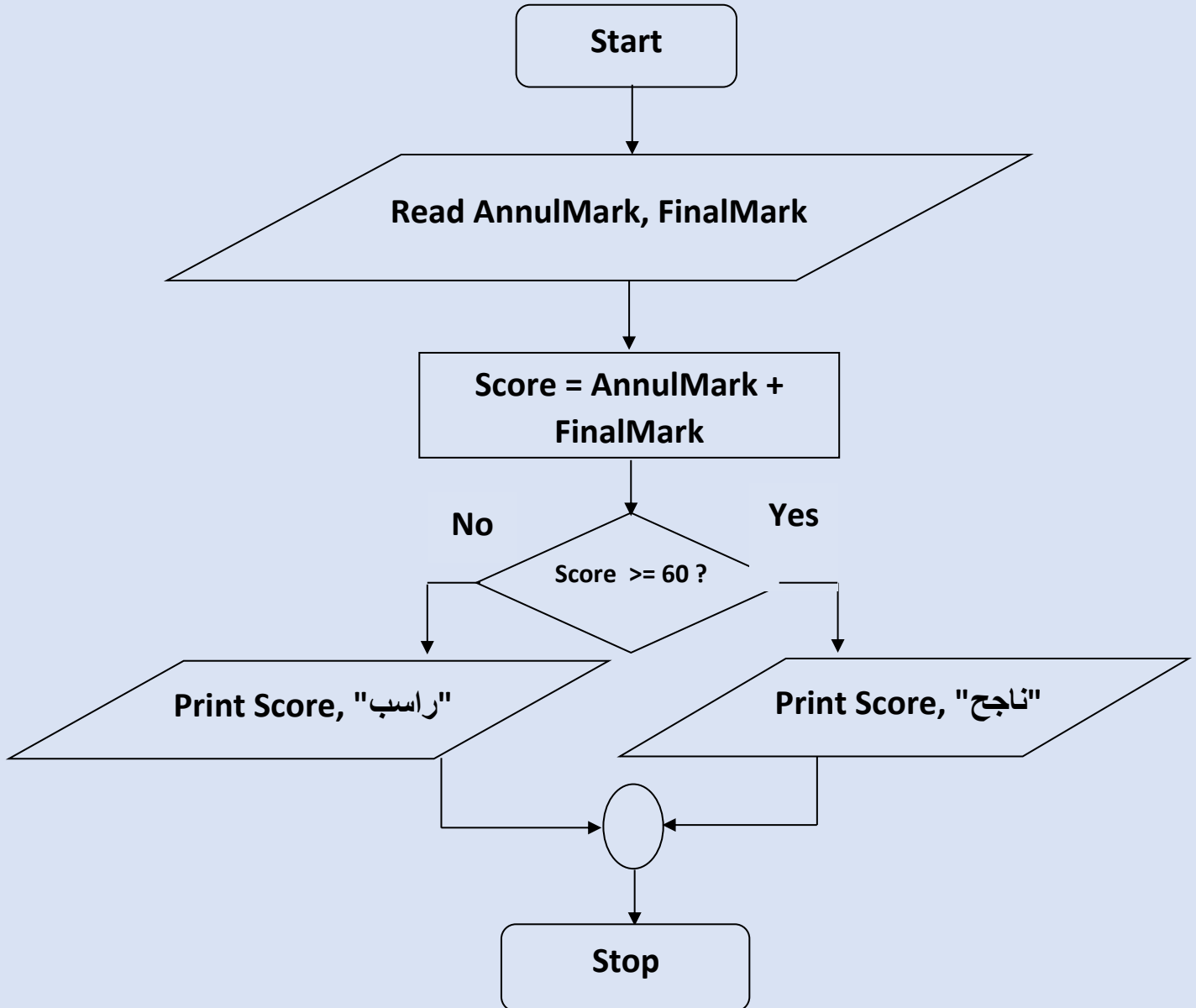
مثال: إرسم خريطه التدفق التي توضح سير العمليات لبرنامج يقوم بحساب القيمة المطلقة (Absolute Value) للعدد X باستخدام الصيغة التاليه:

$$Abs(x) = |x| = \begin{cases} x & \text{if } x \geq 0 \\ -x & \text{if } x < 0 \end{cases}$$

الحل:



مثال: ارسم خريطه التدفق التي توضح سير العمليات لبرنامج يقوم بحساب مجموع الطالب (Score)، وذلك بجمع درجة أعمال السنة (AnnualMark) ودرجة الاختبار النهائي (FinalMark) ثم طباعة المجموع وهل الطالب ناجح أم راسب، علما وأن الطالب لا يعتبر ناجحا إلا إذا كان المجموع يساوي أو يفوق 60 بالمئة.



مثال: - المطلوب رسم خريطة تدفق لحساب صافي المرتبات لعدد 1000 موظف في إحدى الشركات وإذا علمت ان المرتب الاساسي BS والاستقطاعات D يتم حسابها طبقا للقواعد التالية:

$$BS \leq 300 \quad D = 5 \%$$

$$BS \leq 500 \quad D = 7 \%$$

$$BS > 500 \quad D = 10 \%$$

وكان المطلوب طباعة اسم الموظف والراتب الاساسي وقيمة الاستقطاعات وصافي المرتب وكذلك اجمالي الاستقطاعات واجمالي الرواتب الصافية لجميع الموظفين.

#### الحل

يجب تحليل المشكلة والتعرف على المدخلات والمخرجات والعمليات المطلوبة قبل البدء في رسم خريطة التدفق.

1- ان مدخلات هذه المشكلة هي اسم الموظف، الراتب الاساسي BS, عدد الموظفين ونسب الاستقطاعات المطلوبة وهي D1, D2, D3.

2- العمليات المطلوبة بعضها عمليات حسابية والبعض الاخر عمليات منطقية فالعمليات الحسابية هي حساب صافي المرتب لكل موظف، ايجاد مجموع المرتبات الصافية، ايجاد مجموع الاستقطاعات. اما العمليات المنطقية هي مقارنة الراتب الاساسي بحدود نسب الاستقطاعات لتحديد نسبة الاستقطاع المطلوب تنفيذها.

3- مخرجات المشكلة هي اسم الموظف، الراتب الاساسي BS, الاستقطاعات D, الراتب الصافي NS, اجمالي الرواتب الصافية، اجمالي الاستقطاعات.

من هذا التحليل نجد انه لحل هذه المشكلة تحتاج الي توافر ما يلي:

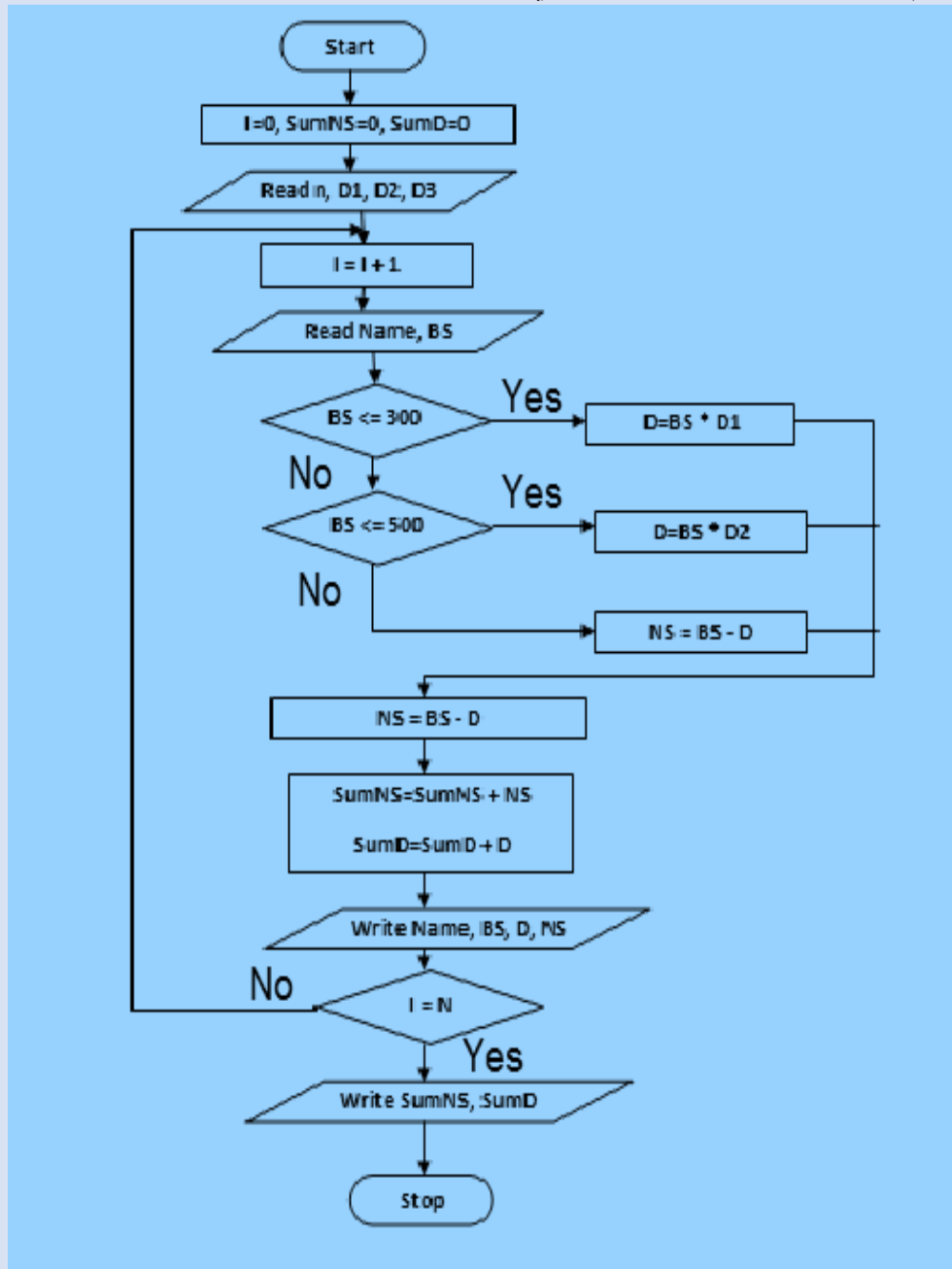
1- عداد لحصر الموظفين وليكن  $I$ .

2- مخزن جمع يبدأ بالقيمة صفر لإيجاد مجموع الرواتب الصافية

وليكن  $SumNS = 0$ .

3- مخزن جمع يبدأ بالقيمة صفر ايجاد مجموع الاستقطاعات وليكن  $SumD = 0$ .

ويمكن تصميم خريطة التدفق بالشكل التالي:



تفسير خطوات خريطة التدفق: -

- 1- تم تنفيذ البداية باستخدام Start.
- 2- تم تخصيص عداد  $I = 0$  لحصر عدد الموظفين ومخزن لجمع الرواتب الصافية  $SumNS = 0$  ومخزن لجمع الاستقطاعات  $SumD = 0$ .
- 3- تم ادخال عدد الموظفين  $n$  ونسب الاستقطاعات  $D1, D2, D3$  بالترتيب  $10\%, 7\%$ ,  $5\%$  وذلك باستخدام الامر `Read n, D1, D2, D3`.
- 4- تم تشغيل العداد وزيادة قيمته المبدئية بمقدار واحد وذلك باستخدام الامر `I = I + 1`.
- 5- تم ادخال اسم الموظف الاول وراتبه الاساسي `Read Name, BS`.
- 6- تتم عملية مقارنة الراتب الاساسي  $BS$  بالقيمة  $300$  حيث ان الشرط هو  $BS \leq 300$  فاذا كانت الاجابة YES فيتم حساب قيمة الاستقطاعات باستخدام المعادلة  $D1 = BS * D1$  وإذا كانت الاجابة NO يتم الانتقال الي الخطوة التالية.
- 7- تتم عملية مقارنة الراتب الاساسي  $BS$  بالقيمة  $500$  حيث ان الشرط هو  $BS \leq 500$  فاذا كانت الاجابة YES فيتم حساب قيمة الاستقطاعات باستخدام المعادلة  $D = BS * D2$  وإذا كانت الاجابة NO يتم الانتقال الي الخطوة التالية.
- 8- يتم حساب قيمة الاستقطاعات بالمعادلة  $D = BS * D3$  ويلاحظ هنا ان هناك ثلاثة شروط لحساب الاستقطاعات وبالتالي فان عدد مرات استخدام IF Statement يجب ان يكون أصغر من عدد الشروط بمقدار واحد.
- 9- في حالة تحقق الشرط الاول خطوة رقم (6) او تحقق الشرط التالي خطوة رقم (7) او حالة تنفيذ الخطوة رقم (8) يتم التوجه الي الخطوة التالية وهي حساب صافي الدخل بالمعادلة  $NS = BS - D$  وذلك للموظف الاول.
- 10- يتم اضافة صافي الدخل الي مخزن جمع الرواتب باستخدام المعادلة  $SumNS = SumNS + NS$  وكذلك اضافة الاستقطاعات الي مخزن الاستقطاعات باستخدام المعادلة  $SumD = SumD + D$ .

11- يتم طباعة اسم الموظف، الراتب الاساسي BS, الاستقطاعات D, الراتب الصافي NS.

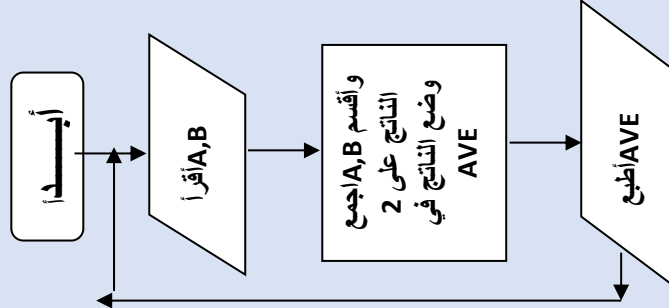
12- يتم التحقق من ان رقم هذا الموظف والذي يشار اليه بالرمز I يساوي اجمالي عدد الموظفين n ام لا والشرط  $I = n$  فاذا لم يتحقق الشرط فهذا يعني انه لم يتم الانتهاء من حساب مرتبات جميع الموظفين وبالتالي يتم التوجه الي الخطوة رقم (4) لزيادة قيمة العداد بمقدار واحد وقراءة بيانات الموظف الثاني وتشغيلها وهكذا اما في حالة تحقق الشرط  $I = n$  فيتم التوجه الي الخطوة التالية.

13- يتم طباعة اجمالي الرواتب الصافية SumNS, واجمالي الاستقطاعات SumD.

14- يتم التوقف باستخدام Stop.

## تمارين

1 - الخريطة التالية تمثل سير العمليات لبرنامج غير محدود



المطلوب: شرح طريقتين مختلفتين لإيقاف العمليات موضحا ذلك باستخدام الخريطة السابقة.

2 - ارسم خريطة التدفق لإيجاد ناتج المعادلة التالية:

$$f(x) = \begin{cases} X & X \geq 0 \\ -X & X < 0 \end{cases}$$

3 - ارسم خريطة التدفق لإيجاد ناتج المعادلة التالية:

$$f(x) = \begin{cases} 2X^2 + 3 & X > 0 \\ X + 8 & X = 0 \\ 3X - 9 & X < 0 \end{cases}$$

4 إرسم خريطة التدفق لإيجاد مكعبات الأرقام الطبيعية من 1 الي 750.

5 إرسم خريطة التدفق لإيجاد مجموع الأرقام المحصورة ما بين 1 و 900.

6 إرسم خريطة التدفق لقراءة أسماء عدد M من الطلبة ومجموع درجاتهم في امتحان نصف العام

والمطلوب أن تحسب خريطة التدفق التقدير العام طبقا للاتي:

GPA < 50

راسب F

50 <= GPA < 65

مقبول P

65 <= GPA < 80

جيد G

80 <= GPA < 90

جيد جدا VG

GPA >= 90

ممتاز E

علما بأن GPA هو مجموع درجات الطالب.

7 إرسم خريطة التدفق لإيجاد الأرقام الأولية المحصورة بين 1 و100 , الرقم الأولي هو الرقم الذي لا يقبل القسمة إلا على نفسه أو الواحد الصحيح، كمثال الرقم 13 هو رقم أولي وكذلك الرقم 7.

8 إرسم خريطة التدفق لإيجاد أصغر عمر من مجموعه تحتوي على أعمار 20 طفل.

9 إرسم خريطة تدفق لحل معادله من الدرجة الثانية:

$$aX^2 + bX + C = 0$$

10 أرسم خريطة التدفق لبرنامج يقوم بحل معادلة من الدرجة الأولى

$$ax + b = 0$$

11 - ارسم خريطة سير العمليات لحساب قيمة كل من المتغيرات A , B , C في المعادلة الآتية:

$$A = x^2 + 2y \dots (1)$$

$$B = 2x - 3A \dots (2)$$

$$C = A^2 + xB \dots (3)$$



الباب الثاني  
التطبيقات التجارية للغة C++  
C++ Language



## الفصل الأول

### أسس البرمجة بلغة C++

بدأ تاريخ لغة C++ منذ بدء تاريخ لغة C. فلغة C++ لم تنشأ من فراغ بل نشأة على أسس وقواعد لغة C. وكذلك الحال بالنسبة للغة C، فهي متأثرة بلغات أخرى سبقتها. لذا لا بد أولاً من معرفة نبذة تاريخية عن لغة C والأسباب التي أدت إلى انتشارها حول العالم.

ظهرت لغة C في عام 1972م على يد "دينيس ريتشي" وكانت متأثرة باللغات التي سبقتها وهي لغة B (1970م)، ولغة BCPL (1969م)، ولغة ALGOL-68 (1968م). في الأصل طورت لغة B وسميت NB ثم أعيد تسميتها إلى لغة C. وبالرغم أن لغة الـ C مصممة لكتابة برامج النظم (Systems Programming) إلا إنها مناسبة لجميع أنواع التطبيقات البرمجية الأخرى.

لغة الـ C مختلفة عن اللغات التي سبقتها من حيث التصميم. فاللغات السابقة كانت مصممة على أيدي لجان نظرية بعيدة عن الجانب العملي. أما لغة الـ C فهي مصممة ومنفذة على أيدي مبرمجين عمليين عملوا على إضافة كل ما يحتاجونه من مميزات في اللغة من واقع خبرتهم العملية. ونتيجة لذلك، تميزت وأصبحت من أشهر اللغات البرمجية المستخدمة في العالم على المستويين الصناعي و التعليمي، فمعظم البرامج المستخدمة بملايين الناس مكتوبة بلغة الـ C. وكذلك يرجع سبب انتشارها كونها جزءاً من نظام تشغيل UNIX المشهور، حيث أن مترجم الـ C (Compiler) ملحقاً مع الـ UNIX وهو مجاناً في الغالب؛ مما أتاح الفرصة للمبرمجين من استخدامه، وحتى أن معظم الـ UNIX مكتوباً بلغة الـ C.

ومن مميزات هذه اللغة التي جعلت المبرمجين يفضلونها عن اللغات الأخرى كونها لغة عامة عالية المستوى (High Level Language) مثل الـ Pascal و الـ ALGOL، تدعم البرمجة الهيكلية والتركييبية ولكنها تختلف عن تلك اللغات فيما توفره من إمكانيات وتسهيلات للمبرمج لا تتوفر إلا بلغة التجميع (Assembly Language). فهي تتيح للمبرمج إمكانية التحكم و السيطرة على مستوى البت الممثلة لجميع أنواع البيانات. هذه الميزة جعلتها اللغة المفضلة لكتابة برامج النظم المختلفة التي تتطلب السرعة والكفاءة والتي تكتب عادة بلغة

التجميع. فيتوفر في لغة C سهولة اللغات عالية المستوى، والسرعة والكفاءة والتحكم التي تتميز بها لغة التجميع؛ ولهذا السبب سمية لغة C باللغة الوسط (Middle Level Language). ويمكن تلخيص مميزات وخصائص لغة الـ C في النقاط التالية:

- القوة والفعالية: لغة C غنية بعملياتها المختلفة، وإجراءاتها المتوفرة في مكتباتها المرفقة مع المترجم، مما يمكن المبرمج من كتابة البرامج الكبيرة بطريقة متيسرة وموجزة، ويكون البرنامج سريع ورائع في الأداء.
- التوفر: مترجم C متوفر على جميع أنواع الأجهزة الكبيرة منها والصغيرة.
- الانتقالية: البرامج أو التطبيقات المكتوبة بلغة C على جهاز ما يمكن نقلها إلى جهاز أو نظام آخر بعد عمل تغييرات بسيطة جداً أو دون الحاجة إلى عمل أي تغيير.
- المرونة: لغة C لغة مرنة جداً فمثلاً يمكنك كتابة تعبير رياضي باستخدام متغيرات مختلفة من حيث النوع، كأن نكتب تعبيراً يجمع متغير من نوع الأعداد الصحيحة مع متغير من نوع الأعداد الحقيقية.
- التحكم: هذا ما يميز لغة C عن اللغات الأخرى عالية المستوى، فلغة C تمكن المبرمج من تناول و معالجة البتات و البايتات و عناوين المتغيرات، أي أن لغة C توفر للمبرمج تحكم و مرونة لغة التجميع و بساطة اللغات عالية المستوى.

وفي هذا الباب سوف نقدم لغة ++C القياسية حسب آخر إصدار لها من ANSI/ISO. وجميع البرامج الواردة في هذا الكتاب متوافقة مع مقاييس ANSI/ISO؛ لذا يمكن استخدامها مع أي مترجم متوافق مع هذه المقاييس الجديدة.

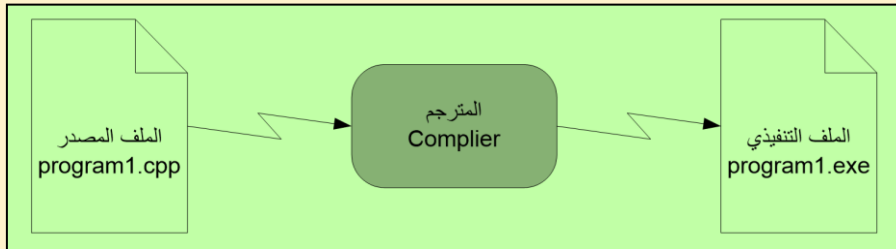
الأدوات التي تحتاجها لكتابة البرامج:

قبل البدء في كتابة البرامج بلغة ++C تحتاج إلى برنامجين أساسيين لبناء البرامج :

- برنامج محرر النصوص Text Editor.
- البرنامج المترجم للغة ++C القياسية C++ Compiler.

يساعد برنامج محرر النصوص في عمليات إدخال وتغيير المعلومات في جهاز الكمبيوتر كما يسهل عمليات إعداد النصوص. وأنت بحاجة إليه لكتابة الملف المحتوي على الكلمات والعناصر المكونة للتعليمات بلغة C++ (نص البرنامج) و الذي يسمى بالملف المصدر Source File. هذه التعليمات المكتوبة بلغة C++ قريبة من لغة الإنسان وغير مفهومة للحاسب إلا بعد تحويلها إلى لغة الآلة؛ لذا فأنت بحاجة إلى المترجم من أجل بناء البرنامج المفهوم للآلة.

فالمترجم هو عبارة عن برنامج خاص يقوم بترجمة البرامج المكتوبة من لغة C++ إلى لغة الآلة. وفي بيئة ويندوز، يقوم المترجم بعملية الترجمة ويحفظها في ملف يحمل نفس اسم الملف المصدر ولكنه ينتهي بالامتداد (.exe) وهو ما يسمى بالملف التنفيذي. فمثلاً عندما تكتب نص برنامج C++ وتحفظه في ملف اسمه program1.cpp ثم تقوم بعملية الترجمة، فإنك ستحصل على الملف التنفيذي program1.exe الذي تفهمه الآلة كما هو موضح في الشكل التالي.



وتمر عملية بناء البرنامج التنفيذي بثلاث مراحل على الأقل. يقوم بكل مرحلة منها برامج معين، كالتالي:

- المعالج المبدئي Preprocessing Stage: هذه المرحلة تسبق مرحلة الترجمة وفيها يقوم المعالج المبدئي بقراءة الملف المصدر و تحضيره للترجمة. حيث يقوم المعالج المبدئي بفحص الملف المصدر وتنفيذ الأوامر الموجهة له، وهي تلك الأوامر التي تبدأ بـ # وتسمى بتوجيهات المترجم. على سبيل المثال، قد يطلب من المعالج المبدئي بأن

يضم ملفات مكتبية خاصة، أو أن يستبدل نص بنص ... وغيره. بعبارة أخرى، يقوم فقط بمعالجات نصية.

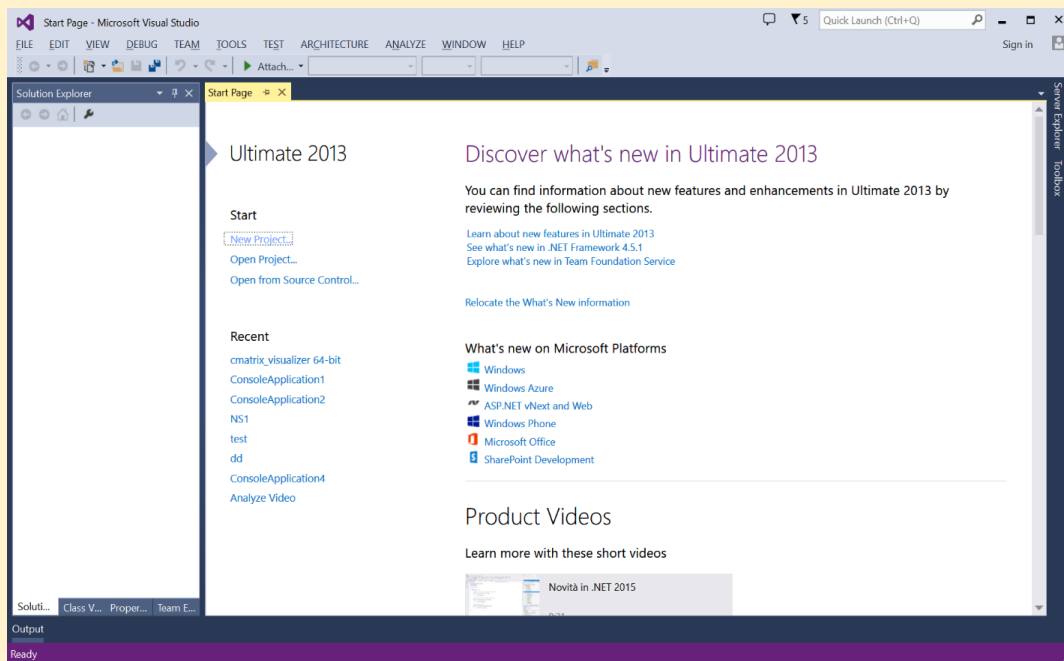
▪ مرحلة الترجمة **Compilation**: هنا يقوم المترجم بفحص قواعد تركيب و ترتيب الكلمات و العناصر المكونة للتعليمات بلغة ++C. وفي حالة عدم وجود أخطاء، يقوم بعملية ترجمة البرنامج من لغة ++C إلى لغة الآلة ويحفظها في ملف يحمل نفس اسم الملف المصدر ولكنه ينتهي بالامتداد (.obj).

▪ مرحلة الربط **Linking**: يقوم برنامج الربط "Linker" بربط برنامج لغة الآلة بعدة برامج أو إجراءات أخرى ويحفظها في ملف ينتهي بالامتداد (.exe) وهو البرنامج القابل للتنفيذ.

في الغالب، تأتي مترجمات ++C على شكل بيئة برمجية متكاملة **integrated environment** تتكون من محرر النصوص، المعالج المبدئي، المترجم، الرابط، و مكتبات **Libraries** تحتوي على أنواع مختلفة من الإجراءات المفيدة و الجاهزة للاستخدام.

و هناك العديد من المترجمات المتوفرة في الأسواق للغة ++C مثل: **Visual C++ .Net**، و **Borland C++** التي تعمل تحت أنظمة ويندوز المختلفة. لا يهم نوع المترجم المتوفر لديك طالما أنه يتفق مع ++C القياسية ولكن عليك معرفة كيفية استخدامه في عملية الترجمة (أنظر كتاب المستخدم المرفق مع مترجمك). كذلك يمكن استخدام مترجمات ++C لترجمة برامج C أيضا. فمعظم مترجمات ++C تترجم الملف المنتهي بالامتداد (.c) على أساس انه برنامج C، و تترجم الملفات المنتهية بالامتداد (.cpp) على أساس إنها برامج ++C. فإذا أردت استخدامه لترجمة ++C يجب عليك حفظ الملف المصدر بالامتداد (.cpp) أما إذا أردت استخدامه لترجمة برنامج بلغة C فعليك حفظه بالامتداد (.c).

يعتبر برنامج **Microsoft visual studio** هو واحد من أشهر البرامج التي يمكنك من كتابه وتنفيذ العديد من اللغات ومن ضمنها لغة الـ ++C ويوضح الشكلين التاليين شاشتي بدء التشغيل بعد تنزيل البرنامج على القرص الصلب الخاص بالكمبيوتر الشخصي لك.

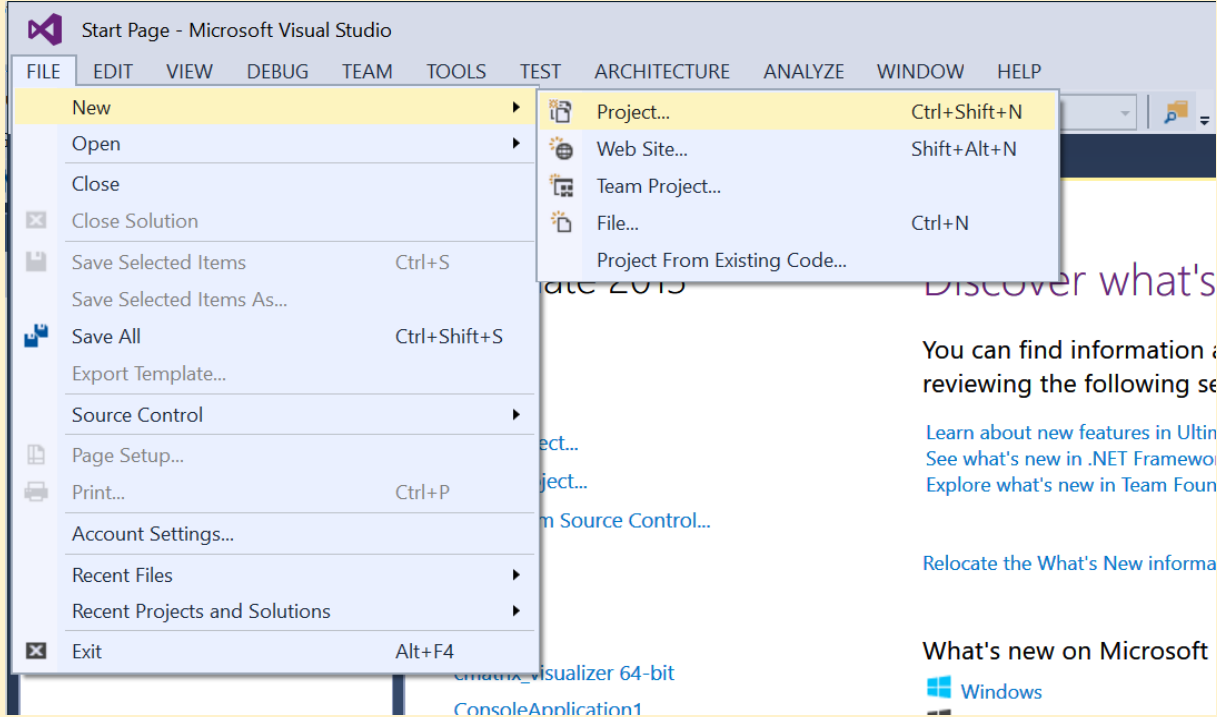


## كيفية كتابه وتنفيذ برنامج في Microsoft visual studio

بعد بدء تشغيل البرنامج تتبع الخطوات الموضحة في الشكل التالي لإنشاء مشروع برنامج

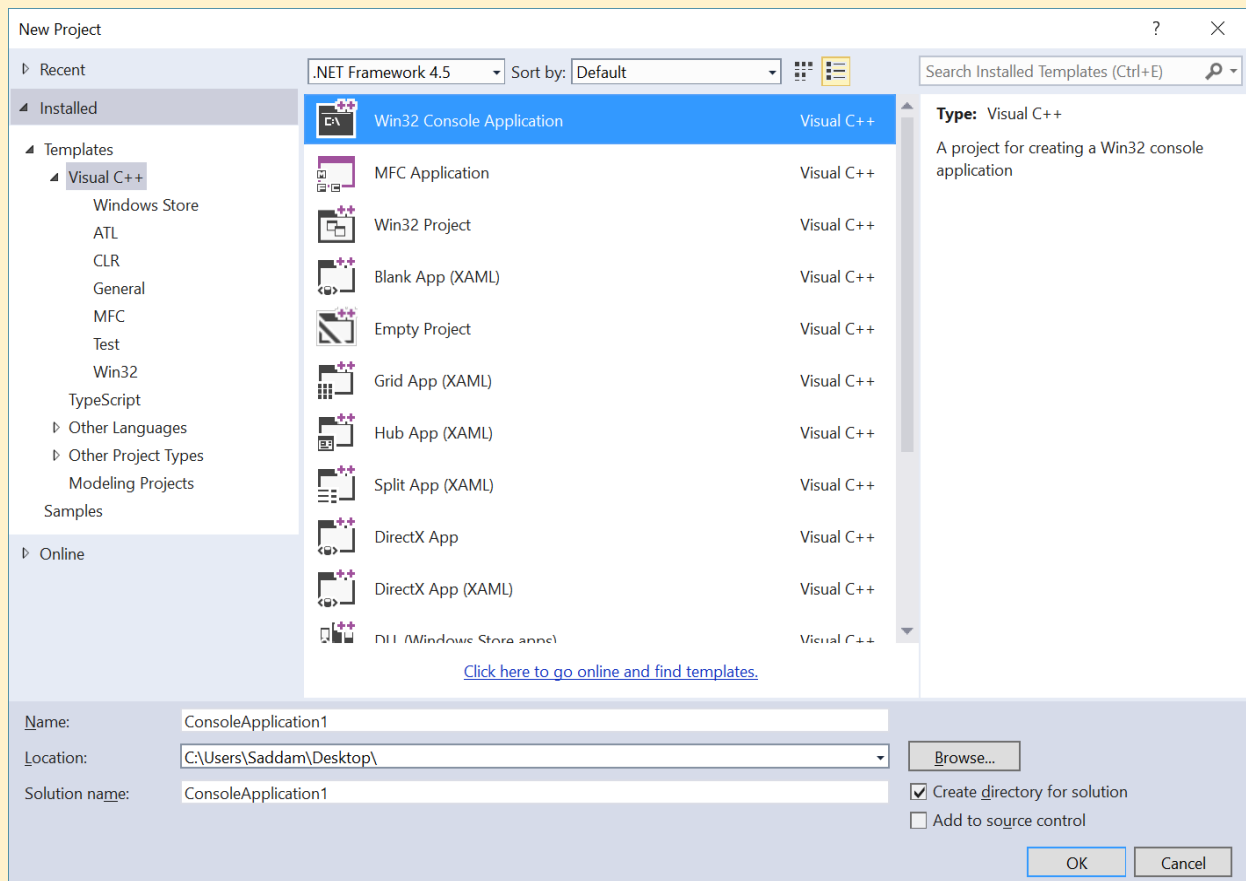
جديد:

### File→New→Project

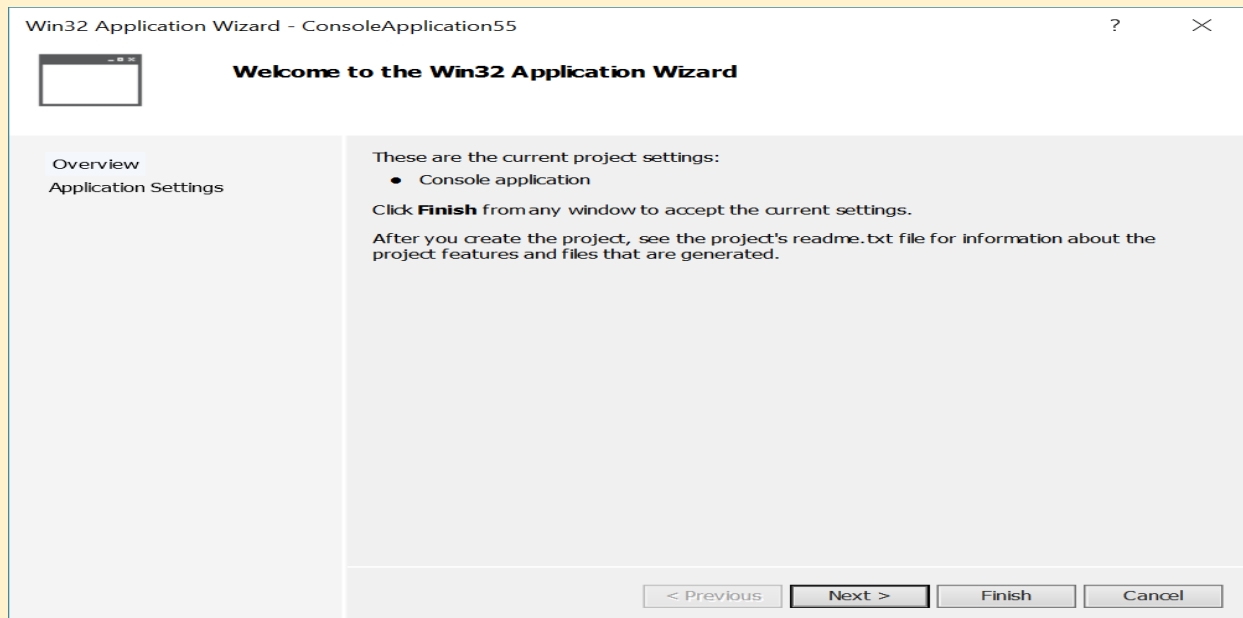


ثم يقوم البرنامج بعرض الشاشة التاليه لإختيار إسم البرنامج و مكان تخزينه على القرص الصلب . يجب عليك أن تختار (موضح في الشاشة التاليه) Visual C++ من القائمه التي في يسار الشاشة وتختار Win32 Console Application من الخيارات التي ستظهر لك. ثم إضغط ok بعد إختيار إسم ومكان حفظ البرنامج على القرص الصلب.

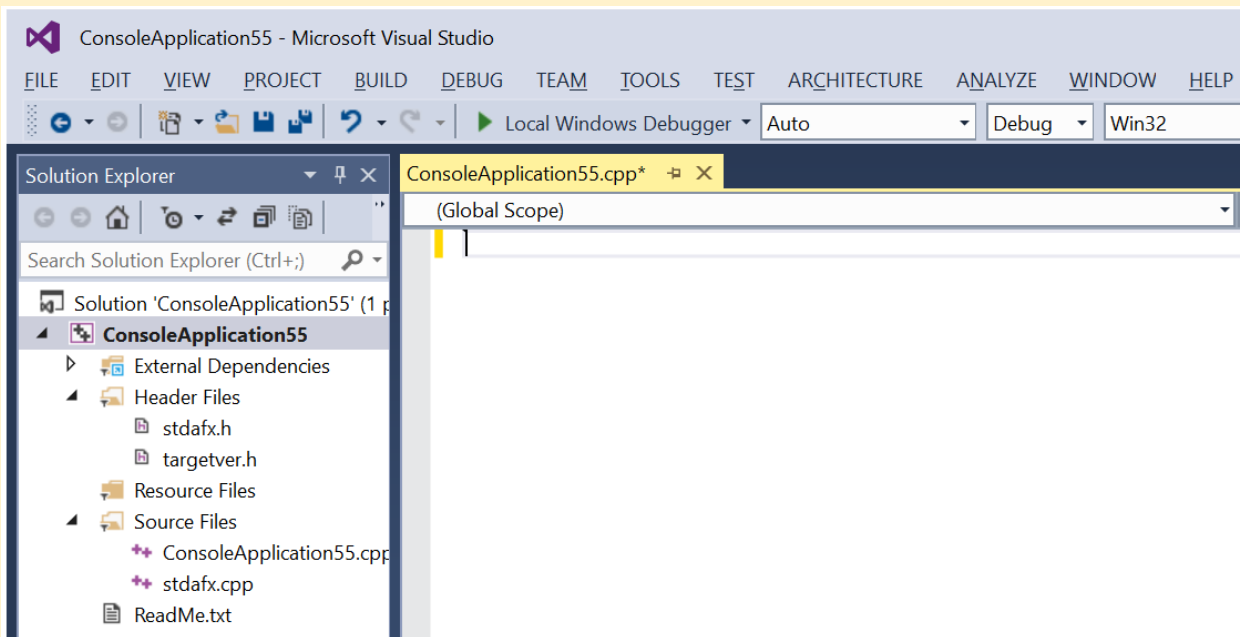




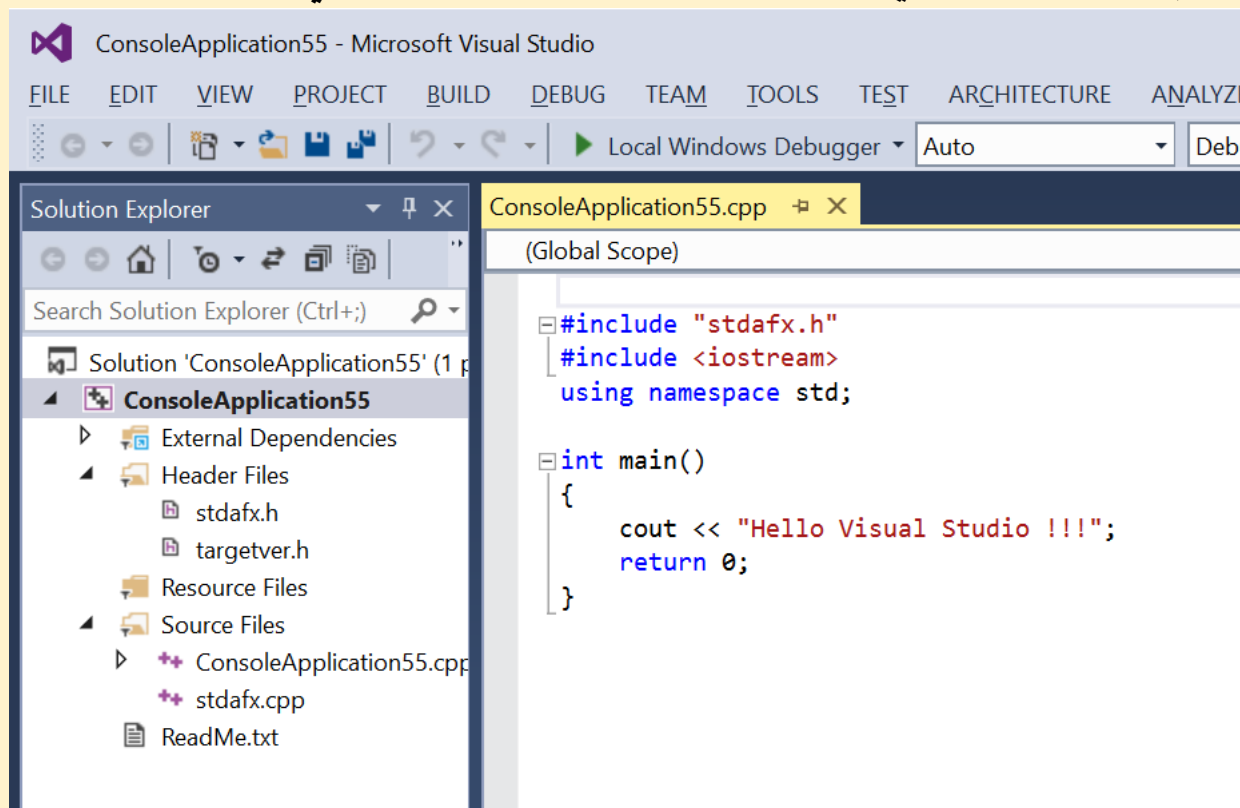
ثم ستظهر لك الشاشة التاليه فأضغط زر **finish**



بعد هذه الخطوه ستظهر لك شاشة المحرر التاليه:



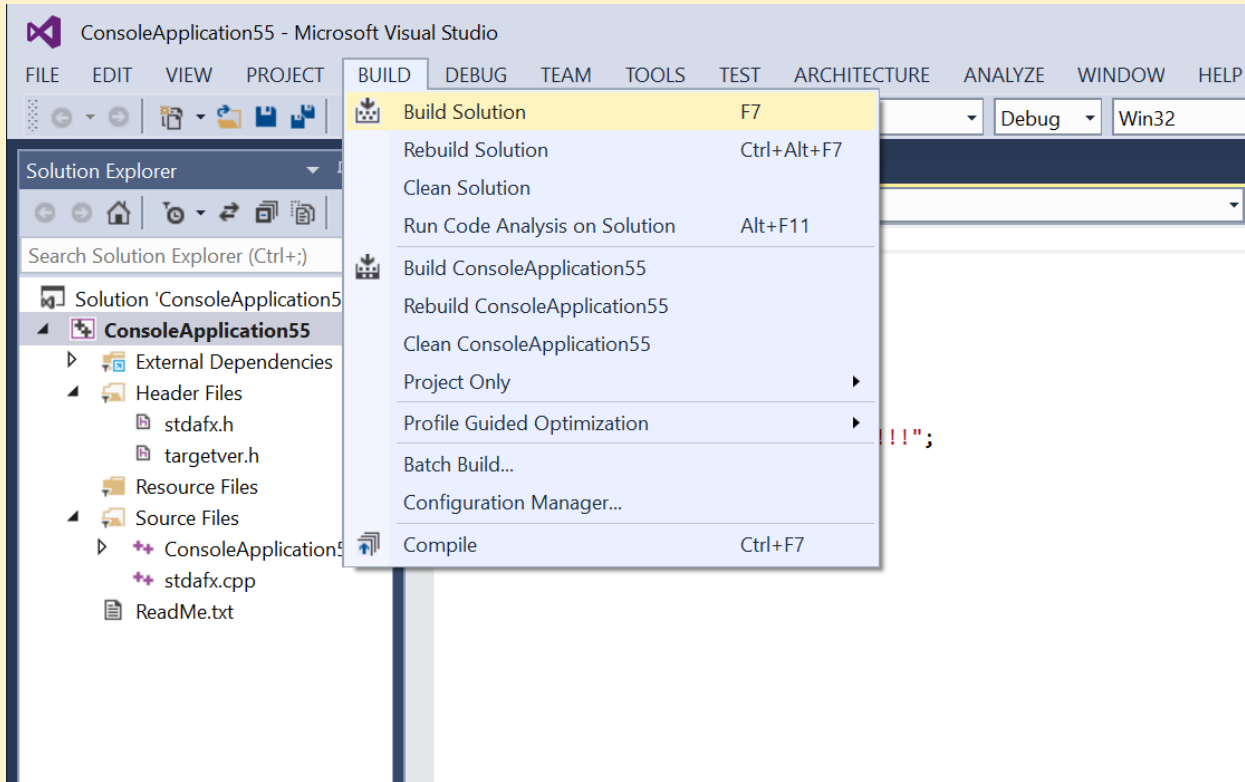
قم بكتابه البرنامج الذي ترغب فيه , وليكن كمثال , البرنامج التالي كما هو موضح:



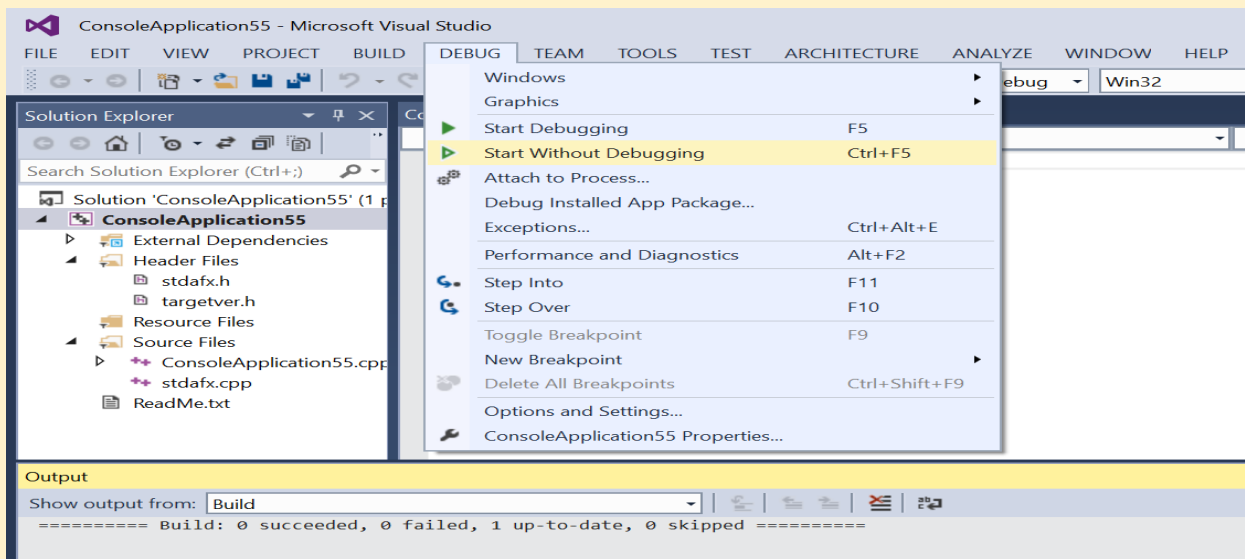
ملحوظه :

ملف التوجيه "stdafx.h" هو متطلب لبيئه الـ C++ لتسريع عمليه تنفيذ البرنامج باستخدام ملفات التوجيه مسبقه الترجمه pre-compiled headers.

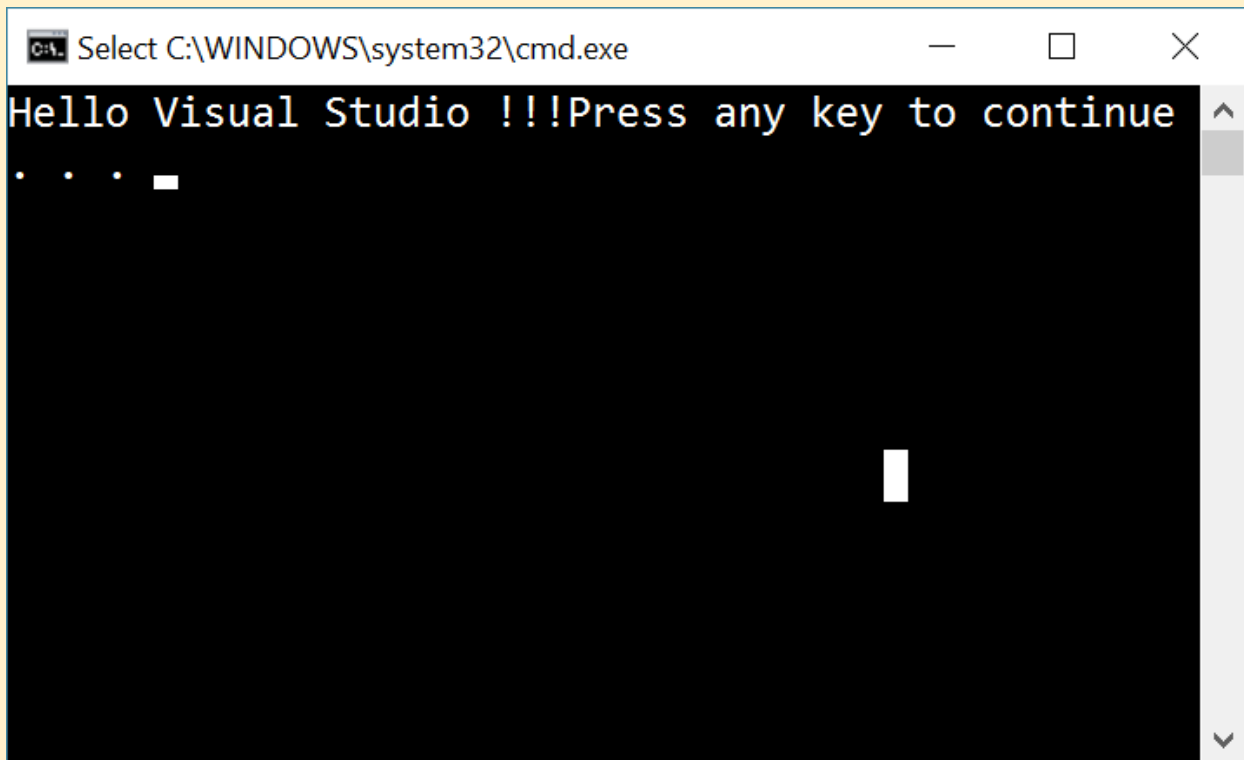
ثم بعد ذلك لتنفيذ البرنامج اضغط قائمة **Build** → **Build Solution** كما هو موضح في الشاشة التاليه .



ولعمل الخطوه النهائيه لتنفيذ البرنامج وعرض ومخرجاته اضغط قائمة **Debug** → **Build Solution** كما هو موضح في الشاشة التاليه :

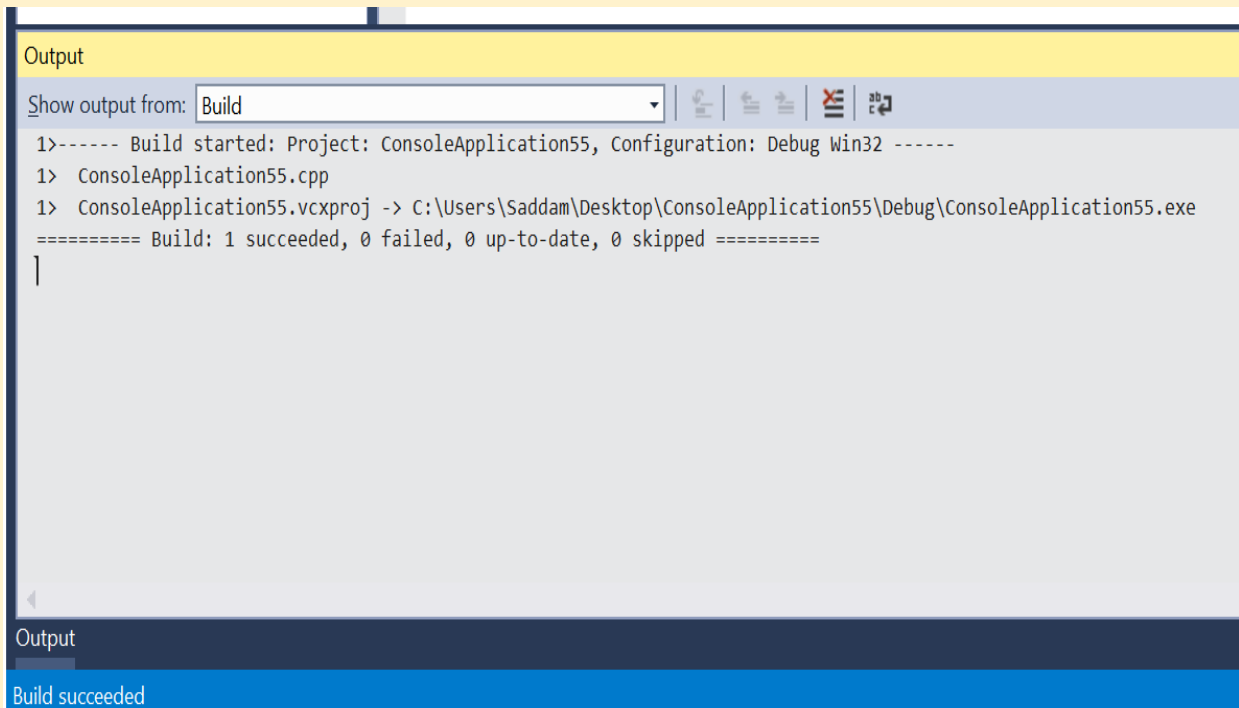


وتكون المخرجات كما هو معروض في الشاشة التاليه :



```
Select C:\WINDOWS\system32\cmd.exe
Hello Visual Studio !!!Press any key to continue
. . . █
█
```

والمحرر Visual Studio ذكي فهو يقوم بإعطاءك تقرير عن الأخطاء الموجوده في البرنامج كما هو موضح في الشاشة التاليه.



```
Output
Show output from: Build
1>----- Build started: Project: ConsoleApplication55, Configuration: Debug Win32 -----
1> ConsoleApplication55.cpp
1> ConsoleApplication55.vcxproj -> C:\Users\Saddam\Desktop\ConsoleApplication55\Debug\ConsoleApplication55.exe
===== Build: 1 succeeded, 0 failed, 0 up-to-date, 0 skipped =====
|
```

Output  
Build succeeded

سنقوم الآن بدراسة لغة C++ من حيث جملها المختلفة ومراعاة أسسها وقواعدها المختلفة من خلال مجموعة من التدريبات والأمثلة العملية حتي نتدرج في تعلمها بشكل سليم.

**\*\* الشكل العام للبرنامج**

### Simple program to print any statement

البرنامج الآتي يبين الشكل العام لبرنامج يقوم بطباعة جملة Can I help you? ولا يهمنا هنا فهم كل جزئية في البرنامج فهذا سوف يتم توضيحه في الصفحات التالية:

```
//how to print a message on the screen
```

```
#include<iostream.h>
```

```
Main ( )
```

```
{
```

```
    cout<<"Can I help you?\n";
```

```
Return O;
```

```
}
```

وتكون المخرجات على الشاشة

Can I help you?

وحتى نتعرف على عبارات ودوال ومكونات هذه اللغة نفسر البرنامج السابق سطر سطر.

السطر الأول:

```
//how to print a message on the screen
```

يبدأ هذا السطر من البرنامج بالشرطة المزدوجة // الدالة على أن بقية السطر عبارة عن تعليق **comment**، وتضاف التعليقات الى البرامج لتساعد المبرمج أو أي شخص آخر أو قارئ البرنامج على فهم ما يفعله البرنامج، لذا يفضل أن يبدأ كل برنامج بلغة C++ بتعليق يوضح الغرض من كتابته.

وجمل التعليق قد تمتد لسطر واحد فقط **single-line comment** وتستخدم في أولها

الشرطة المزدوجة //، وقد تمتد الى عدة أسطر **multi-line comments** ويأخذ الصورة التالية:

**/\*Program 1:**

**how to print a message on the screen**

**\*/**

ونلاحظ أن جملة التعليق تبدأ بالرمز **/\*** وتنتهي بالرمز **\*/**.

السطر الثاني:

**#include<iostream.h>**

كلمة **include** عبارة عن توجيه للمترجم بأن يضم محتويات ملف الرأس **iostream** إلى الملف الحالي (الملف المحتوي على عبارة **#include**) بحث يصبح جزءاً منه. ويوجد ملف الرأس **iostream**، وهو اختصار **input/output stream**، في مكتبة **C++** القياسية، ويحتوي على التعاريف والإعلانات الضرورية والمفيدة لعمليات الإدخال والإخراج القياسية. حيث تتعامل هذه المكتبة مع عمليات الإدخال والإخراج على أنها تيارات (**streams**) من الحروف. ونحن بحاجة إلى ضم ملف الرأس **iostream** إلى البرنامج السابق بسبب استخدامنا لعبارة الإخراج **cout** في السطر 6. فالبرنامج يطبع العبارة "Hello World" على الشاشة باستخدام الكائن **cout** الموجود في الملف **iostream**. بصفة عامة، يجب إضافة ملف الرأس **iostream** في أي برنامج يستخدم أدوات الإدخال أو الإخراج القياسية مثل **cin** و **cout**. والحرف **#** الذي يسبق **include** يستخدم للدلالة على أن السطر الحالي هو توجيهها للمترجم. أما قوسا الزاوية **< >** اللذان يحيطان بالملف **iostream** فيدلان على وجود ملف الرأس هذا في مكتبة **C++** القياسية. وكما ذكرنا سابقاً، يقوم المعالج المبدئي بتنفيذ مثل هذه التوجيهات والتي تبدأ بـ **#**.

السطر الثالث الدالة الرئيسية

**main()**

يبدأ تشغيل أي برنامج **C++** من دالة تسمى **main()**، وتعتبر جزءاً أساسياً في البرنامج،

وتشير الأقواس بعدها لكونها دالة، وكما سنرى قد توجد أكثر من دالة بالبرنامج من

الضروري أن أحدهم **main**. ويتم حصر جسم الدالة بالأقواس **{ }**.

قوس بداية الدالة الرئيسية { السطر الرابع

## السطر الخامس

```
cout<<"Can I help you?\n";
```

تظهر جملة أو عبارة **cout statement** النص المحصور بين علامتي الاقتباس " " على الشاشة ويطلق عليه النص في هذه الحالة ثابت سلسلي، ويجب أن تنتهي كل عبارة في البرنامج بفصلة منقوطة ؛ (semi colon)، والجدير بالذكر يلفظ الاسم **cout** بـ **C out** وهو كائن مقترن مع الشاشة، والعامل << يسمى بعامل الوضع **Put to operator** يرسل الأشياء الموجودة على يمينه الى أي شيء على يساره.

ولتوضيح أكثر نضرب بعض الأمثلة:

```
#include <iostream.h>
main ( )

{
    cout << 7 << " is an integer.\n";
    cout << 'a' << "is a character.\n";
}
```

وتكون مخرجات تنفيذ البرنامج:

```
7 is an integer.
a is a character
```

ويتضح من المخرجات ما يلي:

- 1- يتم حصر النص المطلوب ظهوره على الشاشة بين علامتي اقتباس " is an integer".
  - 2- تتم كتابة الثوابت الرقمية بدون علامتي اقتباس 7 <<.
  - 3- يتم حصر حرف واحد مطلوب ظهوره على الشاشة بعلامة اقتباس فردية 'a' <<.
- تقوم بعض اللغات كـ **Basic** مثلاً بالانتقال إلى سطر جديد تلقائياً في نهاية كل عبارة خرج ، لكن **C++** لا تفعل ذلك كما أن العبارات المختلفة والموضوعة في أسطر مختلفة لا تؤدي إلى ذلك .
- لا ينشئ الكائن **cout** أسطراً جديدة تلقائياً، والمخرجات في البرنامج التالي توضح ذلك:-

// This program displays output on the screen

```
#include<iostream.h>
main ( )
{
    cout<<10;
    cout<<20<<30;
    return 0;
}
```

وتكون المخرجات

102030

نلاحظ في المخرجات أن الأرقام ملتصقة ببعضها، لذلك نجد وجود طرق عديدة في C++ للتحكم في شكل المخرجات يطلق عليها تتابعات الهروب **Escape Sequences**.

حيث يلتصق كل الخرج ببعضه البعض ، لذا من الجيد أن يكون لدينا طرق في C++ للتحكم بطريقة تنسيق الخرج والتي منها تتابعات الهروب (Escape Sequences).

نلاحظ أنه لم تتم طباعة \n على الشاشة ، \ تسمى الشرطة الخلفية (Back slash) أو حرف هروب (Escape character) وتسمى هي والحرف الذي يليها تتابع هروب. تتابع الهروب \n يعنى الانتقال إلى سطر جديد حيث يجبر المؤشر على الانتقال إلى بداية السطر التالي ، الآن إليك بعض تتابعات الهروب الشائعة:-

<u>الوصف</u>	<u>تتابع الهروب</u>
سطر جديد.	\n
مسافة أفقية.	\t
حرف التراجع back space.	\b
طباعة شرطة خلفية \\.	\\
حرف الإرجاع، يجبر المؤشر على الانتقال إلى بداية هذا السطر.	\r
طباعة علامة اقتباس	\"



**Return 0;**

تكتب العبارة **return 0;** في نهاية الدالة **main ( )** وتشير القيمة **0** الى أن البرنامج انتهى نهاية صحيحة وهو ما سنتعرف عليه لاحقا أكثر عند دراسة الدوال.

السطر السابع

قوس نهاية الدالة الرئيسية }

مثال آخر:

إليك الآن مثلاً لبرنامج يستقبل رقمين من المستخدم ويجمعهما ويعرض ناتج الجمع:-

```
#include<iostream.h>
#include<conio.h>
main ( ) {
    int integer1, integer2, sum;
    cout <<"Enter first integer\n";
    cin >> integer1;
    cout <<"Enter second integer\n";
    cin >> integer2;
    sum= integer1+integer2;
    cout <<"sum="<<sum<<endl;
    getch();
    return 0;
}
```

وتكون المخرجات:

```
Enter first integer
7
Enter second integer
3
sum= 10
```

أشرح بنفسك خطوات البرنامج السابق؟

## \*\* أنواع البيانات الأساسية في لغة C++

هنالك سبعة أنواع بيانات أساسية في C++ ، واحد منها يمثل الأحرف وثلاثة تمثل أرقاماً كاملة (أعداد صحيحة) وثلاثة تمثل أرقاماً حقيقية. الجدول الآتي يلخص هذه الأنواع.

اسم النوع	يستعمل لتخزين	أمثلة عن القيم المخزنة
char	أحرف	'a'
short	أرقام صحيحة قصيرة	222
int	أرقام صحيحة عادية الحجم	153,406
long	أرقام صحيحة طويلة	123,456,789
float	أرقام حقيقية قصيرة	3,7
double	أرقام حقيقية مزدوجة	7,533,039,395
long double	أرقام حقيقية ضخمة	9,176,321,236,01202,6

### 1/ الأحرف char :-

يتم تخزين الأحرف في متغيرات من النوع char العبارة:-

**char ch;**

تنشئ مساحة من الذاكرة لحرف وتسميه ch. لتخزين حرف ما في هذا المتغير نكتب

**ch='z'**

ودائماً تكون الأحرف الثابتة كـ 'a' و 'b' محصورة بعلامة اقتباس فردية.

يمكن استعمال المتغيرات من النوع char لتخزين أرقام كاملة بدلاً من أحرف ، فمثلاً يمكننا

كتابة:-

**ch=2;**

لكن نطاق القيم الرقمية التي يمكن تخزينها في النوع char يتراوح بين

-128 إلى 127 لذا فإن هذه الطريقة تعمل مع الأرقام الصغيرة فقط.

## 2/ الأعداد الصحيحة:

تمثل الأعداد الصحيحة أرقاماً كاملة أي قيم يمكن تعدادها ، كعدد أشخاص أو أيام أو عدد صفحات مثلاً ، ولا يمكن أن تكون الأعداد الصحيحة أرقاماً ذات نقطة عشرية ولكنها يمكن أن تكون سالبة.

هنالك ثلاثة أنواع من الأعداد الصحيحة في **C++:short** قصير، **int** عدد صحيح، **long** طويل وهي تحتل مساحات مختلفة في الذاكرة. الجدول التالي يبين هذه الأنواع والمساحة التي تأخذها في الذاكرة ونطاق الأرقام التي يمكن أن تأخذها:

اسم النوع	الحجم	النطاق
char	1byte	-128 إلى 127
short	2byte	-32,768 إلى 32,767
int	مثل short في أنظمة 16bit ومثل long في أنظمة 32bit	
long	4byte	-2,147,483,648 إلى 2,147,483,647

## 3/ الأعداد الصحيحة غير المعلمة (Unsigned):-

كل الأعداد الصحيحة لها إصدارات غير معلمة (unsigned) . لا تستطيع المتغيرات التي ليس لها علامة تخزين قيم سالبة، ونجد أن نطاق قيمها الموجبة يساوي ضعف مثيلاتها التي لها علامة، الجدول التالي يبين هذا:-

اسم النوع	الحجم	النطاق
unsigned char	1byte	0 إلى 255
unsigned short	2byte	0 إلى 65,535
unsigned int	مثل unsigned short في أنظمة 16bit ومثل unsigned long في أنظمة 32bit	
unsigned long	4byte	0 إلى 4,294,967,295

## 4/ الأرقام العائمة Float

يتم استعمال الأرقام العائمة لتمثيل قيم يمكن قياسها كالأطوال أو الأوزان. ويتم تمثيل الأرقام العائمة عادة برقم كامل على اليسار مع نقطة عشرية وكسر على اليمين. هنالك ثلاثة أنواع من الأرقام العائمة في أنظمة التشغيل الشائعة الاستعمال. وأشهر نوع أرقام عائمة هو النوع **double** والذي يتم استعماله لمعظم دالات **C++** الرياضية. يتطلب النوع **float** ذاكرة أقل من النوع **double**. الجدول التالي يوضح هذه الأنواع والحجم الذي تأخذه في الذاكرة.

اسم النوع	الحجم
float	4byte
double	8byte
long double	10byte

### \*\* المتغيرات:

عند كتابة البرنامج، نحتاج لتخزين المعلومات الواردة في ذاكرة الحاسب تحت عناوين يطلق عليها أسماء المتغيرات، والتي تختلف حسب أنواع المعلومات المخزنة، لذا يجب أن نعلم المترجم في بداية البرنامج عن أنواع المتغيرات المراد استخدامها، ويمكن تعريف المتغيرات في أي مكان في البرنامج ولكن يجب تعريفها قبل استعمالها، كما يمكن تعريف المتغيرات التي تنتمي الى نفس النوع في سطر واحد.

يتم تعريف المتغير بذكر الاسم ونوع البيانات التي يمكن ان يحملها هذا المتغير من أي سلسلة تحتوي على أحرف **Letters** أو أرقام **Digits** أو خطأً تحتياً **Under** **score(\_)** بشرط أن لا يبدأ برقم، وبأي حجم كبيراً أو صغيراً فمثلاً الأسماء **integer1** و **Integer1** تعامل كمتغيرات مختلفة.

الادخال من لوحة المفاتيح:

```
cin>>integer1
```

هذه العبارة تخزن الرقم الذي يكتبه المستخدم من لوحة المفاتيح في متغير يدعى `integer1`. ويمثل الكائن `cin` والذي يتلفظ `in` لوحة المفاتيح، ويأخذ عامل الحصول `>>` `get from` الأشياء على يساره ويضعها في المتغير الموجود على يمينه، عند تنفيذ هذه العبارة ينتظر البرنامج أن يكتب المستخدم رقماً من النوع `integer` ونضغط على مفتاح `enter`، فيتم ادخال القيمة المدخلة الى المتغير `integer1`.

ويمكن استعمال عامل الحصول عدة مرات في نفس العبارة:

```
cin>>integer1>>integer2
```

والضغط على مفتاح `enter`، أو مفتاح المسافة `space`، أو مفتاح `tab` بعد كل قيمة، قبل أن يكتب القيمة التالية، ولكنه من الأفضل عادة إدخال قيمة واحدة في كل مرة لتجنب الخطأ.

المناور `endl`:

العبارة:

```
cout<<'sum= '<<sum<<endl
```

تطبع النص `sum=` متبوعاً بقيمة `sum`، نلاحظ أننا استخدمنا `endl` وهو

وسيلة أخرى في `C++` للانتقال إلى سطر جديد، ويسمى مناور `manipulator` و `endl`

اختصاراً لـ `end line`، وهو يعمل تماماً كما يعمل تتابع الهروب `\n`.

## \*\* العوامل الحسابية (math operator)

ويوضح الجدول التالي العوامل الحسابية المستخدمة في لغة C++

العامل	الوظيفة	التعبير الجبري	التعبير في C++
+	جمع	$B+h$	$B+h$
-	طرح	$B-h$	$B-h$
*	ضرب	$Bh$	$B*h$
/	قسمة	$B/h,$	$B/h$
%	الباقي	$B \bmod h$	$B\%h$

العوامل الأربعة الأولى تنجز أعمالاً مألوفة لدينا، أما عامل الباقي % المسمى أيضاً المعامل modulus، يتم استعماله لحساب باقي القسمة لعدد صحيح على عدد آخر، لذلك فالتعبير  $20\%3$  يساوي 2 . تسمى هذه العوامل الحسابية بالعوامل الثنائية لأنها تعمل على قيمتين.

## \*\* العوامل العلائقية أو المنطقية (Relational Operators)

تقارن العوامل العلائقية قيمتين، وتؤدي إلى نتيجة صحيح/خطأ وفقاً لما إذا كانت المقارنة صحيح/خطأ. هنالك ستة عوامل علائقية مبينة في الجدول أدناه:

الرمز	المعنى	مثال
==	يساوى	$a==b$
!=	لا يساوى	$a!=b$
>	أكبر من	$a>b$
<	أصغر من	$a<b$
>=	أكبر من أو يساوى	$a>=b$
<=	أصغر من أو يساوى	$a<=b$

تكون التعابير المبينة في عمود المثال صحيحة أو خطأ وفقاً لقيم المتغيرين  $a$  و  $b$ .

فلنفرض مثلاً أن:

□  $a$  يساوى 9

□  $b$  يساوى 10.

التعبير  $a==b$  خطأ.

التعبير  $a!=b$  صحيح وكذلك التعبيرين  $a<b$  و  $a<=b$  ،

والتعبيرين  $a>b$  و  $a>=b$  خطأ..

## \*\* تعليمات الإسناد:

يستخدم الرمز = للدلالة على الإسناد، والشكل العام لعبارة الإسناد البسيطة هو:

**Result = expression ;**

وكمثال لحساب المتوسط

$$\text{Average}=(a+b)/2$$

فيتم إسناد ناتج العملية في الطرف الأيسر للمتغير في الطرف الأيمن

يمكن في الصيغ الحسابية الموجودة إلى يمين الرمز = أن نجد الرموز التالية:

+ للجمع

- للطرح

\* للضرب

/ للقسمة

% باقي القسمة (modulus)

مثال

**i=3;**

**sum=0.0;**

**perimeter=2.0\*(length+breadth);**

**ratio=(a+b)/(c+);**

إن نمط المعاملات المستخدمة داخل الصيغة هام جداً لتحديد النتيجة، وهنا تراعى القواعد التالية:

◀ إذا كان المعاملان من النمط **int** تكون النتيجة من النمط **int** كذلك.

◀ إذا كان أحد المعاملين أو كلاهما من النمط **float** تكون النتيجة من النمط **float** كذلك.

◀ إذا كانت نتيجة الصيغة من النمط **int** وكان المتحول إلى يسار معامل الإسناد من النمط **float**، عندها يجري تحويل القيمة الناتجة من **int** إلى **float** قبل تخزينها في المتحول إلى يسار معامل الإسناد.

◀ إذا كانت نتيجة الصيغة من النمط **float** وكان المتحول إلى يسار معامل الإسناد من النمط **int**، عندها يجري تحويل القيمة الناتجة من **float** إلى **int** قبل تخزينها في المتحول إلى يسار معامل الإسناد، وهذا يعني ضياع الجزء العشري الموجود في القيمة الأصلية. ويجري هذا التحويل بقطع الجزء العشري دون إجراء أي عملية تنوير.

حسب القاعدة الأخيرة، فإن الإسناد قد يؤدي إلى فقد الدقة ما لم تراعى الأنماط على نحو دقيق.



فمثلاً ما هو محتوى i بعد تنفيذ التعليمات التالية:

```
int i;
```

```
i = 3.5 ;
```

تكون الأجابة 3

تواجهنا المشكلة ذاتها عند استعمال معامل القسمة، حيث تقول القاعدة:

◀ ناتج قسمة عددين من النمط int يكون من النمط int كذلك. ويجري القطع عندما يكون حاصل القسمة عدداً عشرياً موجباً. أما في حالة الناتج السالب، فلا تحدد اللغة ما الذي يحدث. ومن هنا يجب عليك الانتباه لئلا تفقد الدقة!

فمثلاً

```
i = 1/7;
```

تؤدي الى وضع القيمة في i

كما يلاحظ أن أولويات المعاملات هي

\* / % + =

مثال: إذا طلب منك كتابة برنامج يقرأ عدداً صحيحاً يدخله المستخدم، ويمثل درجة الحرارة

بالفهرنهايت، ثم يقوم بتحويلها الى قيمتها بالسنتيجرام. يكون البرنامج:

```
//convert Fahrenheit to centigrade
//enter a Fahrenheit value from the user
// converts it to centigrade and outputs
// the result
#include<iostream.h>
Void main()
{
    const float mult=5.0/9.0; //5/9 returns zero
                                //integer division
    const int sub=32;
```

```

float fahr,cent;
cout<<"enter Fahrenheit temperature:";
cin>>fahr;
cent=(fahr-sub)*mult;
cout<<"centigrade equivalent of"<<fahr<<"is"<<cent<<endl;
}

```

معاملات الزيادة increment والانقاص decrement  
يكثر في جميع البرامج ورود شكلين من التعليمات هما:

$n=n+1;$

$n=n-1;$

كما يوجد معاملات وهما ++للزيادة و - للانقاص، بحيث يمكننا كتابة

$n++$  بدلاً من  $n=n+1$

و  $n--$  بدلاً من  $n=n-1$

كما توجد تعليمات تنفذ التعليمة الحسابية المطلوبة والإسناد معاً، فمثلاً في حالة

الجمع وبدلاً من كتابة  $sum = sum + x;$  نكتب  $Sum += x;$

ويبين المثال التالي استعمال التعليمات الحسابية المختصرة +, \*, /, %:

$M += k;$             or             $M=M+k;$

$D *= 50;$             or             $D=D*50;$

$C/=y+1;$             or             $C=C/(y+1);$

$H %= 5;$             or             $H = H\% 5;$

تعليمات الصيغ المنطقية:

تعطى الصيغة المنطقية نتيجة صح أو خطأ وتستعمل المعاملات < وتعنى أصغر من و

> وتعنى أكبر من و <= وتعنى أصغر من أو يساوى و >= وتعنى أكبر من أو يساوى و

== يساوى و != وتعنى لا يساوى.

## تدريب

\*\* حدد ما إذا كانت العبارات الآتية صحيحة أم خطأ:

\* التعليقات تجبر الحاسب على طباعة النص الذي يلي // على الشاشة عند تنفيذ البرنامج.

\* تتابع الهروب /n يجبر المؤشر على الانتقال الى سطر جديد.

\* برنامج السي بلس بلس والذي يقوم بطباعة ثلاث أسطر على الشاشة يجب أن يحتوي على ثلاث عبارات تستعمل cout  
\*\* ما هي مخرجات العبارة التالية:

```
cout<<"/n ** /n *** /n **** /n";
```



## الفصل الثاني

### الجمل الشرطية

عادة يتم تنفيذ العبارات حسب تسلسل ورودها في البرنامج ويسمى هذا بالتنفيذ التتابعي (Sequential Execution). لكننا سنتعرض لبعض عبارات C++ والتي تجعل التنفيذ ينتقل لعبارة أخرى قد لا تكون التالية في تسلسل البرنامج، ويسمى هذا بنقل التحكم Transfer of control. تنقسم بنيات التحكم في C++ إلى قسمين: بنيات التحكم الشرطية وسنفرده هذه الوحدة لتوضيحها. والنوع الثاني وهو بنيات التحكم التكرارية والتي سنفرده الوحدة التالية للحديث عنها.

#### \*\* جملة if

تأخذ هذه الجملة الشرطية الشكل التالي:

**If(condition)**

**Statement;**

فإذا تحقق الشرط condition بعد if يتم تنفيذ الجملة statement وإذا لم يتحقق

لا يتم تنفيذها.

حيث يكون الشرط condition إما متحولاً منطقياً أو صيغة منطقية (راجع الفقرة السابقة).

أما statement فهو إما تعليمة وحيدة تنتهي بفاصلة منقوطة، أو مجموعة من التعليمات. وفي حالة مجموعة من التعليمات نضع هذه التعليمات بين قوسين كبيرين.

ويمكن توضيح مثال كما يلي:

```

#include <iostream.h>
main ( )
{
int num1 , num2;
cout << " Enter two integers, and I will tell you\n"
<< " the relation ships they satisfy: ";
cin >> num1>> num2;
if (num1== num2)
cout << num1 << " is equal to " << num2 << endl;
if (num1!= num2)
cout << num1 << " is not equal to " << num2 << endl;
if (num1< num2)
cout << num1 << " is less than " << num2 << endl;
if (num1> num2)
cout << num1 << " is greater than " << num2 << endl;

if (num1<= num2)
cout << num1 << " is less than or equal to " << num2
<< endl;
if (num1>= num2)
cout << num1 << " is greater than or equal to " << num2
<< endl;
return 0;
}

```

فإذا كانت:  $num1=3$  ,  $num2=7$

تكون المخرجات:

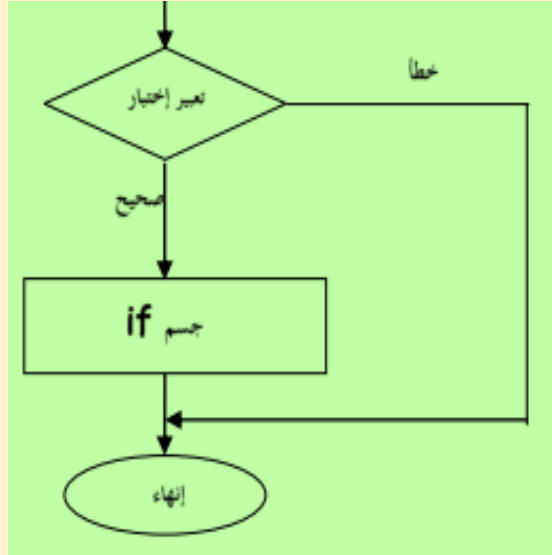
```

Enter two integers , and I will tell you
The relation ships they satisfy: 3 7
3 is not equal to 7
3 is less than 7
3 is less than or equal to 7

```

تتألف العبارة `if` من الكلمة الأساسية `if`، يليها تعبير اختبار بين قوسين، ويتألف جسم القرار الذي يلي ذلك إما من عبارة واحدة ، أو من عدة عبارات تحيطها أقواس حاصرة { }

ويمكن توضيح عمل الجملة `if` كما يلي:



**\*\* جملة if..... else**

في العبارة if البسيطة يحدث شيء إذا كان الشرط صحيحاً، لكن إذا لم يكن كذلك لا يحدث شيء على الإطلاق. لكن لنفترض أننا نريد حدوث شيء في الحالتين إذا كان الشرط صحيحاً وآخر إذا لم يكن كذلك، لتحقيق ذلك نستخدم العبارة if... else

وتأخذ هذه الجملة الشكل العام التالي:

```

if (condition)
    statement T;
else
    statement F;
  
```

حيث يكون الشرط condition إما متحولاً منطقياً أو صيغة منطقية. أما statement T و statement F فتتدل كل منهما إما على تعليمة وحيدة تنتهي بفاصلة منقوطة، أو مجموعة من التعليمات. وفي حالة مجموعة من التعليمات نضع هذه التعليمات بين قوسين كبيرين.

مثال

```
if (disc >= 0.0)
    cout<<"roots ara real";
if (disc > 0.0)
    cout<<"roots ara complex";
```

مثال

```
#include <iostream.h>
main ( )
{
    int grade ;
    cout << " Enter the grade";
    cin >>grade;
    if(grade>= 50)
        cout<<"pass" <<endl;
    else
        cout <<"fail"<<endl;
    return 0;
}
```

وبفرض أن grade=90

تكون المخرجات:

```
Enter the grade 90
Pass
```

هناك طريقة أخرى للتعبير عن المثال السابق باستخدام العامل المشروط

```
cout<<(grade>=50?"pass":"fail")<<endl;
```



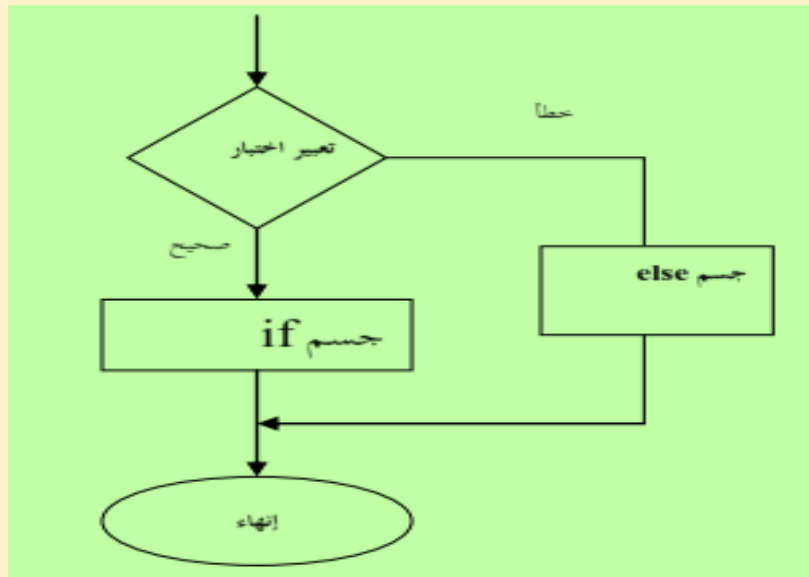
أولاً يأتي تعبير الاختبار، ثم علامة الاستفهام، ثم قيمتان تفصلهما نقطتان. إذا كان تعبير الاختبار صحيحاً ينتج التعبير بأكمله القيمة الموجودة قبل النقطتين وإذا كان تعبير الاختبار خطأ ينتج التعبير بأكمله القيمة التي تلي النقطتين.

ال مثال التالي يحسب القيمة المطلقة (Absolute value) وهي تساوي سالب العدد إذا كان العدد أقل من الصفر وتساوي موجب العدد إذا كان العدد أكبر من الصفر.

$Abs\_value = (n < 0) ? -n : n;$

النتيجة هي  $-n$  إذا كان  $n$  أقل من  $0$  و  $n$  في الحالة الأخرى.

والشكل التالي يوضح طريقة عمل هذه الجملة



ما هو الخطأ في الآتي ؟

```
if (gender==1)
  cout<<women <<endl;
else
  cout <<man<<endl;
```

العبارات `if ... else` المتداخلة:-  
يمكن وضع العبارات `if .....else` ضمن بعضها البعض ،  
البرنامج التالي يوضح ذلك:

```

#include <iostream.h>
main ( )
{
int grade;
cout <<"Enter the grade:" ;
cin >> grade;
if(grade>= 75)
cout<<'A'<< endl;
else
if(grade>= 65)
cout<<'B'<< endl;
else
if(grade>= 55)
cout<<'C'<< endl;
else
if(grade>= 40)
cout<<'D'<< endl;
else
cout<<"fail"<<endl;
return 0;
}

```

تنتهي العبارات المتداخلة في الجسم else وليس في الجسم if ،  
يمكن أن تحدث مشكلة عندما نضع العبارات if .....else ضمن بعضها البعض. فمثلاً المفروض من العبارات التالية أن تعرض الكلمة infant عندما يكون عمر الشخص أقل أو يساوي 2:-

```

if (age >2)
if (age<18)
cout <<"\n child";
else
cout <<"\n infant";

```

ولكن هنا لن يحدث ، ستظهر الكلمة infant كلما كان العمر أكبر أو يساوي 18 وذلك لأن الجزء else يتبع أقرب عبارة if إليه والتي ليس لها جزء else خاص بها. لذا إذا كنا نريد جزء else تابع لعبارة if غير موجودة قبله مباشرة علينا حصر العبارة if الموجودة بينهما بأقواس حاصرة .

```

if (age >2)
{
if (age<18)
cout <<"\n child";
} else
cout <<"\n infant";

```

## \*\* جملة Switch

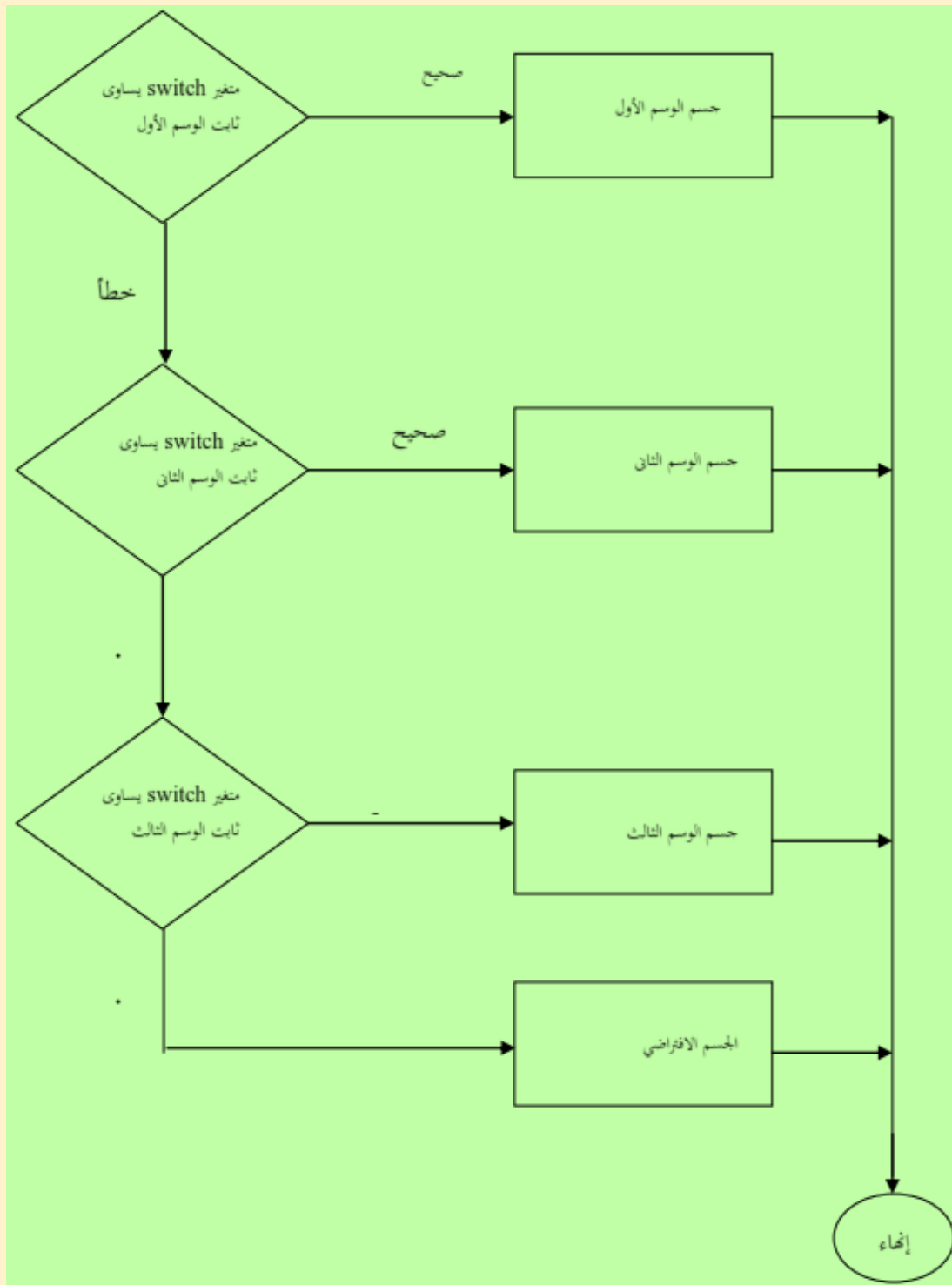
تأخذ جملة switch في C++ الشكل العام التالي:-

```
Switch (Variable name)
{
case constant1 : statement1; break;
case constant2 : statement2; break;
.
.
case constant n : statement n; break;
default : last statement;
}
```

تتألف العبارة switch من الكلمة الأساسية switch يليها اسم متغير بين قوسين، ثم جسمها بين أقواس حاصرة ، تفحص العبارة switch المتغير وتوجه البرنامج نحو أقسام مختلفة وفقاً لقيم ذلك المتغير. يتضمن جسم العبارة switch عدداً من الوسوم وهـ ي أسماء تليها نقطتان. تتألف هذه الوسوم من الكلمة الأساسية case ثم ثابت ثم نقطتين. عندما تكون قيمة متغير العبارة switch مساوية للثابت المذكور في أحد وسوم case ينتقل التنفيذ إلى العبارات التي تلي ذلك الوسم وتؤدي العبارة break إلى منع تنفيذ بقية العبارة switch، وإذا لم تتطابق قيمة متغير العبارة switch مع أي وسم ينتقل التنفيذ إلى الوسم الافتراضي default .

سنقوم بكتابة برنامج لحساب عدد حروف العلة (vowels letters) وهي (a, e, i, u, o) في نص مدخل من لوحة المفاتيح . يقوم البرنامج بفحص الحرف المدخل فإذا كان الحرف a تتم إضافة 1 إلى acounter والذي تم تمهيده عند 0 . أما إذا كان الحرف المدخل e فنتم إضافة 1 إلى ecounter وهكذا بالنسبة لـ u و i و o ، إذا لم يكن الحرف المدخل حرف علة يتم تنفيذ الوسم الافتراضي والذي يقوم بإضافة 1 لـ OtherLettersCounter .

والشكل التالي يوضح طريقة عملها



واليك المثال التالي

```

#include <iostream.h>
enum vowels{a='a',u='u',i='i',o='o',e='e'};
main( )
{
char ch ;
int acounter=0,ecounter=0,icounter=0;
int ucounter=0,ocounter=0,otherletterscounter=0;
while(cin>>ch)
switch(ch) {
case a:
++acounter;
break;
case e:
++ecounter;
break;
case i :
++icounter;
break;
case o:
++ocounter;
break;
case u:
++ucounter;
break;
default:
++ otherletterscounter;
};
cout<<endl;
cout<<endl;
cout<<endl;
cout <<"acounter: \t"<<acounter<<" \n";
cout<< "ecounter: \t"<<ecounter<<" \n";

```

```
cout<< "icounter: \t"<<icounter<<" \n";
cout<< "ocounter: \t"<<ocounter<<" \n";
cout<< "ucounter: \t"<<ucounter<<" \n";
cout<<"otherletterscounter: \t"<<otherletterscounter
    <<" \n";
return 0;
}
```

فإذا كان النص المدخل

“you are very punctional”

تكون المخرجات

```
acounter:      2
ecounter:      2
icounter:       1
ocounter:       2
ucounter:       2
OtherLettersCounter: 11
```

## تدريبات

### 1/ أكتب عبارة C++ تؤدي التالي:

- إدخال قيمة متغير صحيح x باستخدام cin و >> .
- إدخال قيمة متغير صحيح y باستخدام cin و >>.
- تمهيد قيمة متغير صحيح i عند 1.
- تمهيد قيمة متغير صحيح power عند 1.
- ضرب قيمة المتغير x في المتغير power وتعيد النتيجة للمتغير .power
- زيادة قيمة المتغير y ب 1.
- اختبار ما إذا كانت قيمة المتغير y أقل من أو تساوي x.
- طباعة قيمة المتغير .power

2/ بافتراض أن  $x = 9$  و  $y = 11$  ما هي مخرجات الجزء التالي من البرنامج:

```
if ( x < 10)
if ( y > 10)
    cout << " * * * * *" << endl;
else
    cout << " # # # # #" << endl;
    cout << " $ $ $ $ $" << endl;
```

### 3/ اشرح البرنامج ووضح مخرجاته

```
switch (i)
{grade
  case 1 : grade = ' A ' ;
           break;
  case 2 : grade = ' B ' ;
           break;
  case 3 : grade = ' C ' ;
           break;
  default : cout << i
            << "not in range";
}
```



## الفصل الثالث

### أوامر التكرار

#### (الدورات)

### The Looping

#### \*\* عوامل التعيين الحسابي

باستعمال عوامل التعيين الحسابي يمكن إعادة كتابة تعبير مثل:

$$x=x+2$$

على النحو

$$x+=2$$

يأخذ عامل التعيين الحسابي += القيمة الموجودة على يمينه ويضيفها إلى المتغير

الموجود على يساره. هنالك تعين حسابي لكل من العوامل الحسابية:-

$a+= b$	→	$a= a+ b$
$a-= b$	→	$a= a- b$
$a*= b$	→	$a= a* b$
$a/= b$	→	$a= a/ b$
$a\%= b$	→	$a= a\% b$

مثال:

```
#include<iostream.h>
main ( )
{
int n;
cin >> n;
cout<< " n after adding 2 = " << a+= 2 <<endl;
cout<< " n after a subtracting 2 = " << a-= 2 <<endl;
cout<< " n after dividing by 2 = " << a/= 2 <<endl;
cout<< " n after multiplying by 2 = " << a*= 2 <<endl;
cout<< " n mod 2 = " << a %= 2 <<endl;
return 0;
}
```

تكون المخرجات:

وإدخال N=10

10

n after adding 2 = 12  
n after a subtracting 2 = 8  
n after dividing by 2 = 5  
n after multiplying by 2 = 20  
n mod 2 = 0

## \*\* عوامل التزايد والتناقص

هناك دائماً حاجة في البرمجة إلى زيادة 1 أو طرح 1. هذه الحالات شائعة لدرجة أن C++ تتضمن عاملين خاصين ينفذان هذه المهمة، يقوم عامل التناقص (-- ) بطرح 1 من المتغير ويضيف عامل التزايد (++) 1 إليه ، المثال الآتي يبين طريقة الاستعمال:-

**++a**

**a++**

معناه إضافة 1 إلى a ، ويمكن كتابته بصورة مكافئة على النحو  $a=a+1$

وبالطريقة نفسها يمكن إنقاص 1 من قيمة a على النحو --a أو a-- وهو يكافئ  $a=a-1$ .

ومما يجب التنبيه إليه هنا أن هنالك فرق بين ++ a أو a++ فعلى الرغم من

كليهما يجمع 1 إلى a إلا أنه عند استعمال ++a تستخرج قيمة التعبير باستعمال قيمة a

الحالية قبل زيادتها وينطبق هذا أيضاً على --a و a-- .

مثال

```
#include<iostream.h>
main ( )
{
int c;
c = 5;
cout << c << endl;
cout << c++ <<endl;
cout << c <<endl;
c=5;
cout << c << endl << endl;
cout << ++c << endl;
```

```
cout << c << endl;
return 0;
//Continued
}
```

وتكون المخرجات

```
5
5
6

5
6
6
```

## \*\* العوامل المنطقية

يمكن العمل على القيم صحيح/خطأ بواسطة العوامل المنطقية ، هنالك ثلاثة عوامل منطقية في C++ هي Not,Or,And كما موضح في الجدول أدناه:-

العامل المنطقي	معناه	مثال
&&	(and) (و)	$x > 0 \ \&\& \ x < 10$
	(or) (أو)	$x = 0 \    \ x = 1$
!	(not) (نفي)	$!x$

يكون التعبير and صحيحاً فقط إذا كان التعبيرين الموجودان على جانبي العامل && صحيحين بينما يؤدي العامل or إلى نتيجة صحيحة إذا كان أحد التعبيرين أو كليهما صحيحاً. العامل not (!) يبطل تأثير المتغير الذي يليه لذا التعبير  $!x$  صحيح إذا كان المتغير x خطأ وخطأ إذا كان x صحيحاً.

### أولوية العوامل (Operator Precedence):-

يتم تنفيذ عمليات الضرب والقسمة في التعبيرات الرياضية قبل عمليات الجمع والطرح . في التعبير التالي مثلاً :

$$10 * 10 + 2 * 3$$

يتم ضرب  $10 * 10$  ثم يتم ضرب  $2 * 3$  وبعدها يتم جمع نتيجتي الضرب مما يؤدي إلى القيمة  $100 + 6 = 106$ .

يتم تنفيذ عمليات الضرب قبل الجمع لأن العامل \* له أولوية أعلى من أولوية العامل + . نحدد أن أولوية العوامل مهمة في التعبيرات الرياضية العادية كما أنها مهمة أيضاً عند استعمال عوامل C++ المختلفة ، الجدول التالي يبين ترتيب أولويات العوامل في C++ من الأعلى إلى الأدنى.

العوامل	أنواع العوامل	الأولوية
* , / , %	مضاعفة	أعلى
+ , -	جمعية	
< , > , <= , >= , == , !=	علائقية	
&&    !	منطقية	
=	تعيين	أدنى

## \*\* الحلقات Loops

توفر **C++** عدداً من أساليب التكرار (حلقات) التي تستخدم لتكرار أجزاء من البرنامج قدر ما تدعو الحاجة، لتحديد عدد مرات تكرار الحلقة تفحص كل حلقات **C++** ما إذا كان تعبير ما يساوي صحيح (**true**) أو خطأ (**false**) يبلغها هذا ما إذا كان عليها التكرار مرة إضافية أخرى أو التوقف فوراً. هنالك ثلاثة أنواع من الحلقات في **C++**:-

### \* الحلقة **while**

تأخذ التعليمة **while** الشكل التالي:

**while (condition)**

**statement;**

حيث يكون الشرط **condition** إما متحولاً منطقياً أو صيغة منطقية. أما **statement** فهو إما تعليمة وحيدة تنتهي بفاصلة منقوطة، أو مجموعة من التعليمات

تتيح الحلقة **while** تكرار فعل جزء من البرنامج إلى أن يتغير شرط ما .

فمثلاً:-

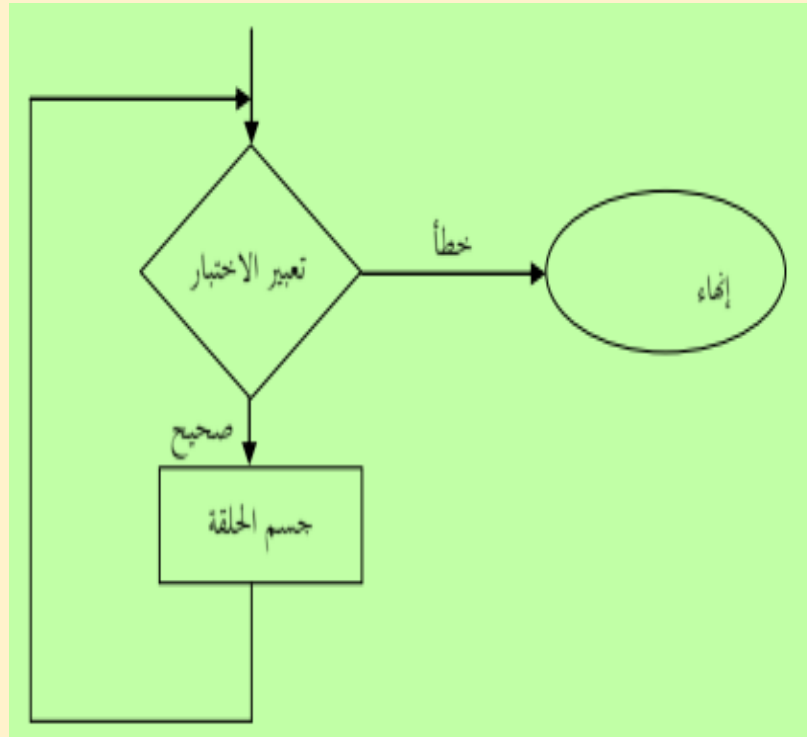
```
while (n<100)
```

```
n=n*2
```

ستستمر هذه الحلقة في مضاعفة المتغير **n** إلى أن تصبح قيمة **n** أكبر من 100 عندها تتوقف . تتكون الحلقة من الكلمة الأساسية **while** يليها تعبير اختبار بين أقواس ويكون جسم الحلقة محصوراً بين أقواس حاصرة { } إلا إذا كان يتألف من عبارة واحدة.

مما يجدر التنويه إليه هنا أنه يتم فحص تعبير الاختبار قبل تنفيذ جسم الحلقة،  
وعليه لن يتم تنفيذ جسم الحلقة أبداً إذا كان الشرط خطأ عند دخول الحلقة وعليه المتغير  $n$  في  
المثال السابق يجب تمهيده عند قيمة أقل من 100 .

والشكل التالي يوضح كيفية عملها:



مثال

```
sum = 0.0 ;  
cin>> x;  
While (x>0.0)  
{  
    sum += x;  
    cin >> x;  
}
```

```
#include<iostream.h>
main ( )
{
int counter, grade, total ,average;
total = 0;
counter = 1;
while (counter <= 10) {
cout<< " Enter grade : ";
cin >>grade;
total = total + grade;
```

```
counter = counter + 1;
}
cout<<endl;
average = total /10;
//Continued
cout << " Class average is: " << average <<endl;
return 0;
```

وتكون مخرجات البرنامج

```
Enter grade: 75 65 50 89 71 54 86 79 81 90
Class average is : 74
```

ما هو الخطأ في الحلقة الآتية:

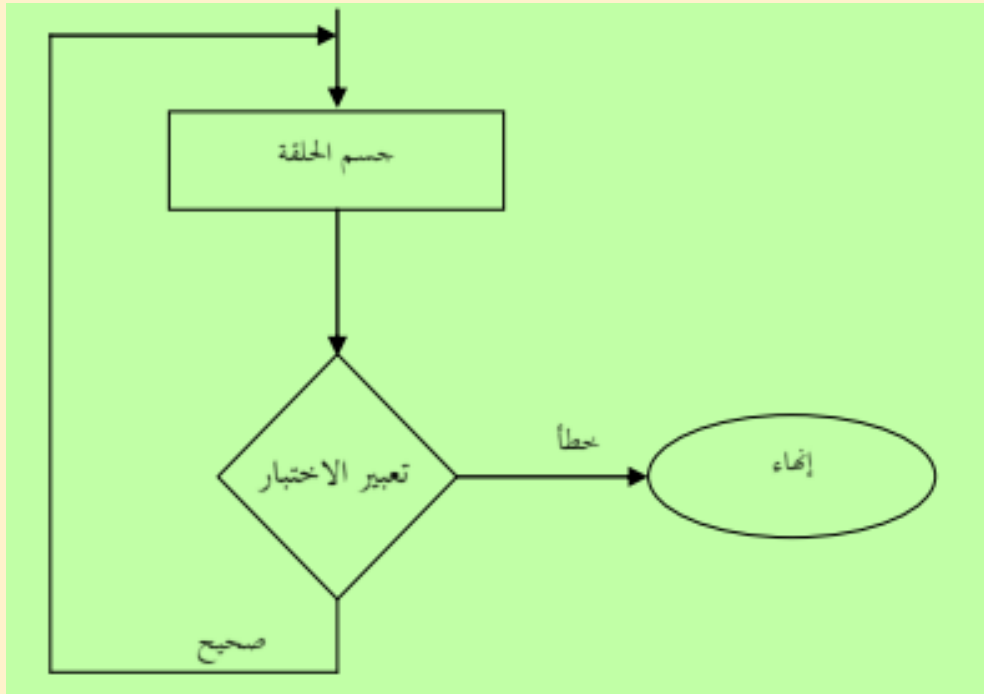
```
while(c<5) {
product *=c;
++c;
```



## الحلقة do.....while

تعمل الحلقة do (غالباً تسمى do...while...) كالحلقة while، إلا أنها تفحص تعبير الاختبار بعد تنفيذ جسم الحلقة. وتستخدم أيضاً عندما نريد القيام بجزء من البرنامج مرة واحدة على الأقل.

والشكل التالي يوضح كيفية عملها



تبدأ الحلقة do بالكلمة الأساسية do يليها جسم الحلقة بين أقواس حاصرة { ثم الكلمة الأساسية while ثم تعبير اختبار بين أقواس ثم فاصلة منقوطة.

مثال:-

البرنامج التالي يقوم بطباعة الأعداد من 1 إلى 10 .

```
// using do repetition structure
#include<iostream.h>
main ( )
{ int counter = 1;
do
cout << counter << "  " ;
```

```
while ( + + counter <= 10);
//Continued
return 0;
}
```

تقوم `cout << " " ;` بطباعة مسافة بحالية بين كل رقم والآخر وعليه الخرج من البرنامج يكون كالتالي:

1 2 3 4 5 6 7 8 9 10

**for** \* الحلقة \*

تأخذ التعليمة **for** الشكل التالي

**for( initialise ; test ; update )**

**statement;**

حيث يمثل **initialise** عملية إعطاء قيمة ابتدائية لمتحول التحكم بالحلقة

ويمثل **test** عملية تقويم الصيغة المنطقية

ويمثل **update** عملية تعديل قيمة متحول التحكم بالحلقة قبل إعادة تنفيذها ثانية

فمثلا لطباعة الأرقام من 1 الى 10 باستخدام العبارة **for**

```
for(i = 1 ; i <= 10; i++)
{
    cout<< i << endl;
}
```

وهو مساوى للبرنامج بعبارة **while**

```
i = 1;
while ( i <= 10)
{
    cout << i <<endl;
    i++;
}
```

عادة لا تعرف الحلقات **do** و **while** عدد مرات تكرار الحلقة. لكن في الحلقة **for** يكون عدد مرات تنفيذ الحلقة مذكوراً عادة في بدايتها. المثال التالي يقوم بطباعة قيم المتغير **counter** من 1 إلى 10 .

```
// using the for repetition structure
#include<iostream.h>
main( )
{
for ( int counter = 1; counter<= 10; counter++)
cout << counter <<endl ;
return 0;
}
```

1  
2  
3  
4  
5  
6  
7  
8  
9  
10

تحتوي الأقواس التي تلي الكلمة الأساسية **for** على ثلاثة تعابير مختلفة تفصلها فاصلة منقوطة. تعمل هذه التعابير الثلاثة في أغلب الأوقات على متغير يدعى متغير الحلقة ، وهو المتغير **counter** في المثال السابق.  
هذه التعابير هي:-

تعبير التمهيد، الذي يمهد قيمة متغير الحلقة عادة `int counter = 1;`  
تعبير الاختبار، الذي يفحص عادة قيمة متغير الحلقة ليرى ما إذا كان يجب تكرار الحلقة مرة أخرى أو إيقافها `counter <= 10;`  
تعبير التزايد، الذي يقوم عادة بزيادة (أو إنقاص) قيمة متغير الحلقة `counter++`.  
المثال التالي يقوم بإنقاص متغير الحلقة بـ 1 كلما تكررت الحلقة

```
#include <iostream.h>
main ( )
{
    for ( int j=10; j>0; -- j)
        cout <<j<<' ';
    return 0;
}
```

وتكون المخرجات

1 2 3 4 5 6 7 8 9 10

ويمكن أيضاً زيادة أو إنقاص متغير الحلقة بقيمة أخرى .  
البرنامج التالي يوضح ذلك :

```
#include<iostream.h>
main ( )
{
```

```
for (int j=10; j<100; j+=10)
cout <<j<<'  ';
return 0;
{
```

وتكون المخرجات

```
10 20 30 40 50 60 70 80 90 100
```

يمكن استعمال عدة عبارات في تعبير التمهيد وتعبير الاختبار كما في البرنامج التالي :-

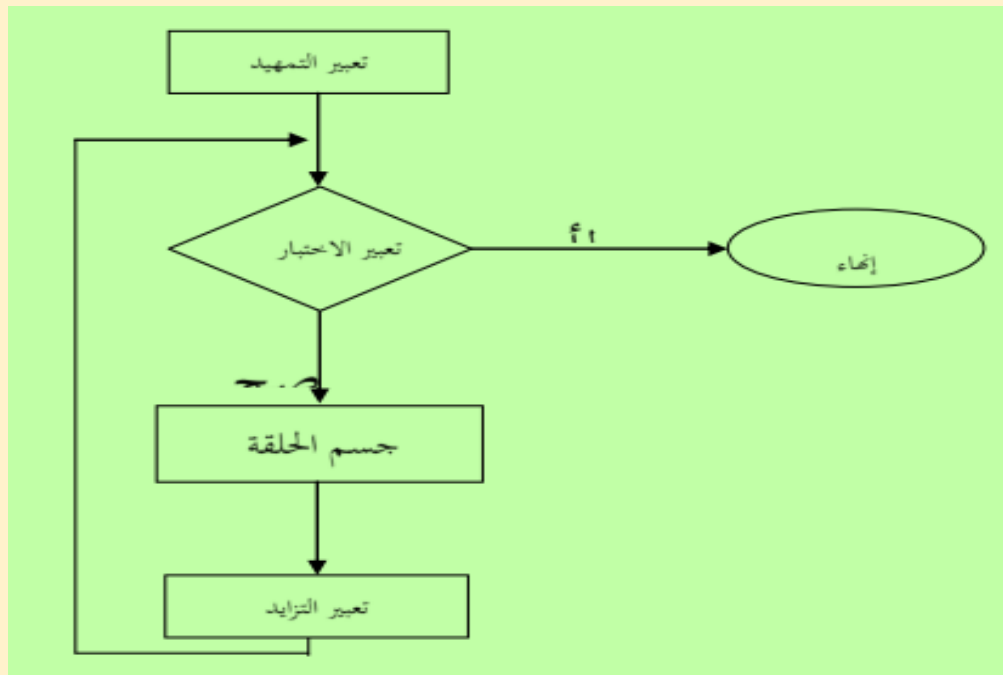
```
#include<iostream.h>
main ( )
{
for ( int j=0;int total=0; j<10; ++ j;total+=j)
cout <<total<<'  ';
return 0 ;
}
```

وتكون المخرجات

0 1 3 6 10 15 21 28 36 45

أيضاً يمكن في الحلقة for تجاهل أحد التعابير أو ثلاثتها كلياً مع المحافظة على الفواصل المنقوطة فقط.

والشكل التالي يوضح كيفية عملها



\*\*الحلقات المتداخلة

```
// An Example of 2 nested loops
#include<iostream.h>
main( )
{
int i,j;
for (i=1 ; i<3;++i)
{
for (j=1 ; j<4;++j)
cout << i*j<<' ' <<endl;
}
return 0;
}
```

نلاحظ هنا أن الحلقة الداخلية تتكرر 4 مرات لكل قيمة من قيم  $i$  (عدد الحلقة

الخارجية).

الخروج من البرنامج:

```
1 2 3 4
2 4 6 8
   6 9 12
```

يمكننا وضع أي نوع من الحلقات ضمن أي نوع آخر، ويمكن مداخله الحلقات في حلقات متداخلة في حلقات أخرى وهكذا.

### التحكم بالحلقات

تعمل الحلقات عادة بشكل جيد إلا أننا في بعض الأوقات نحتاج للتحكم بعمل الحلقات ، العبارتين **break** و **continue** توفران هذه المرونة المطلوبة.

#### العبارة **break** :-

تتيح لنا العبارة **break** الخروج من الحلقة في أي وقت.

المثال التالي يبين لنا كيفية عمل العبارة **break** :

```

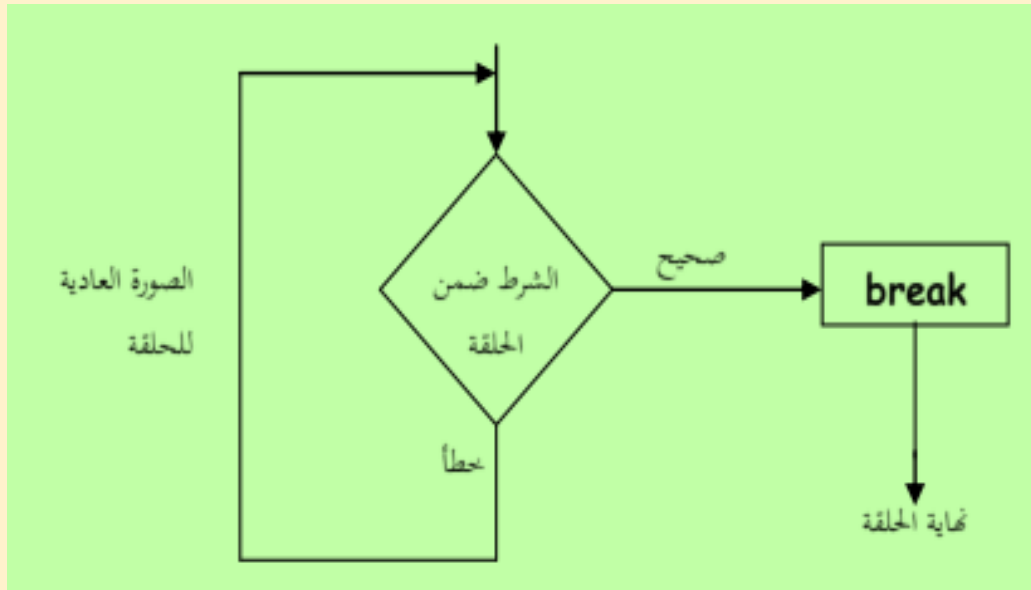
//An Example on break as a loop exit
#include<iostream.h>
main( )
{
int isprime ,j ,n;
isprime = 1;
cin>>n;
for (j=2,j<n;++j)
{
if (n%j== 0)
{
isprime =0;
break;
}
}
}
}

```

هذا البرنامج يجعل قيمة المتغير **isprime** عند 1 إذا كان **n** عدد أولي **prime** يجعل قيمته 0 إذا لم يكن كذلك ( الرقم الأولي هو رقم يقبل القسمة على نفسه وعلى الرقم واحد فقط). لمعرفة ما إذا كان الرقم أولياً أم لا تتم قسمته على كل الأرقام وصولاً إلى **n-1** ، إذا قبل الرقم **n** القسمة على أحد هذه القيم من دون باقي فإنه لا يكون أولياً لكن إذا قبل الرقم **n** القسمة على أحد هذه القيم بشكل صحيح لا داعي لإكمال الحلقة فحالما يجد البرنامج الرقم الأول الذي يقسم **n** بشكل صحيح يجب أن يضبط قيمة المتغير **isprime** عند 0 ويخرج فوراً من الحلقة.

والشكل التالي يوضح كيفية عملها





### العبارة continue :-

تعيد العبارة `continue` التنفيذ إلى أعلى الحلقة.

المثال التالي يوضح كيفية عمل العبارة `continue` :-

```

//An Example on continue statement
#include<iostream.h>
main( )
{
int dividend , divisor;

```

```

do
//Continued
{
cout << "Enter dividend:  ";
cin>>dividend;
cout<< "Enter divisor:  ";
//Continued
cin>>divisor;
if( divisor == 0)
{
cout<<" divisor can't be zero\n" ;
continue;
}
cout <<"Quotient is "<< dividend/divisor;
cout<<" do another (y/n)?";
cin>>ch
}
while (ch!= 'n');
}

```

القسمة على 0 أمر غير مرغوب فيه لذا إذا كتب المستخدم 0 على أنه القاسم يعود التنفيذ إلى أعلى الحلقة ويطلب من البرنامج إدخال قاسم ومقسوم جديدين.

```

If ( divisor == 0)
{
cout << "divisor can't be zero\n";
continue;
}

```

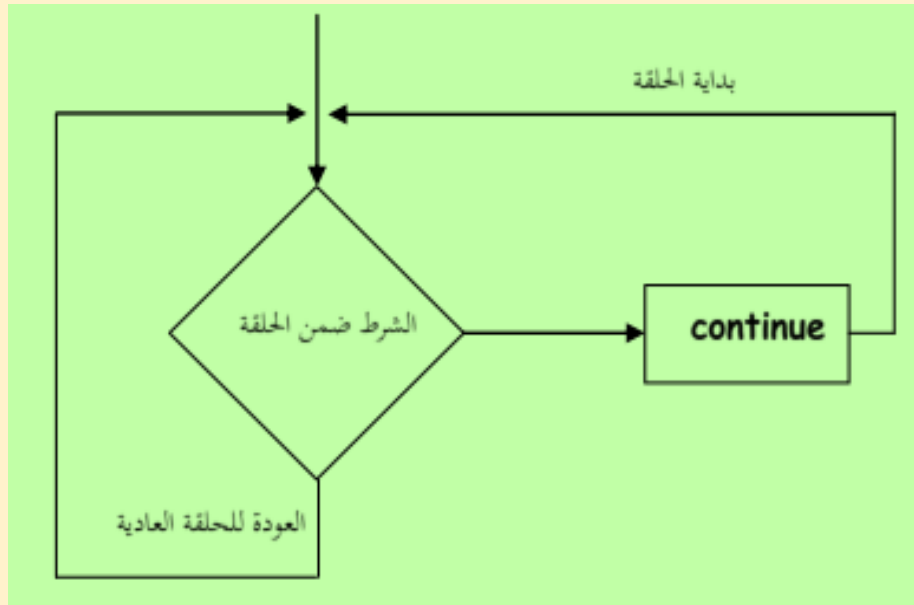
يستمر تنفيذ الحلقة إلى أن يدخل المستخدم الحرف n .

```

while( ch != 'n' );

```

والشكل التالي يوضح كيفية عملها





الفصل الرابع

المصفوفات والمؤشرات

Arrays & Pointers

المصفوفة هي نوع من أنواع بنية البيانات ، لها عدد محدود ومرتب من العناصر التي تكون جميعها من نفس النوع **type**، فمثلاً يمكن أن تكون جميعها صحيحة **int** أو قائمة **float** ولكن لا يمكن الجمع بين نوعين مختلفين في نفس المصفوفة .

الشكل التالي يبين مصفوفة **C** تحتوي على 13 عنصر من النوع **int**، ويمكن الوصول إلى أي من هذه العناصر بذكر اسم المصفوفة متبوعاً برقم موقع العنصر في المصفوفة محاطاً بـ [ ] .

يرمز لرقم العنصر في المصفوفة بفهرس العنصر **index** . فهرس العنصر الأول في المصفوفة هو **0** ولهذا يشار إلى العنصر الأول في المصفوفة **C** بـ **C[0]** والثاني **C[1]** والسابع **C[6]** وعموماً يحمل العنصر **i** في المصفوفة **C** الفهرس **C[i-1]** .  
تتبع تسمية المصفوفات نفس قواعد تسمية المتغيرات.

<b>C[0]</b>	<b>-45</b>
<b>C[1]</b>	<b>6</b>
<b>C[2]</b>	<b>0</b>
<b>C[3]</b>	<b>72</b>
<b>C[4]</b>	<b>1543</b>
<b>C[5]</b>	<b>-89</b>
<b>C[6]</b>	<b>0</b>
<b>C[7]</b>	<b>62</b>
<b>C[8]</b>	<b>-3</b>
<b>C[9]</b>	<b>1</b>
<b>C[10]</b>	<b>6453</b>
<b>C[11]</b>	<b>78</b>
<b>C[12]</b>	<b>15</b>

أحياناً يسمى فهرس العنصر برمز منخفض **subscript** ويجب أن يكون الفهرس **integer** أو تعبير جبري تكون نتيجته **integer** . فمثلاً إذا كانت **a=5** و **b=6** فالعبارة:  
**C[a+b]+=2,**

◆ تقوم بإضافة 2 إلى العنصر الثاني عشر **C[11]** في المصفوفة **C** .

- ◆ يحمل العنصر 0 في المصفوفة C القيمة -45 والعنصر 1 القيمة 6. لطباعة مجموع الثلاثة عناصر الأولى في المصفوفة C يمكن كتابة:

```
cout<<C[0]+C[1]+C[2]<<endl;
```

## الإعلان عن المصفوفات:-

تحتل المصفوفات حيزاً في الذاكرة لذا يجب على المبرمج تحديد نوع عناصر المصفوفة وعددها حتى يتسنى للمعرف تخصيص الحيز اللازم من الذاكرة لحفظ المصفوفة ، وحتى تخبر المبرمج بأن يخصص حيزاً لـ 12 عنصر من النوع int في مصفوفة C ، استخدم الإعلان:

```
int C[12];
```

يمكن تخصيص الذاكرة لعدة مصفوفات باستخدام نفس الإعلان وذلك كالآتي:

```
int b[100], x[20];
```

أيضاً يمكن الإعلان عن مصفوفات من أي نوع بيانات آخر ، فمثلاً للإعلان عن

مصفوفة عناصرها من النوع char نكتب:

```
char ch[20];
```

مثال عن استخدام المصفوفات:

يستخدم البرنامج التالي حلقة for لتمهيد عناصر المصفوفة n عند 0 وطباعة عناصر المصفوفة.

```

//initializing an array
#include <iostream.h>
#include <iomanip.h>
main( )
{
int n[10];
  for (int i=0; i<10;i++) // initialize array
n[i] = 0;
cout << "Element" << setw(13) << " value" << endl;
for (i=0 ; i< 10; i++) // print array
cout << setw(7) <<i<<setw(13) <<n[i]<<endl;
return 0;
}

```

وتكون المخرجات

Element	Value
	0
	0
	0
	0
	0
	0
	0
	0
	0
	0
9	0



في البرنامج السابق تم تضمين الملف `iomanip.h` وذلك لأننا استخدمنا المناور `setw(13)` والذي يعني ضبط عرض الحقل عند 13 ( أي أن القيمة التي ستتم طباعتها ستكون على بعد 13 مسافة من القيمة التي تمت طباعتها قبلها ) .

يمكن تمهيد عناصر المصفوفة باتباع الإعلان عن المصفوفة بعلامة المساواة (=) تليها لائحة من القيم المطلوب تمهيد عناصر المصفوفة عندها ، ويتم الفصل بين القيم بفواصل ، وتحيط هذه اللائحة الأقواس الحاصرة { } . البرنامج التالي يقوم بتمهيد عناصر من النوع `integer` لتحتوي قيم محددة عند الإعلان عن المصفوفة، وطباعة هذه القيم.

```
//initializing an array with a declaration
#include <iostream.h>
#include <iomanip.h>
main( )
{
int n[10] = {32,27,64,18,95,14,90,70,60,37};
cout << "Element" << setw(13) << " value" << endl;
for (i=0 ; i< 10; i++) // print array
cout << setw(7) <<i<<setw(13) <<n[i]<<endl;

return 0;
```

إذا كانت قيم التمهيدي الموجودة في اللائحة أكثر من حجم المصفوفة المحدد سيعترض  
المصرف، وإذا كانت أقل سيملاً المصرف بقية العناصر أصفاراً، لذا إذا كنا نريد تمهيدي عناصر  
مصفوفة مهما كان حجمها بأصفار كل ما علينا فعله هو كتابة إعلان كالتالي:-

```
int anyarray[10]={0};
```

سيتم تمهيدي العنصر الأول عند القيمة 0 التالي كتبناها والعناصر المتبقية عند 0  
كوننا لم نحدد قيمة لها.

البرنامج التالي يقوم بجمع 12 عنصر في مصفوفة من النوع int .

```
// compute the sum of the elements of the array
#include <iostream.h>
main( )
{
const int arraysize =12;
int a[arraysize] = {1, 3, 5, 4, 7, 2, 99, 16, 45, 67, 89, 45};
int total = 0;
for (int i= 0; i<arraysize ; i++)
total += a[i];
cout <<" total of array element values is " << total << endl;
return 0;
}
```

وتكون المخرجات

```
total of array element values is 383
```

نلاحظ أننا في العبارة:

```
const int arraysize = 12;
```

استعملنا كلمة جديدة هي `const` . يتم استعمال هذه الكلمة الأساسية في تعريف المتغير الذي لا يمكن تغيير قيمته في البرنامج ولذلك يجب تمهيده عند قيمة أولية عند تعريفه ( في البرنامج السابق تم تمهيده لـ 12 )

كما ذكرنا أنه يمكن تعريف مصفوفات من أي نوع بيانات آخر، سنقوم الآن

بتخزين سلسلة حروف في مصفوفة من النوع `char`.

يتم تمهيد المصفوفة من النوع `char` باستخدام ما يسمى بالثابت السلسلي

(string literal)

```
char string1[ ]="first";
```

حجم المصفوفة `string1` هنا يتم تحديده بواسطة المصرف بناءً على طول الثابت

السلسلي `"first"`.

من المهم هنا أن نذكر أن السلسلة `"first"` تحتوي على خمسة عناصر زائداً حرفاً

خامداً يشير إلى نهاية السلسلة ويسمى الحرف الخامد `null character` ويتم تمثيله باستخدام

تتابع الهروب `'\0'` وتنتهي كل السلاسل بهذا الحرف الخامد وعليه فإن المصفوفة `string1`

تحتوي على ستة عناصر.

يمكن أيضاً تمهيد السلسلة "first" باستخدام لائحة قيم تفصلها فواصل لذا

الإعلان:-

```
char string1[ ]="first";
```

يكافئ:

```
char string1[ ]={'f','i','r','s','t','\0'}
```

وبما أن السلسلة في الواقع هي مصفوفة أحرف ، عليه يمكن الوصول إلى أي حرف

من حروف السلسلة مباشرة باستخدام الفهرس واسم المصفوفة ، فمثلاً `string1[0]='f'`

ومثلما يمكن تمهيد السلسلة عند الإعلان عنها ، يمكن أيضاً إدخال السلاسل عن طريق لوحة

المفاتيح باستعمال `cin` و `>>` فمثلاً الإعلان :-

```
char string2[20];
```

ينشئ مصفوفة أحرف تسمح بتخزين 19 حرفاً إضافة إلى الحرف الخامد والعبارة

```
cin>>string2;
```

تقوم بتخزين السلسلة المدخلة عن طريق لوحة المفاتيح وتخزينها في المصفوفة

```
.string2
```

يمكن خرج السلسلة المخزنة في مصفوفة الأحرف باستخدام `cout` و `<<` وعليه

يمكن طباعة المصفوفة `string2` باستخدام العبارة:-

```
cout << string2 << endl;
```

cout مثل cin لا تحتم بحجم المصفوفة حيث تقوم بطباعة حروف السلسلة حتى

تصل إلى الحرف الخامد الذي يحدد نهاية السلسلة.

البرنامج التالي يقوم بتمهيد مصفوفة أحرف عند ثابت سلسلي ويقوم باستعمال حلقة التكرار

for للوصول إلى عناصر المصفوفة وطباعتها .

مثال

```
//Treating character arrays as strings
#include<iostream.h>
main( )
{
char string1[20], string2[ ] = " stringliteral" ;
cout << "Enter a string: ";
cin>> string1;
cout << "string1 is : " << string1<<endl
    << "string2 is : " << string2<<endl
    << "string1 with spaces between characters is: "
    << endl;
for (int i= 0; string1[i] ; = '\0' ; i++)
    cout << string1[i]<< ' ';
cout << endl;
//Continued
return 0;
}
```

Hello there

يفترض أن المستخدم قد أدخل السلسلة

تكون المخرجات

Enter a string: Hello there

string1 is : Hello

string2 is : string Literal

string1 with spaces between characters is : H e l l o

استخدمت حلقة التكرار `for` للوصول إلى حروف السلسلة `string1` وطباعتها مع طباعة مسافة بين كل حرف والآخر حتى تصل إلى الحرف الخامد `'\0'` ( `string1[i] != '\0'` والذي يحدد نهاية السلسلة. )

مكتبة دالات السلاسل

توجد عدة دالات تعمل على السلاسل، إذا أردنا استعمال أي من هذه الدوال في برنامج يجب أن نقوم بتضمين ملف الترويسة `string.h` . من هذه الدالات :

`strlen()` /1-

تعيد الدالة `strlen()` طول السلسلة الممررة كوسيط لها، البرنامج التالي يوضح

ذلك :-

```
// using strlen
#include<iostream.h>
#include<string.h>
```

```
main ( )
{
char *string1= " abcdefghijklmnopqrstuvwxyz";
//Continued
char *string2 = "four";
char *string3 = "Boston";
cout << " The length of \ " " << string1
    << " \" is << strlen (string1) <<endl
    << " The length of \" << string2
    <<" \" is << strlen (string2) << endl
    << "The length of \ " " << string3
    << " \" is << strlen( string3) <<endl;
return 0;
}
```

وتكون المخرجات

```
The length of "abcdefghijklmnopqrstuvwxyz" is 26
The length of "four" is 4
The length of "Boston" is 6
```

لاحظ أن الحرف  $\backslash 0$  غير محسوب في الطول الذي تعيده الدالة `strlen` على الرغم من أنه موجود في `s1` ويحتل مكاناً في الذاكرة.

`strcpy( ) / 2`

تستعمل الدالة `strcpy` لنسخ سلسلة إلى سلسلة أخرى



```
// using strcpy
#include<iostream.h>
```

```
#include<string.h>
main ( )
{
char x[ ] = "Happy Birthday to you";
//Continued
char y[25];
cout<<" The string in array x is : "<< x << endl;
cout<<" The string in array y is : "<< strcpy(y, x)
<< endl;
return 0;
}
```

بعد تنفيذ العبارة `strcpy(y, x)` ستحتوى السلسلة `y` على `Happy Birthday to you`. لاحظ هنا أن الدالة `strcpy` تنسخ السلسلة الممررة كالوسيلة الثانية إلى السلسلة الممررة كالوسيلة الأولى.

وتكون المخرجات

```
The string in array x is : Happy Birthday to you
The string in array y is : Happy Birthday to you
```

–: `strcat( )` /3

تقوم الدالة `strcat( )` بإلحاق السلاسل ، الذي يمكن أن يسمى جمع السلاسل  
فمثلاً إذا ألحقنا السلسلة `science` بالسلسلة `computer` ستكون نتيجة السلسلة  
–: `computer science`



```
// using strcat
#include<iostream.h>
#include<string.h>
int main ( )
```

```
{
char s1[20]="computer" ;
char s2[ ]="science" ;
cout<<"s1= " <<s1 << endl << "s2= " << s2 <<endl;
cout<< "strcat(s1, s2) = " << strcat (s1, s2) << endl;
//Continued
return 0;
}
```

وتكون المخرجات

```
s1= computer
s2 = science
strcat(s1, s2)= computerscience
```

–:strcmp( )/4

الدالة strcmp تقارن السلسلة الممارة إليها كوسيلة أولى مع السلسلة الممارة إليها كوسيلة ثانية، وترجع 0 إذا كانتا متطابقتين وقيمة سالبة إذا كانت السلسلة الأولى أصغر من السلسلة الثانية وقيمة موجبة إذا كانت السلسلة الأولى أكبر من السلسلة الثانية. البرنامج التالي يوضح ذلك:

```

// using strcmp
#include<iostream.h>
#include<string.h>
int main ( )
{
char *s1 = " Happy New Year";
char *s2 = " Happy New Year";
char *s3 = " Happy Holidays";
cout << "s1= " << s1<< endl<< "s2= " <<s2<<endl
<< "s3= " << s3<< endl<< endl<< "strcmp(s1, s2)= "
<< strcmp(s1, s2) <<endl<< "strcmp(s1, s3)= "
<< strcmp(s1, s3) <<endl<< "strcmp(s3, s1)= "
<< strcmp(s3, s1) <<endl<< endl;
return 0;
}

```

وتكون المخرجات

```

s1= Happy New Year
s2= Happy New Year
s3 = Happy Holidays

strcmp (s1, s2) = 0
strcmp (s1, s3) = 6
strcmp (s3, s1) = 6

```

## المصفوفات متعددة الأبعاد

يمكن للمصفوفات في `C++` أن تكون متعددة الأبعاد ويمكن كذلك أن يكون كل

بعد بحجم مختلف ، الاستعمال الشائع للمصفوفات متعددة الأبعاد هو تمثيل الجداول **Tables** التالي تحتوي على بيانات مرتبة في صورة صفوف وأعمدة ولتمثيل الجدول نحتاج لبعدين الأول يمثل الصفوف والثاني يمثل الأعمدة.  
الشكل التالي يبين مصفوفة **A** تحتوي على ثلاثة صفوف وأربع أعمدة.

	Column 0	Column1	Column2	Column 3
Row 0	A[0][0]	A[0][1]	A[0][2]	A[0][3]
Row 1	A[1][0]	A[1][1]	A[1][2]	A[1][3]
Row 2	A[2][0]	A[2][1]	A[2][2]	A[2][3]

يتم تمثيل أي عنصر في المصفوفة **A** على الصورة **A[i][j]** حيث:-  
**A** : اسم المصفوفة.

**i** : رقم الصف الذي ينتمي إليه العنصر.

**j** : رقم العمود الذي ينتمي إليه العنصر.

لاحظ أن كل العناصر الموجودة في الصف الأول مثلاً يكون الفهرس الأول لها هو **0** وكل العناصر الموجودة في العمود الرابع يكون الفهرس الثاني لها هو **3**.

يتم الإعلان عن مصفوفة **a** تحتوي على **x** صف و **y** عمود هكذا:

```
int a[x][y];
```

يمكن تمهيد قيمة المصفوفة المتعددة الأبعاد عند الإعلان عنها وذلك كالآتي:

```
int b[2][2]={{1,2},{3,4}};
```

حيث:

```
b[1][1]=4, b[1][0]=3, b[0][1]=2, b[0][0]=1
```

أيضاً هنا في المصفوفة متعددة الأبعاد إذا تم تمهيدها عند قيم لا يتوافق عددها مع حجم المصفوفة فإن المصرف سيملاً بقيمة العناصر أصفار.

البرنامج التالي يوضح كيفية تمهيد مصفوفات متعددة الأبعاد عند الإعلان عنها:

```
// initializing multidimensional arrays
#include<iostream.h>
void printarray(int [ ] [3]);
```

```
int main( )
//continued
{
int array1[2] [3] = { {1, 2, 3}, {4, 5, 6}},
    array2[2] [3] = {1, 2, 3, 4, 5},
    array3[2] [3] = { {1, 2}, {4} };
cout << "values in array1 by row are : " << endl;
printArray(array1);
//Continued
cout << "values in array2 by row are : " << endl;
printArray(array2);
cout << "values in array3 by row are : " << endl;
printArray(array3);
return 0;
}
void printArray(int a[ ][3])
{
for (int i=0; i<1; i++) {
    for (int j=0; j<2; j++)
    cout << a[i][j] << ' ';
    cout << endl;
}
}
```

وتكون المخرجات

values in array 1 by row are:

1 2 3

4 5 6

values in array 2 by row are:

1 2 3

4 5 0

values in array 3 by row are:

1 2 0

4 0 0

## المؤشرات

يستخدم المؤشر في لغة ++C كعنوان لمتغير في الذاكرة ، أحد الاستعمالات المهمة للمؤشرات هو التخصيص الديناميكي للذاكرة حيث يتم استعمال المؤشرات لإنشاء بنية بيانات لتخزين البيانات في الذاكرة. يتم الإعلان عن المؤشرات قبل استخدامها في البرنامج فمثلاً العبارة :

```
int *countptr;
```

تعلن عن مؤشر `countptr` ليشير إلى متغير من النوع `int` (\* المذكورة قبل اسم المؤشر تشير لذلك) وكل متغير يعلن عنه كمؤشر يجب أن يكتب في الإعلان مسبقاً بـ \* فمثلاً الإعلان :

```
float *xptr, *yptr;
```

يشير لأن كلاً من `xptr` و `yptr` موقعي مؤشرات لقيم من النوع `float` ويمكن أن تستخدم المؤشرات لتشير لأي نوع بيانات آخر.

تذكر دائماً عند الإعلان عن أي مؤشر أن تسبق \* كل مؤشر على حدة فمثلاً الإعلان :

```
Int *xptr, yptr;
```

يجب أن تعلن عن هذه المؤشرات كالتالي:

```
int *xptr, *yptr;
```

يمكن تمهيد المؤشرات عند الإعلان عنها عند قيمة 0 أو null أو عند قيمة عنوان في الذاكرة . المؤشر الذي يحمل القيمة 0 أو null لا يشير لأي متغير . تمهيد المؤشر عند 0 يكافئ تمهيده عند null ولكن في ++C يفضل تمهيد المؤشر عند القيمة 0.

عوامل المؤشرات:-

1/ عامل العنوان &:-

العامل & يسمى عامل العنوان وهو عامل أحادي يستعمل لمعرفة العنوان الذي يحتله متغير ما [يرجع عنوان معاملة] فمثلاً إذا استعملنا الإعلان:

```
int y= 5;
```

```
int *yptr;
```

العبارة: `yptr = &y;`

تقوم بتعيين عنوان المتغير `y` للمؤشر `yptr` ويقال أن `yptr` يشير لـ `y` .

## 2/ العامل \* :

العامل \* أيضاً عامل أحادي وهو يرجع القيمة التي يحملها معاملة ، وعليه العبارة

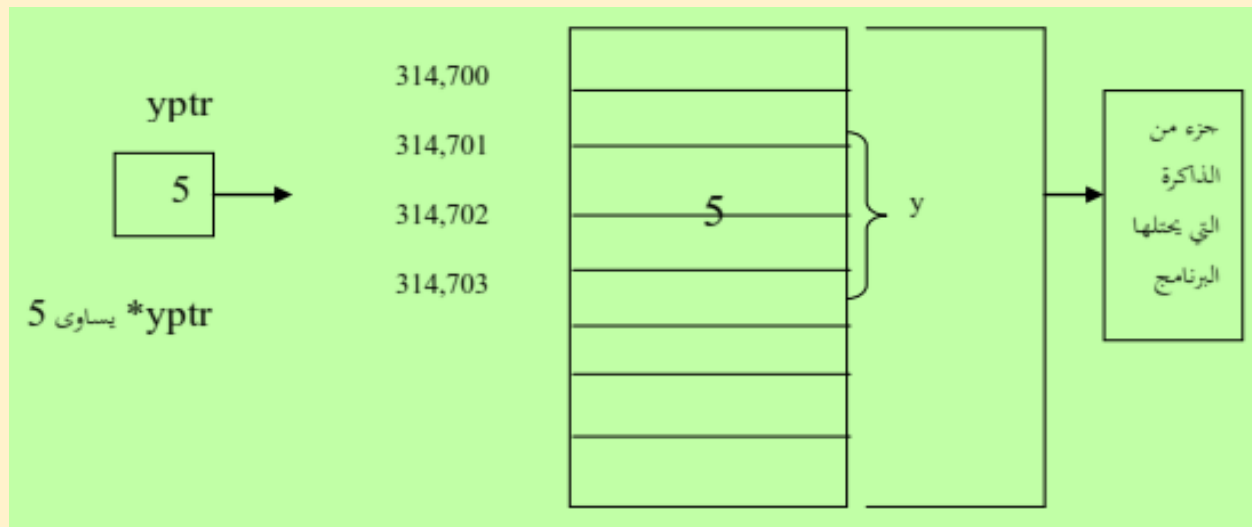
```
cout << * yptr << endl ;
```

تقوم بطباعة قيمة المتغير  $y$  والتي هي 5 .

والعبارة: `cout<<yptr;` تقوم بطباعة القيمة 314,701 والتي هي عنوان المتغير  $y$  ، بعد أن

تم تعيين المتغير  $y$  إلى `yptr` .

والشكل التالي يوضح هذا



وعندما يتم استعمال العامل \* على يسار اسم المتغير كما حصل في التعبير `*yptr` فإنه يسمى عامل المواربة `indirection`.

العامل \* عند استعماله كعامل مواربة له معنى مختلف عن معناه عند استعماله للإعلان عن المتغيرات المؤشرة. يسبق عامل المواربة اسم المتغير ويعني قيمة المتغير المشار إليه. أما \* المستعملة في الإعلان فتعني مؤشر إلى.



Int \*yptr ; (إعلان)

\*yptr=5; (موازية)

البرنامج يوضح استعمال العامل & والعامل \* .

```
// using the & and * operators
#include<iostream.h>
main ( )
{
    int a ;                //a is an integer
    int *aptr;            // aptr is a pointer to an integer
    a = 7;
    aptr = &a;            // aptr set to address of a
    cout << " The address of a is " << &a << endl
         << "The value of aptr is " << aptr << endl << endl;

    cout << "The value of a is " << a << endl
         << "The value of *aptr is " << *aptr << endl << endl;
    cout << " Proving that * and & are complement of "
         << "each other." << endl << " & *ptr = " << & *aptr
         << endl << " *&aptr = " << *&aptr << endl;
    return 0;
}
```

وتكون المخرجات

```
The address of a is 0xffff4
The value of aptr is 0xffff4
```

```
The value of a is 7
The value of *aptr is 7
```

```
Proving that * and & are complements of each other
&* aptr = 0xffff4
*& aptr = 0xffff4
```



## مؤشرات إلى void:-

عادة العنوان الذي نضعه في المؤشر يجب أن يكون من نفس نوع المؤشر، فمثلاً لا يمكننا تعيين عنوان متغير float إلى مؤشر int ، لكن هنالك نوع من المؤشرات يمكننا أن تشير إلى أي نوع من البيانات وتسمى مؤشرات إلى void ويتم تعريفها كالتالي:-

```
void * ptr;
```

لهذا النوع من المؤشرات استعمالات خاصة فهو يستخدم مثلاً لتمرير المؤشرات إلى دالات تعمل على عدة أنواع بيانات.

المثال التالي يبين أنه إذا لم يتم استعمال مؤشرات إلى void يجب أن نعین للمؤشر عنواناً من نفس نوعها:

```
#include<iostream.h>
void main( )
int intvar;
float flovar;
int* ptrint;
void* ptrvoid;
ptr* ptrflovar;
ptrint=&intvar;
// ptr int = &flovar; //Error
// ptr flo = &intvar; //Error
```

```
ptrvoid=&intvar;
ptrvoid=&flovar;
}
```

في المثال السابق يمكن تعيين عنوان المتغير intvar إلى المؤشر ptr int لأنهما من النوع int\* لكن لا يمكننا تعيين عنوان المتغير flovar إلى المؤشر ptrint لأن الأول من النوع float\* والثاني من النوع int\* . لكن يمكن تعيين أي نوع مؤشرات إلى المؤشر ptrvoid لأنه مؤشر إلى void.

## المؤشرات في استدعاء الدوال

هنالك ثلاث طرق لتميرير الوسائط للدوال :-

- ١ - التميرير بالقيمة `call-by-value` .
- ٢ - التميرير بالمرجع `call-by-reference` .
- ٣ - التميرير بالمرجع مع مؤشر `call by reference with pointer` .arguments

كما ذكرنا سابقاً أن العبارة `return` تستعمل لإعادة قيمة من دالة مستدعاة ورأينا أيضاً أنه يمكن تميرير الوسائط للدوال بالمرجع حتى يتسنى للدالة التعديل في البيانات الأصلية للوسائط، يستخدم مبرمجو `C++` المؤشرات لمحاكاة استدعاء الدوال بالمرجع . عند استدعاء الدالة يتم تميرير عنوان الوسيطة ويتم ذلك بكتابة عامل العنوان للوسيطه المطلوب معالجتها . عندما يتم تميرير عنوان الوسيطة للدالة يتم استعمال العامل \* للوصول لقيمة المتغير .

البرنامجان أدناه يحتويان على إصدارين من دالة تقوم بتكعيب عدد صحيح.

```
// Cube a variable using call-by-value
#include<iostream.h>
int cubeByValue(int); // prototype
int main( )
{
int number = 5;
cout <<" The original value of number is "
<<number<<endl;
number = cubeByValue(number);
cout << " The new value of number is " << number<< endl;
return 0;
}
int cubeByValue(int n)
```

```
{
return n*n*n; // cube local variable n
}
```

ومخرجاته هي

```
The original value of number is 5
The new value of number is 125
```

يقوم هذا البرنامج بتمرير المتغير كوسيلة للدالة مستخدماً طريقة التمرير بالقيمة حيث تقوم الدالة `cubebyvalue` بتكعيب المتغير `number` وتقوم بإرجاع النتيجة للدالة `main` باستخدام العبارة `return` .

في البرنامج التالي يتم تمرير عنوان المتغير `number` كوسيلة للدالة `cube by reference` حيث تقوم الدالة بتكعيب القيمة التي يشير إلى المؤشر `nptr` .

```
// cube a variable using call-by-reference with a
// pointer argument
#include<iostream.h>
void cubeByReference (int *); // prototype
main( )
{
    int number = 5;
    cout<< " The original value of number is " << number
        <<endl;
    cubeByReference(&number);
    cout<< " The new value of number is " << number <<endl;
    return 0;
}
void cubeByReference (int *nPtr)
{
    *nPtr = *nPtr * *nPtr * *nPtr;    // cube number in
    main
}
```

```
The original value of number is 5  
The new value of number is 125
```

نذكر هنا أن الدالة التي يتم تمرير عنوان متغير كوسيط لها يجب أن يتم فيها تعريف مؤشر يحمل قيمة العنوان ، فمثلاً في الدالة `cubeByReference` :-

```
void cubeByReference (int *nptr)
```

المصرح في الدالة `cubeByReference` يشير إلى أنه سيتم تمرير عنوان المتغير من النوع `integer` كوسيط لها ويتم تخزين العنوان في المؤشر `nptr` وهي لا ترجع قيمة للدالة `.main`.

وكما ذكرنا سابقاً أنه في الإعلان عن الدالة يكفي فقط ذكر نوع المتغير الذي سيتم تمريره كوسيط للدالة دون ذكر اسم المتغير ثم الإعلان عن الدالة `cube by reference` كالآتي :-

```
void cubeByReference (int *)
```

## المؤشرات والمصفوفات

عرفنا سابقاً كيف يمكن الوصول إلى العناصر المخزنة في المصفوفات باستخدام اسم المصفوفة وفهرس العنصر. المثال التالي يوضح هذا:

```
int array1[3]={1,2,3};
for (int j=0;j<3;j++)
cout<<endl<<array1[j];
```

يعرض الجزء السابق عناصر المصفوفة `array1` كالتالي:

```
1
2
3
```

يمكن الوصول إلى عناصر المصفوفات أيضاً باستخدام المؤشرات. المثال التالي يوضح كيف يمكن الوصول إلى عناصر نفس المصفوفة السابقة باستخدام

المؤشرات:

```
int array1[3]={1,2,3};
for (int j=0;j<3;j++)
cout<<endl<< *(array1+j);
```

أيضاً يعرض هذا الجزء:

```
1
2
3
```

التعبير `*(array1+j)` له نفس تأثير التعبير `array1[j]` وذلك للآتي:

افرض أن  $j=1$  لذا يكون التعبير `*(array1+j)` مرادفاً للتعبير `*(array1+1)` ويمثل هذا محتويات العنصر الثاني في المصفوفة `array1` وإن اسم المصفوفة يمثل عنوانها وهو عنوان أول عنصر في المصفوفة، ولهذا فالتعبير `array+1` يعني عنوان العنصر الثاني في المصفوفة و `array1+2` يعني عنوان العنصر الثالث في المصفوفة، ولكننا نريد طباعة قيم عناصر المصفوفة `array` وليس عناوينها، لهذا استعملنا عامل المواربة للوصول إلى قيم عناصر المصفوفة.

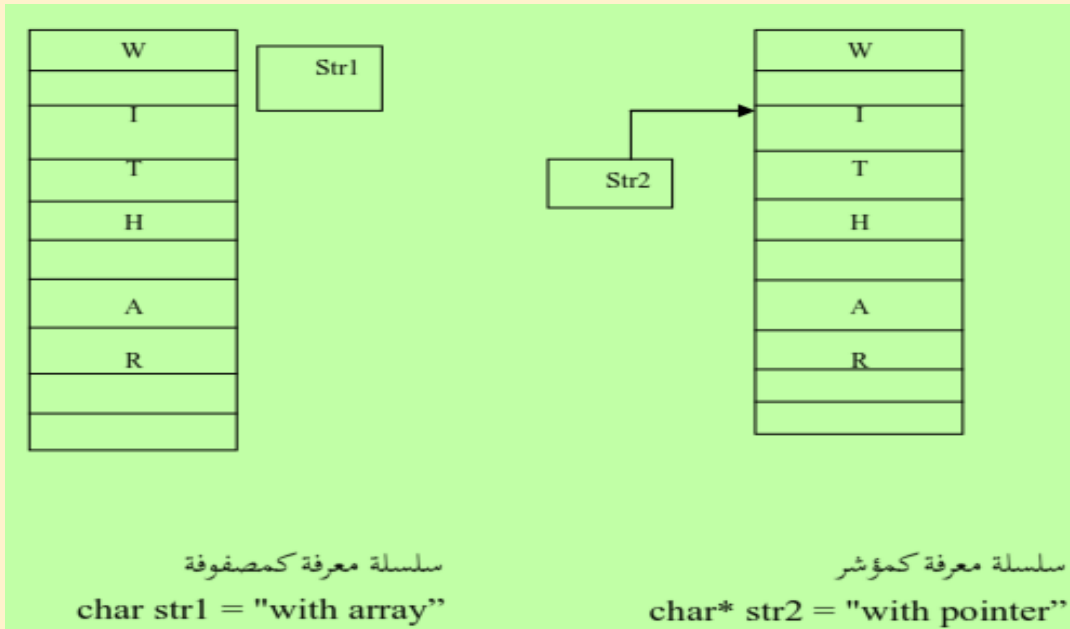
## المؤشرات والسلاسل

كما ذكرنا سابقاً السلاسل هي مجرد مصفوفات من النوع `char` لذا يمكننا استخدام المؤشرات مع أحرف السلاسل مثلما يمكن استخدامه على عناصر أي مصفوفة. المثال التالي يتم فيه تعريف سلسلتين واحدة باستعمال المصفوفات كما في أمثلة السلاسل السابقة والأخرى باستعمال المؤشرات:

```
char str1[ ]="with array";
char str2[ ]="with pointer";
cout <<endl<<str1;
cout <<endl<<str2;
str2++;
cout <<endl<<str2;
```

تشابه السلسلتان السابقتان في عدة نواحي إلا أن هنالك فرق مهم : `str1` هو عنوان أي ثابت مؤشر بينما `str2` هو متغير مؤشر.

والشكل التالي يوضح كيف يبدو هذان النوعان من الذاكرة



لذا يمكننا زيادة `str2` لأنه مؤشر ولكن بزيادته سيشير إلى الحرف الثاني في السلسلة  
وعليه الخرج من المثال السابق:-

with array  
with pointer  
ith pointer

### استعمال العوامل الحسابية مع المؤشرات

يمكن أن تكون المؤشرات معاملات في التعبيرات الحسابية وفي تعابير التعيين والتعابير  
العلائقية . سنتطرق هنا للعوامل التي يمكن أن تكون المؤشرات معاملات لها وكيفية استعمال  
هذه العوامل مع المؤشرات .  
يمكن استعمال `(++)` أو `(--)` لزيادة أو نقصان المؤشرات بمقدار واحد كما يمكن أيضاً إضافة  
متغير صحيح للمؤشر عن طريق استعمال العامل `(+)` أو العامل `(+=)` ويمكن نقصان متغير  
صحيح من مؤشر عن طريق استعمال `(-)` أو `(-=)` كما يمكن أيضاً نقصان أو زيادة مؤشر  
لمؤشر آخر .

افتراض أنه تم الإعلان عن مصفوفة `int v[10]` ، يحمل العنصر الأول في المصفوفة  
العنوان 3000 في الذاكرة.

افتراض أيضاً أنه تم تمهيد مؤشر `vptr` ليشير للعنصر الأول في المصفوفة `v[0]`

وعليه قيمة المؤشر `vptr` هي 3000



يمكن تمهيد المؤشر `vptr` ليشير لمصفوفة `v` بإحدى العبارتين التاليتين:

```
vptr = v;
```

```
vptr = &v[0];
```

عنوان العنصر `v[0]` في المصفوفة `v` هو 3000 وعنوان العنصر `v[1]` هو 3004 وذلك لأن عناصر المصفوفة `v` هي عبارة عن متغيرات صحيحة `integer` واستخدام `4bytes` تمثل من الذاكرة، وعليه عند إضافة أو طرح متغير صحيح `integer` من مؤشر تتم إضافة المتغير مضروباً في حجم المتغير في الذاكرة والثاني يعتمد على نوع المتغير حيث يحتل المتغير الصحيح كما ذكرنا `4bytes` والمتغير الحرفي `char` يحتل `1byte` وعموماً يعتمد ذلك على عدد `bytes` التالي يحتلها المتغير ، فمثلاً العبارة التالية :

```
vptr +=2;
```

تؤدي لإضافة 8 للمؤشر `vptr` بافتراض أن المتغير الصحيح يحتل `4bytes` من

الذاكرة.

### إدارة الذاكرة باستعمال العوامل `new` و `delete`:-

تستعمل المصفوفة لتخزين عدد من الكائنات أو المتغيرات فالعبارة:

```
int ar1[50];
```

تحجز الذاكرة ل 50 عدد صحيح فالمصفوفات هي أسلوب مفيد لتخزين البيانات لكن لها عائق مهم : علينا معرفة حجم المصفوفة في وقت كتابة البرنامج . في معظم الحالات قد لا نعرف كمية الذاكرة التالي سنحتاج إلي أثناء تشغيل البرنامج.  
تزود `C++` أسلوباً خاصاً للحصول على كتل من الذاكرة :

### العامل `new`:-

يخصص العامل `new` كتل ذاكرة ذات حجم معين ويعيد مؤشراً لنقطة بداية كتلة الذاكرة تلك، يحصل العامل `new` على الذاكرة ديناميكياً أثناء تشغيل البرنامج .  
الصورة العامة لكتابة العامل `new` هي:

```
p-var = new type;
```

حيث:-

`p-var`: متغير مؤشر يتم فيه تخزين عنوان بداية كتلة الذاكرة المخصصة بواسطة

العامل `new` تسمح بتخزين متغير من النوع `type` .

### العامل `delete`:-



إذا تم حجز العديد من كتل الذاكرة بواسطة العامل **new** سيتم في النهاية حجز كل الذاكرة المتوفرة وسيتوقف الحاسوب عن العمل . لضمان استعمال آمن وفعال للذاكرة يرافق العامل **new** عامل يسمى **delete** يعيد تحرير الذاكرة لنظام التشغيل .  
الجزء من البرنامج التالي يبين كيف يتم الحصول على ذاكرة لسلسلة :

```
char * str=" It is the best.";
int len = strlen(str);
char*ptr;
ptr= new char[len+1];
strcpy(ptr,str);
cout<<"ptr="<<ptr;
delete [ ] ptr ;
```

تم استعمال الكلمة الأساسية **new** يليها نوع المتغيرات التي سيتم تخصيصها وعدد تلك المتغيرات ، يقوم المثال بتخصيص متغيرات من النوع **char** ويحتاج إلى **len+1** منها حيث تساوي **len** طول السلسلة **str** ، الرقم **1** ينشئ بايتاً إضافياً للحرف الخامد الذي ينهي السلسلة ويعيد العامل **new** مؤشراً يشير إلى بداية قطعة الذاكرة التي تم تخصيصها. تم استعمال المعقفات للدلالة على أننا نخصص ذاكرة لمصفوفة .

```
ptr =new char[len+1];
```

العبارة:

```
delete [ ] ptr;
```

تعيد للنظام كمية الذاكرة التي يشير إليها المؤشر **ptr**.

المعقفات [ ] التي تلي العامل **delete** تشير لأننا نقوم بحذف مصفوفة، لا نحتاج

لاستعمالها إذا كنا نقوم بحذف متغير واحد بواسطة العامل **delete**.

### المؤشر This:

يتملك كل كائن في فئة مؤشراً خاصاً يسمى **this** يشير إليه، وباستخدام هذا المؤشر

يستطيع أي عضو دالي في الفئة معرفة عنوان الكائن الذي استدعاه .

المثال التالي يوضح هذا :-

```
#include<iostream.h>
```

```
class where
```

```
{
```

```
    private:
```

```
        char chararray[10];
```

```
    public:
```

```
    //Continued
```

```
        void reveal( )
```

```
{ cout <<"My Objects address is "<<this;
```

```
};
```

```
main( )
```

```
{
```

```
    where w1,w2;
```

```
    w1.reveal( );
```

```
    w2.reveal( );
```

```
}
```

ينشئ هذا البرنامج كائنات من النوع `where`، ويطلب من كل منها عرض عنوانه باستعمال الدالة `( ) reveal`، والتي تعرض قيمة المؤشر `.this`.

وتكون المخرجات

```
My object's address is 0x8f4effec  
My object's address us 0x8f4effe2
```

نلاحظ إن عنوان الكائن `w2` يتعد `10 Bytes` عن عنوان `w1`، وذلك لأن البيانات في كل كائن تتألف من مصفوفة من `10 Bytes`.  
يمكن معاملة المؤشر `this` كأبي مؤشر كائنات آخر، لذا يمكن استخدامه للوصول إلى بيانات الكائن الذي يشير إليه كما هو مبين في البرنامج أدناه.

```
#include<iostream.h>
class test {
public:
    test(int=0);
    void print( ) const;
private:
    int x;
};
void test::print( ) const
//Continued
{
    cout <<" X="<<x<<endl
    <<"this-> x= "<<this->x<<endl;
        <<"(*this).x="<<(*this).x<<endl;
    }
main ( )
{
    test a(12);
    a.print( );
    return 0;
}
```

وللتوضيح فإن العضو الدالي `print` يقوم أولاً بطباعة `x` مباشرة، ثم يستعمل طريقتين

للوصول إلى `x` باستعمال المؤشر `this`:-

الأولى: باستعمال العامل `(->)`.

الثانية: باستعمال العامل `(.)`.

لاحظ الأقواس التي تحيط بـ `*this`، عندما نقوم باستخدام العامل `(.)` للوصول إلى

أعضاء الفئة نستعمل الأقواس، وذلك لأن العامل `(.)` له أولوية أعلى من العامل `*`، وعليه بدون

الأقواس يتم تقييم التعبير `*this.x` كآتي:

`*(this.x)`

والذي ينتج عرض رسالة خطأ من المصرف لأن العامل `(.)` لا يستخدم مع المؤشرات.

هنالك استعمالات أخرى للمؤشر `this` سنتطرق لها عند تحميلنا للعوامل بشكل زائد.

مثال ١ باستخدام مصفوفة ذات N من الاعداد اكتب برنامج لطباعة الارقام الزوجية بشكل عمودي بواسطة حلقة التكرار ( Do-While ) .

• Example ( 25 )

• مثال ( 25 )

```
#include <iostream.h> _____ ( 1 )
#include <conio.h> _____ ( 2 )
main() _____ ( 3 )
{ _____ ( 4 )
Clrscr ( ) ; _____ ( 5 )
int i = 0 , a[ 10 ] ; _____ ( 6 )
Do _____ ( 7 )
{ _____ ( 8 )
Cin >> a[ i ] ; _____ ( 9 )
IF ( a[ i ] % 2 == 0 ) _____ ( 10 )
Cout << a [ i ] << " " << endl ; _____ ( 11 )
} _____ ( 12 )
While ( i < 10 ) ; _____ ( 13 )
} _____ ( 14 )
```

- ( 1 ) دالة مكتبية تفسر ايعازات الادخال و الاخراج للبرنامج .
- ( 2 ) دالة مكتبية تفسر ايعاز مسح الشاشة ( Clrscr ) و وجودها او عدم وجودها لا يؤثر على نتائج البرنامج فهي لمسح مخلفات تنفيذ سابق .
- ( 3 ) البرنامج الرئيسي .
- ( 4 ) بداية البرنامج الرئيسي .
- ( 5 ) ايعاز مسح الشاشة .
- ( 6 ) وصف المتغيرات .
- ( 7 ) و هو الابعاز الذي سينفذ الابعازات المتواجدة ما بين الاقواس .
- ( 8 ) هذا القوس سيكون بداية الحلقة و كل ما موجود بداخله سيتكرر لاحقا عن طريق الـ ( While ) في الخطوة رقم ( 13 ) .
- ( 9 ) هنا سيتم ادخال البيانات الى المصفوفة و لتكن الارقام من واحد الى عشرة .
- ( 10 ) شرط الطباعة : و هو اذا كانت اعداد المصفوفة تقبل القسمة على ( 2 ) بدون باقي ..... .
- ( 11 ) اطبع الارقام التي تقبل القسمة على ( 2 ) بدون باقي .
- ( 12 ) قوس نهاية الابعازات التي ستكرر .
- ( 13 ) حلقة التكرار : و التي سوف تكرر الاعداد ابتداءً من الصفر و تنتهي بشرط التوقف .
- ( 14 ) نهاية البرنامج الرئيسي .



## الفصل الخامس

### الدوال

### Functions

الدالة :عبارة عن برنامج فرعى (كيان مستقل) استدعائه ذاتى، يحتوى على مجموعة من الإيعازات والتي تقوم بوظيفة معينة حيث يقوم المبرمج باستدعائها في البرنامج الرئيسي لكى تؤدي وظيفتها داخله.

وتمثل الدوال جزء من البرنامج تم تصميمه بغرض الاختصار في كتابة الأكواد الخاصة باللغة ومن أهم فوائدها:

1 - تختصر البرنامج أي يكفي استدعائها بواسطة كتابة اسمها في البرنامج الرئيسي لكى تقوم بالعمل المطلوب منها

2 - تجنب المبرمج التكرار في كتابة الإيعازات

3 - توفير مساحة تخزينية في الذاكرة

4 - تختصر زمن البرمجة في حالة تواجد دوال جاهزة

5 - البرنامج المكتوبة بواسطة الدوال تكون معالجته للبيانات المدخلة اليه اسرع عند التنفيذ

وبأستخدام الدوال يصبح البرنامج اكثر وضوحا حيث يأخذ البرنامج الشكل التركيبى فيمكن أن

يصبح بالشكل التالى:

```
#include<iostream.h>
```

```
int , float , char , string variables ;
```

```
Function Name ( )
```

```
{
```

```
the instruction of function
```

```
}  
main ( )  
{  
Function Name ( );  
}
```

ويتم توضيح ما سبق كما يلي:

السطر الثاني وصف المتغيرات  
السطر الثالث أسم الدالة المسمى من قبل المبرمج ويفضل ان يدل على عمل الدالة  
السطر الرابع قوس بداية الدالة  
السطر الخامس إيعازات الدالة أو الكود الذي سيكتب لكي ينفذ عند الأستدعاء  
السطر السادس قوس نهاية الدالة  
السطر السابع البرنامج الرئيسي  
السطر التاسع عند كتابة أسم الدالة يمكن للبرنامج الرئيسي استدعاء الدالة للتنفيذ أي أستدعى الدالة  
للقيام بالوظيفة الخاصة بها والمبرمج على أساسها

ورثت اللغة ++C من اللغة C مكتبة ضخمة وغنية بدوال تقوم بتنفيذ العمليات الرياضية، التعامل مع السلاسل والأحرف، الإدخال والإخراج، اكتشاف الأخطاء والعديد من العمليات الأخرى المفيدة مما يسهل مهمة المبرمج الذي يجد في هذه الدوال معيناً كبيراً له في عملية البرمجة.  
يمكن للمبرمج كتابة دوال تقوم بأداء عمليات يحتاج لها في برامجه وتسمى مثل هذه الدوال  
Programmer- defined functions



ومن أهم فوائدها:

- 1/ تساعد الدوال المخزنة في ذاكرة الحاسب على اختصار البرنامج إذ يكفي باستدعائها باسمها فقط لتقوم بالعمل المطلوب .
- 2/ تساعد البرامج المخزنة في ذاكرة الحاسب أو التي يكتبها المستخدم على تلافي عمليات التكرار في خطوات البرنامج التي تتطلب عملاً مشابهاً لعمل تلك الدوال.
- 3/ تساعد الدوال الجاهزة في تسهيل عملية البرمجة.
- 4/ يوفر استعمال الدوال من المساحات المستخدمة في الذاكرة.
- 5/ كتابة برنامج الـ C++ في شكل دوال واضحة المعالم يجعل البرنامج واضحاً لكل من المبرمج والقارئ على حد سواء.

وفيما يلي شرح لأحد الدوال الهمة بالبرنامج

تحتوي مكتبة الدوال الرياضية على العديد من الدوال التي تستخدم في تنفيذ العمليات الرياضية الحسابية. فمثلاً المبرمج الذي يرغب في حساب وطباعة الجذر التربيعي للعدد 900 قد يكتب عبارة كالتالية:

```
cout << sqrt ( 900);
```

عند تنفيذ هذه العبارة يتم استدعاء الدالة المكتبية `sqrt` لحساب الجذر التربيعي للعدد بين القوسين (900). يسمى العدد بين القوسين وسيطة الدالة `argument` وعليه فالعبارة السابقة تقوم بطباعة العدد 30 ، تأخذ الدالة `sqrt` وسيطة من النوع `double` وتكون النتيجة قيمة من نفس النوع وينطبق هذا على جميع الدوال الرياضية.

عند استعمال الدوال الرياضية في أي برنامج بلغة `C++` يجب تضمين الملف `math.h` والذي يحتوي على هذه الدوال.

الجدول التالي يلخص بعض الدوال الرياضية:

Function	Description	Example
<code>sqrt(x)</code>	الجذر التربيعي لـ $x$	<code>sqrt(9.0) is 3</code>
<code>exp(x)</code>	<code>exp(1.0) is 2.718282</code>	$e^x$
<code>fabs(x)</code>	القيمة المطلقة لـ $x$	if $x > 0$ <code>fabs(x) = x</code> if $x = 0$ <code>fabs(x) = 0</code> if $x < 0$ <code>fabs(x) = -x</code>
<code>ceil(x)</code>	تقرب $x$ لأصغر عدد صحيح أكبر من $x$	<code>ceil(9.2) is 10.0</code> <code>ceil(-9.8) is 9.0</code>
<code>floor(x)</code>	تقرب $x$ لأكبر عدد صحيح أصغر من $x$	<code>floor(9.2) is 9</code> <code>floor(-9.8) is -10.0</code>

الدوال تمكن المبرمج من تقسيم البرنامج إلى وحدات **modules**، كل دالة في البرنامج تمثل وحدة قائمة بذاتها، ولذا نجد أن المتغيرات المعرفة في الدالة تكون متغيرات محلية (**Local**) ونعني بذلك أن المتغيرات تكون معروفة فقط داخل الدالة. أغلب الدوال تمتلك لائحة من الوسائط (**Parameters**) والتي هي أيضاً متغيرات محلية.

هنالك عدة أسباب دعت إلى تقسيم البرنامج إلى دالات وتسمى هذه العملية

(**Functionalizing a program**) وهي:

- 1/ تساعد الدوال المخزنة في ذاكرة الحاسب على اختصار البرنامج إذ يكفي باستدعائها باسمها فقط لتقوم بالعمل المطلوب .
- 2/ تساعد البرامج المخزنة في ذاكرة الحاسب أو التي يكتبها المستخدم على تلافى عمليات التكرار في خطوات البرنامج التي تتطلب عملاً مشابهاً لعمل تلك الدوال.
- 3/ تساعد الدوال الجاهزة في تسهيل عملية البرمجة.
- 4/ يوفر استعمال الدوال من المساحات المستخدمة في الذاكرة.
- 5/ كتابة برنامج **C++** في شكل دوال واضحة المعالم يجعل البرنامج واضحاً لكل من المبرمج والقارئ على حد سواء.

كل البرامج التي رأيناها حتى الآن تحتوي على الدالة **main** وهي التي تنادى الدوال

المكتيبة لتنفيذ مهامها. سنرى الآن كيف يستطيع المبرمج بلغة الـ **C++** كتابة دوال خاصة به.

## نموذج الدالة

عندما يولد المصرف تعليمات لاستدعاء دالة، ما فإنه يحتاج إلى معرفة اسم الدالة وعدد وسيطاتها وأنواعها ونوع قيمة الإعادة ، لذا علينا كتابة نموذج أو (تصريح) للدالة قبل إجراء أي استدعاء لها وتصريح الدالة هو سطر واحد يبلغ المصرف عن اسم الدالة وعدد وسيطاتها وأنواعها ونوع القيمة المعادة بواسطة الدالة. يشبه تصريح الدالة ، السطر الأول في تعريف الدالة ، لكن تليه فاصلة منقوطة.

فمثلا في تصريح الدالة التالي:-

```
int anyfunc(int);
```

النوع `int` بين القوسين يخبر المصرف بأن الوسيط الذي سيتم تمريره إلى الدالة سيكون

من النوع `int` و `int` التي تسبق اسم الدالة تشير إلى نوع القيمة المعادة بواسطة الدالة.

يأخذ تعريف الدوال في C++ الشكل العام التالي:

```
return-value-type function-name (parameter list)
{
    declarations and statements
}
```

حيث:

**return-value-type**: نوع القيمة المعادة بواسطة الدالة والذي يمكن أن يكون أي نوع من أنواع بيانات C++. وإذا كانت الدالة لا ترجع أي قيمة يكون نوع إعادتها **void**.

**function-name**: اسم الدالة والذي يتبع في تسميته قواعد تسمية المعرفات (identifiers).

**parameter list**: هي لائحة الوسيطات الممرة إلى الدالة وهي يمكن أن تكون خالية (void) أو تحتوي على وسيطة واحدة أو عدة وسائط تفصل بينها فاصلة ويجب ذكر كل وسيطة على حدة.

**declarations and statements**: تمثل جسم الدالة والذي يطلق عليه في بعض الأحيان **block**. يمكن أن يحتوي الـ **block** على إعلانات المتغيرات ولكن تحت أي ظرف لا يمكن أن يتم تعريف دالة داخل جسم دالة أخرى. السطر الأول في تعريف الدالة يدعى **declarator** والذي يحدد اسم الدالة ونوع البيانات التي تعيدها الدالة وأسماء وأنواع وسيطاتها.

## استدعاء الدالة

يتم استدعاء الدالة (التسبب بتنفيذها من جزء آخر من البرنامج، العبارة التي تفعل ذلك هي استدعاء الدالة) يؤدي استدعاء الدالة إلى انتقال التنفيذ إلى بداية الدالة. يمكن تمرير بعض الوسيطات إلى الدالة عند استدعائها وبعد تنفيذ الدالة يعود التنفيذ للعبارة التي تلي استدعاء الدالة.

بإمكان الدالة أن تعيد قيم إلى العبارة التي استدعتها. ويجب أن يسبق اسم الدالة في معرفتها وإذا كانت الدالة لا تعيد شيئاً يجب استعمال الكلمة الأساسية `void` كنوع إعادة لها للإشارة إلى ذلك .

هنالك ثلاث طرق يمكن بها إرجاع التحكم إلى النقطة التي تم فيها استدعاء الدالة:

١ / إذا كانت الدالة لا ترجع قيمة يرجع التحكم تلقائياً عند الوصول إلى نهاية الدالة

٢ / باستخدام العبارة `return;`

٣ / إذا كانت الدالة ترجع قيمة فالعبارة `return expression;` تقوم بإرجاع قيمة

التعبير `expression` إلى النقطة التي استدعتها .

خذ برنامجاً يستخدم دالة تدعى `square` لحساب مربعات الأعداد من 1 إلى 10.

```

#include<iostream.h>
int square(int); //function prototype
main()
{
    for(int x=1;x<=10;x++)
    cout<<square(x)<<" ";
    cout<<endl;
}
//now function definition
int square(int y)
{
    return y*y;
}

```

وتكون المخرجات

1 4 9 16 25 36 49 64 81 100

يتم استدعاء الدالة `square` داخل الدالة `main` وذلك بكتابة `square(x)`. تقوم الدالة `square` بنسخ قيمة `x` في الوسيط `y`. ثم تقوم بحساب `y*y` ويتم إرجاع النتيجة إلى الدالة `main` مكان استدعاء الدالة `square`، حيث يتم عرض النتيجة وتكرر هذه العملية عشر مرات باستخدام حلقة التكرار `for`.

تعريف الدالة `square()` يدل على أنها تتوقع وسيطة من النوع `int` و `int` التي تسبق اسم الدالة تدل على أن القيمة المعادة من الدالة `square` هي من النوع `int` أيضاً. العبارة `return` تقوم بإرجاع ناتج الدالة إلى الدالة `main`.  
السطر:

`int square (int)`

هو نموذج أو تصريح الدالة (function prototype).



الوسيطات هي الآلية المستخدمة لتمرير المعلومات من استدعاء الدالة إلى الدالة نفسها حيث يتم نسخ البيانات وتخزين القيم في متغيرات منفصلة في الدالة تتم تسمية هذه المتغيرات في تعريف الدالة. فمثلاً في المثال السابق تؤدي العبارة `cout<< square(a);` في `main()` إلى نسخ القيمة `a` إلى البارامتر `y` المعرف في تعريف الدالة.

المصطلح وسيطات `Argument` يعني القيم المحددة في استدعاء الدالة بينما يعني المصطلح بارامترات `parameters` المتغيرات في تعريف الدالة والتي تم نسخ تلك القيم إليها، ولكن غالباً ما يتم استعمال المصطلح وسيطات لقصد المعنيين.

البرنامج التالي يستخدم دالة تدعى `maximum` والتي ترجع العدد الأكبر بين ثلاثة أعداد صحيحة.

يتم تمرير الأعداد كوسائط للدالة التي تحدد الأكبر بينها وترجعه للدالة `main` باستخدام العبارة `return` ويتم تعيين القيمة التي تمت إعادتها إلى المتغير `largest` الذي تتم طباعته.



```

#include <iostream.h>
int maximum (int, int, int);
main( )
{
int a, b, c;
cout << "Enter three integers: " ;
cin >> a >> b >> c ;
cout << " maximum is : " << maximum (a, b, c) << endl;
return 0;
}
int maximum (int x, int y, int z)
{
int max = x;
if (y > x)
max = y;

```

```

if (z > max)
max = z;
//Continued
return max;
}

```

فإذا تم ادخال 17,85,22 تكون المخرجات

```

Enter three integers: 22 85 17
Maximum is: 85

```

في لغة الـ C++ تكتب الدالة التي لا تمتلك وسيطات إما بكتابة `void` بين

القوسين الذين يتبعان اسم الدالة أو تركهما فارغين ، فمثلاً الإعلان

```
void print ( );
```

يشير إلى أن الدالة `print` لا تأخذ أي وسيطات وهي لا ترجع قيمة .

المثال التالي يبين الطريقتين اللتين تكتب بهما الدوال التي لا تأخذ وسيطات:

```
// Functions that take no arguments
#include <iostream.h>
void f1 ( );
```

```
void f2 (void);
//Continued
main( )
{
    f1 ( );
    f2 ( );
return 0;
}

void f1 ( )
{
    cout << "Function f1 takes no arguments" << endl;
}
void f2 (void)
{
    cout << "Function f2 also takes no arguments" <<
endl;
}
```

وتكون المخرجات

Function f1 takes no arguments  
Function f2 also takes no arguments

الدوال السياقية

تحتاج بعض التطبيقات التي يعتبر فيها وقت تنفيذ البرنامج أمراً حيوياً وحاسماً، لإبدال عملية استدعاء واستخدام دالة بما يسمى دالة سياقية . وهي عبارة عن شفرة تقوم بالعمل المطلوب نفسه، يتم تعريف الدالة السياقية باستعمال نفس التركيب النحوي المستخدم لتعريف الدالة الاعتيادية ، لكن بدلاً من وضع شفرة الدالة في مكان مستقل يضعها المصنف في

السياق الطبيعي للبرنامج مكان ظهور استدعاء الدالة. يتم جعل الدالة سياقية عن طريق استخدام الكلمة الأساسية **inline** في تعريف الدالة.

```
inline void func1( )  
{  
statements  
}
```

تستخدم الدالة السياقية فقط إذا كانت الدالة قصيرة وتستخدم مرات عديدة في

البرنامج.

مثال:

```
#include<iostream.h>  
inline float cube(float s){return s*s*s;}  
main()  
{  
cout<<"\nEnter the side length of your cube : ";  
float side;  
cin>>side;  
cout<<"volume of cube is "  
<<cube(side)  
<<endl;  
}
```

وتكون المخرجات من البرنامج .....؟

مثال آخر

```

#include <iostream.h>
inline int mult( int a, int b)
{
return (a*b);
}
//Continued
main( )
{
int x, y, z;
cin >> x >> y >> z;
cout << "x = " << x << " y = " << y << " z = " << z << endl;
cout << "product1" << mult (x ,y) << endl;
cout << "product2" << mult (x +2, y) << endl;
return 0;
}

```

وتكون المخرجات في حالة ادخال

:(x = 3, y =4, z = 5)

```

x = 3    y = 4    z = 5
product1  12
product2  32

```

التمرير بالقيمة والتمرير بالمرجع

لنفرض أننا لدينا متغيرين صحيحين في برنامج ونريد استدعاء دالة تقوم بتبديل قيمتي الرقمين ،لنفرض أننا عرفنا الرقمين كالاتي:

```
int x=1;
int y=2;
```

أ/ التمرير بالقيمة (pass-by-value):-

ترى هل تقوم الدالة التالية بتبديل القيمتين:

```
void swap (int a, int b)
{
```

```
int temp =a;
a=b.
b=temp;
}
```

تقوم هذه الدالة بتبديل قيمتي **a** و **b** ، لكن إذا استدعينا هذه الدالة كالاتي:

```
swap( x,y);
```

سنجد أن قيمتي **x** و **y** لم تتغير وذلك لأن الوسيطات الاعتيادية للدالة يتم تمريرها بالقيمة وتنشئ الدالة متغيرات جديدة كلياً هي **a** و **b** في هذا المثال لتخزين القيم الممررة إليها وهي (1,2) ثم تعمل على تلك المتغيرات الجديدة وعليه عندما تنتهي الدالة ورغم أنها قامت بتغيير **a** إلى 2 و **b** إلى 1 لكن المتغيرات **x** و **y** في استدعاء الدالة لم تتغير.

ب/ التمرير بالمرجع (pass-by-refrence):

التمرير بالمرجع هو طريقة تمكن الدالة ( swap ) من الوصول إلى المتغيرات الأصلية **x**

و **y** والتعامل معها بدلاً من إنشاء متغيرات جديدة . ولإجبار تمرير الوسيطة بالمرجع نضيف الحرف **&** إلى نوع بيانات الوسيطة في تعريف الدالة وتصريح الدالة .

مثال

كيفية كتابة الدالة `swap` وتمييز وسيطاتها بالمرجع:

```
#include <iostream.h>
void swap (int & , int&);
main ( )
{
int x= 1;
int y= 2;
  swap (x, y);
return 0;
}
void swap (int& a, int & b)
{
cout <<"Original value of a is " << a<<endl;
int temp =a;
a=b;
b=temp;
```

```
cout <<"swapped value of a is " << a<<endl;  
}
```

بعد تنفيذ هذه الدالة تتغير قيمة  $x$  إلى 2 و  $y$  إلى 1 . ويكون الخرج من البرنامج كالتالي:

```
Original value of a is 1  
Swapped value of a is 2
```



**الباب الثالث**

**التطبيقات التجارية للبرنامج الإحصائي ساس**

**SAS**

**The Statistical Analysis System**



يعتبر نظام (SAS) أحد برامج التحليل الإحصائي الذي أنتجته شركة ساس والتي تأسست عام

ساس

1976 عن طريق جيمس غودنايت وجون سول، الأساتذة بجامعة ولاية كارولينا الشمالية، لصالح

وزارة

الزراعة الأمريكية، وتعتبر ساس أو SAS اختصاراً لنظام التحليل الإحصائي أو

### Statistical Analysis System

ويتميز نظام ساس بأنه نظام مرن ومفتوح حيث يمكننا إضافة المكونات التي تلئم احتياجاتنا، وهناك العديد من المكونات التي يمكن إضافتها تتعلق بإدخال البيانات و تقارير الرسوم البيانية والتحليل الإحصائي والرياضي المتقدم والتخطيط الإداري والتنبؤ والمساعدة في اتخاذ القرارات.

وفي هذا الجزء سندرس كيفية تحليل البيانات المعطاة بشكل ميسر واستخراج التقارير المتنوعة، وسوف نبين كيفية حساب المقاييس الإحصائية الوصفية المختلفة كالمتوسط الحسابي والتباين والانحراف المعياري، وكذلك حساب معاملات الارتباط والإقتران، وسنبحث عن اختبار الفرضيات على نظام ساس، وغيرها من المواضيع الإحصائية التي تهتم بالدرجة الأولى المتخصصون في مجال الإحصاء وتفيد غير المتخصصين من باحثين في مجالات أخرى كالتطب والصناعة وغيرها.

## الفصل الأول

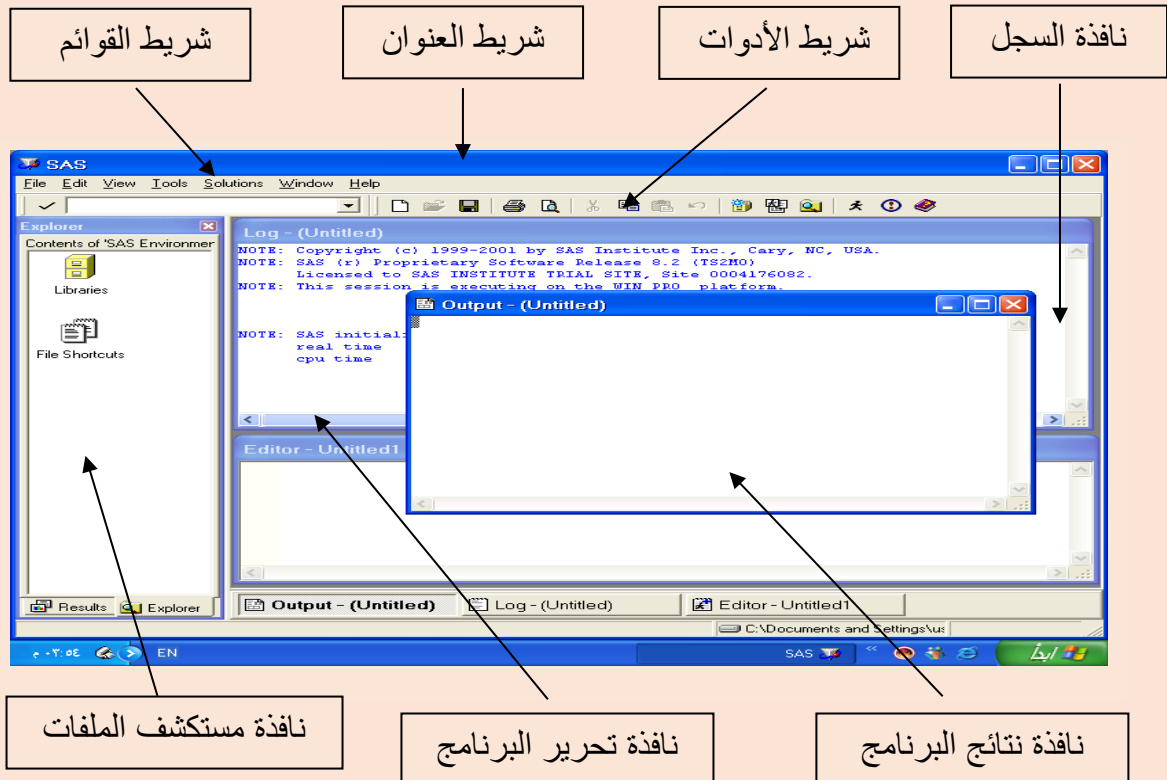
### مكونات ونوافذ حزمة ساس

\* واجهة البرنامج:

من خلالها يتم كتابة البرامج بلغة ساس وتحريرها ويمكن إدارة البيانات والمخرجات، وعليه يمكن استخدام لغة ساس للبرمجة والإجراءات الجاهزة مع واجهة تطبيق مبنية على نوافذ سهلة الاستخدام للوصول إلى البيانات من مصادر متعددة وإدارتها وكذلك لتحليل البيانات وعرضها كمعلومات مفيدة في تقارير على أي نظام تشغيل.

\*\* أقسام واجهة النظام:

والشكل التالي يوضح هذه الأقسام:



ومن الشكل السابق نلاحظ المكونات التالية:

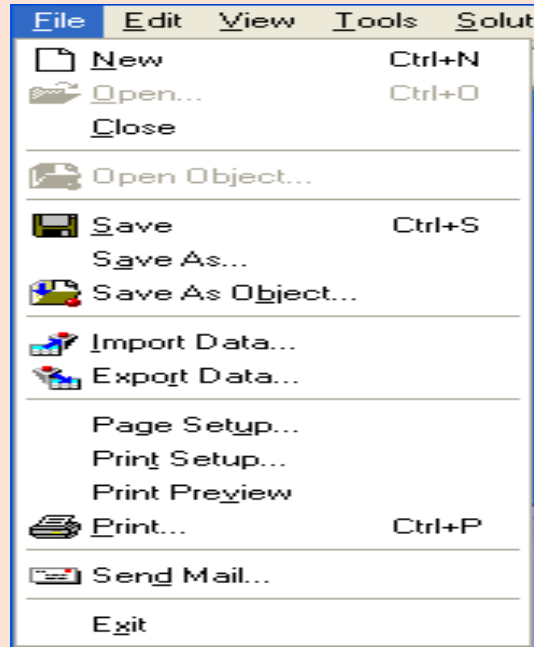
\*\*\* شريط القوائم:

File Edit View Tools Solutions Window Help

من خلال شريط القوائم يمكنك التعامل مع برنامج ساس بشكل ميسر وسهل وسنوضح بعض الخيارات في شريط القوائم كالتالي :

\*\*\*\* قائمة ملف (File):

قائمة ملف هي القائمة الأساسية في البرنامج ، نبدأ منها بإنشاء ملفات جديدة وحفظها واسترجاعها وطباعتها وإرسالها لتطبيقات أخرى للخروج من البرنامج كما تظهر في الشكل التالي:



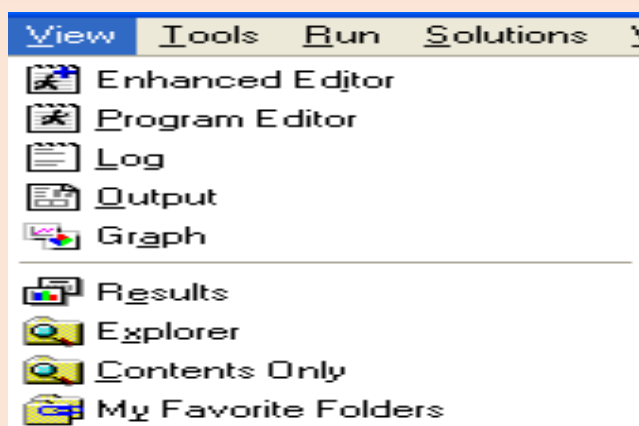
\*\*\*\* قائمة تحرير (Edit)

القائمة التي يمكن من خلالها عمل التصحيحات وإجراء عمليات البحث والاستبدال.



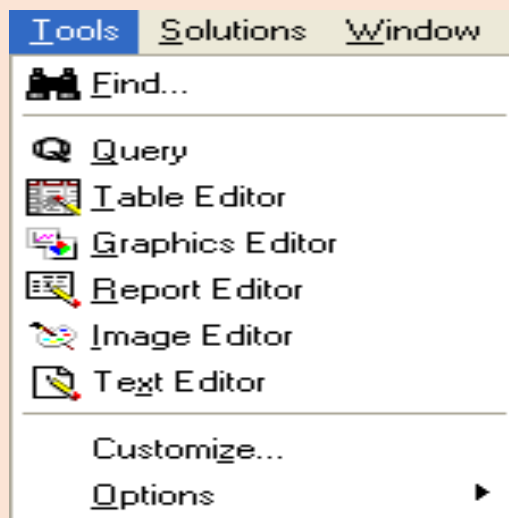
### \*\*\*\* قائمة عرض (View)

هي القائمة التي يمكن من خلالها إضافة محرر برنامج جديد أو نافذة مخرجات جديدة , وتتحكم هذه القائمة بالشكل الظاهر للبرنامج ومنها يمكن الوصول إلى الملفات بشكل سريع.



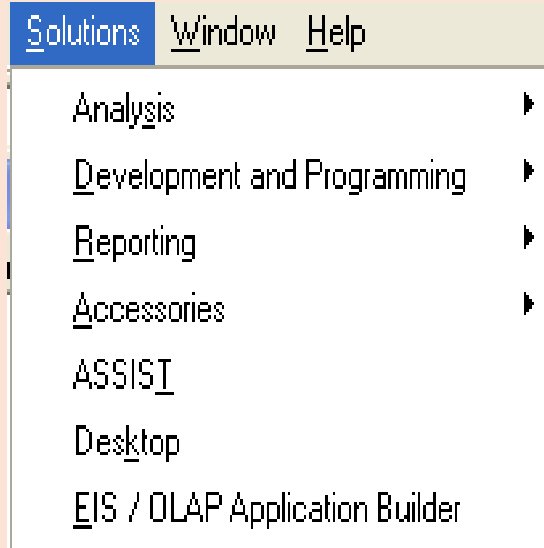
### \*\*\*\* قائمة أدوات (Tools)

قائمة أدوات تحتوي على مجموعة الأدوات المختلفة مثل الرسم وتنسيقه وتصاميم جدولية



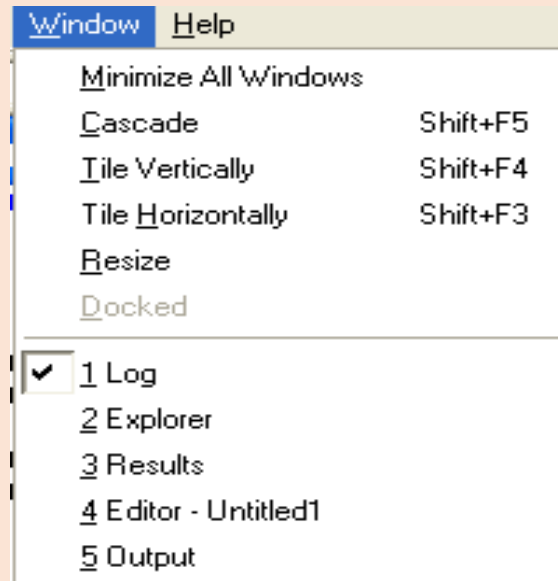
### \*\*\*\* قائمة حلول ( Solution )

تقوم هذه القائمة بجدولة البيانات وتنظيمها وتحليلها إحصائياً.



### \*\*\*\* قائمة نوافذ (Windows)

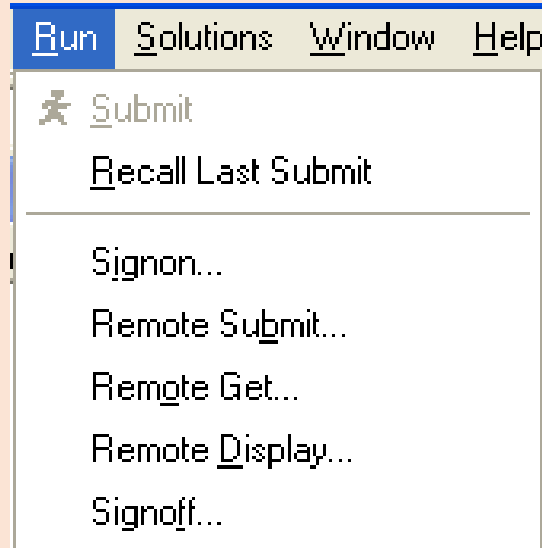
لتشغيل نوافذ البرنامج وتنظيمها .





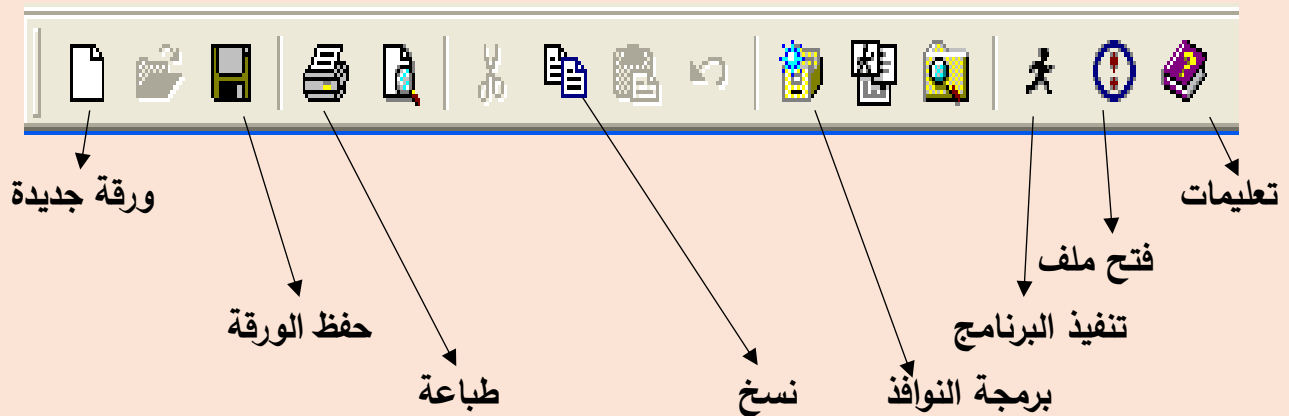
### \*\*\*\* قائمة (Run)

وهي قائمة تظهر عندما تكون نافذة تحرير البرنامج نشطة ( في حالة الاستعداد ) وهي تقوم بتنسيق بيانات البرنامج وجدولتها ثم تحليلها وإظهار نتائج البرنامج .



### \*\*\* شريط الأدوات:

حيث يظهر شريط الأدوات تحت شريط القوائم ويمكنك من خلاله الوصول إلى أوامر البرنامج بشكل مختصر وسريع، وهو ما يوضحه الشكل التالي:



\* نوافذ الحزمة ساس:

\*\* نافذة الأخطاء Log windows

وهي نافذة تقييم البرمجة وتكتشف فيه الأخطاء .

\*\* نافذة تحرير البرنامج Program Editor windows

ومن خلال هذه النافذة نقوم ببرمجة البيانات بلغة ساس

\*\* نافذة النتائج Output windows

وهي تظهر مخرجات أو نتائج البرنامج المنفذ

\*\* نافذة مستكشف الملفات File explorer windows

وهي نافذة تعرض كل ملفات الساس ويمكنك من خلالها إضافة ملفات جديدة وحفظها .

\* إدخال البيانات إلى حزمة ساس

يسمح برنامج بدخول البيانات عامودية أو أفقية أو على شكل مصفوفة، فهناك ثلاثة طرق

لإدخال البيانات إلى برنامج ساس .

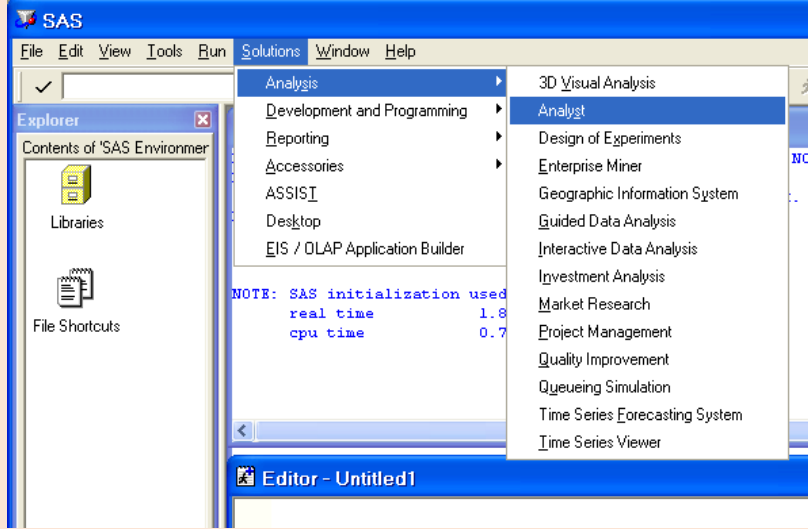
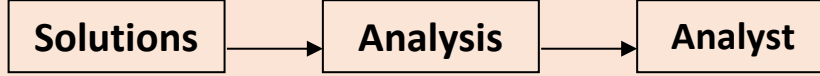
سنأخذ مثال بسيط نبين فيه كيفية إدخال البيانات في البرنامج .

مثال: الجدول التالي يوضح اسعار بيع نوعين من المنتجات X1, X2

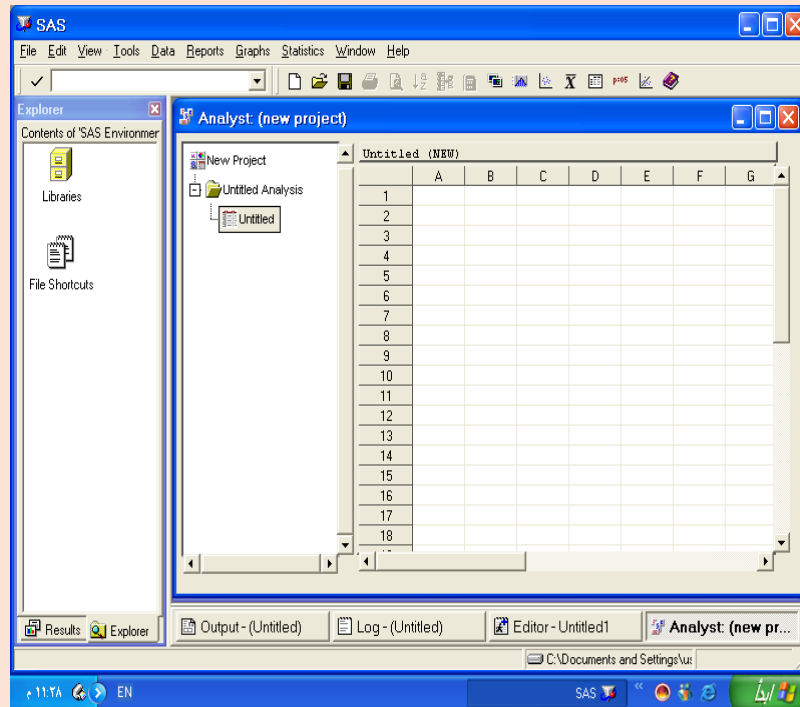
( 11 وحدة من النوعين ) .

3.0	4.4	7.3	3.9	3.9	4.2	4.2	8.4	1.2	3.4	5.0	X1
30	46	32	36	41	35	36	17	12	32	47	X2

الطريقة الأولى: ادخال البيانات مباشرة إلى ورقة عمل ساس والتي على شكل جدول .  
\* نختار المسار التالي من شريط القوائم

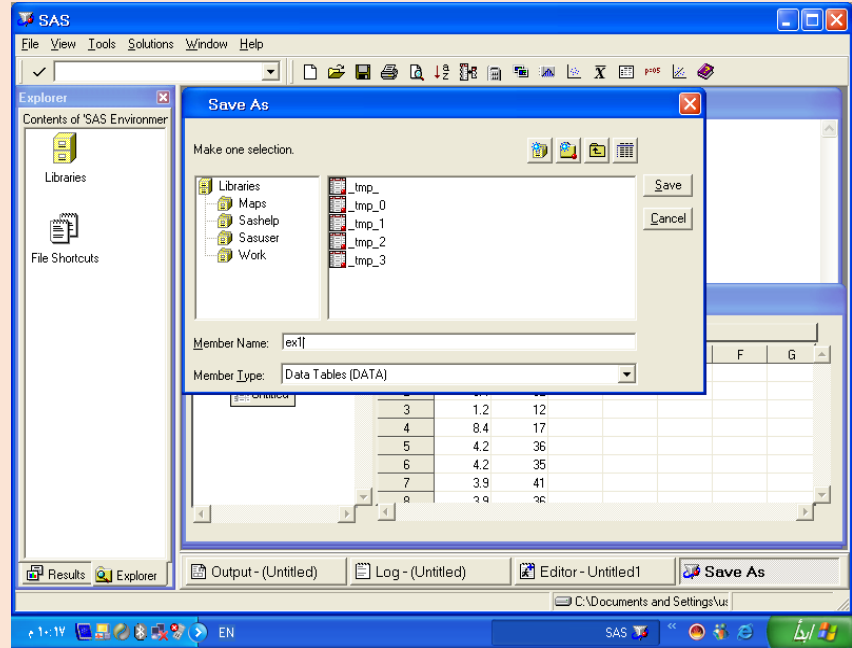


سوف يظهر لنا التالي :



\* ندخل البيانات داخل ورقة العمل في الشاشة السابقة ونختار حفظ البيانات داخل أحد المجلدات

وليكن work وبأى أسم وليكن ex1 كالتالى:

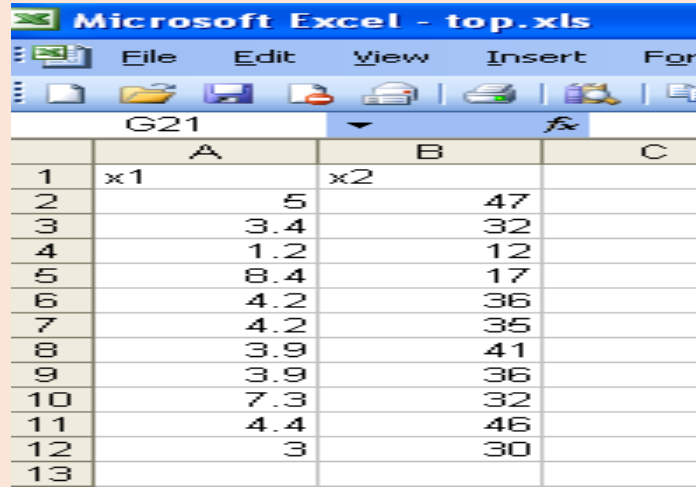


ثم نختار save وبذلك تكون البيانات حفظت بملف أسمه ex1 داخل مجلد Work.

الطريقة الثانية: إذا كانت البيانات موجودة مسبقا Excel فيتم إدخالها داخل برنامج SAS داخل برنامج

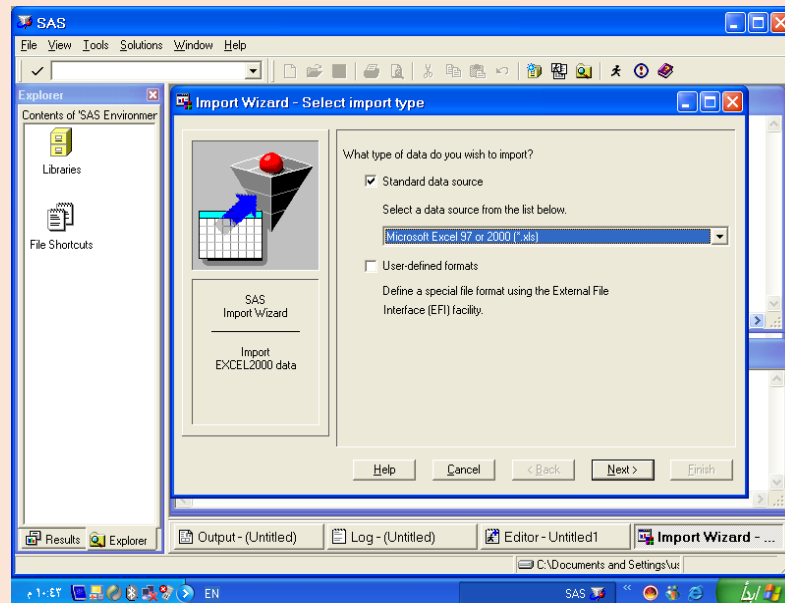
وإليك خطوات هذه الطريقة :

أ - إذا كانت البيانات موجودة في برنامج الاكسيل باسم Top كما يلي:



	A	B	C
1	x1	x2	
2	5	47	
3	3.4	32	
4	1.2	12	
5	8.4	17	
6	4.2	36	
7	4.2	35	
8	3.9	41	
9	3.9	36	
10	7.3	32	
11	4.4	46	
12	3	30	
13			

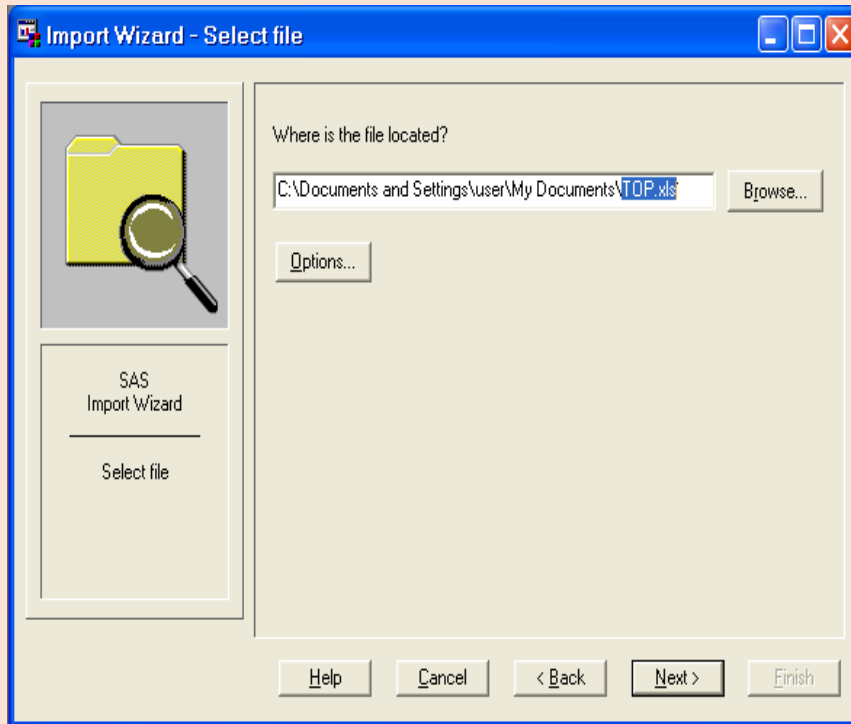
ب - بعد فتح برنامج ساس نختار الأمر **File Import Data** من قائمة **File** فتظهر الشاشة التالية :



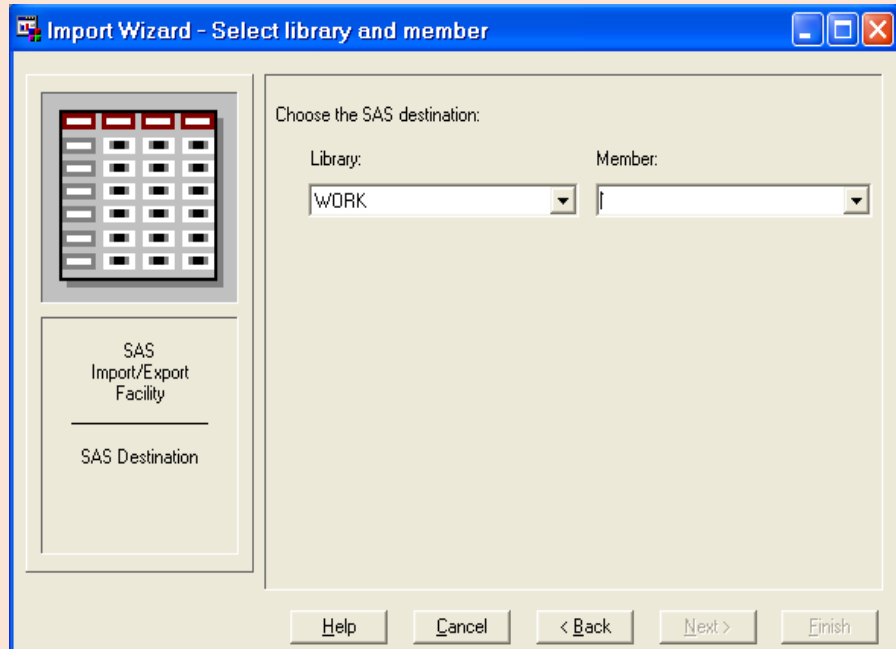
ثم نختار **Next** فتظهر الشاشة التالية



نقوم بتحديد المجلد وأسم الملف الموجود في **Browse** فتظهر الشاشة



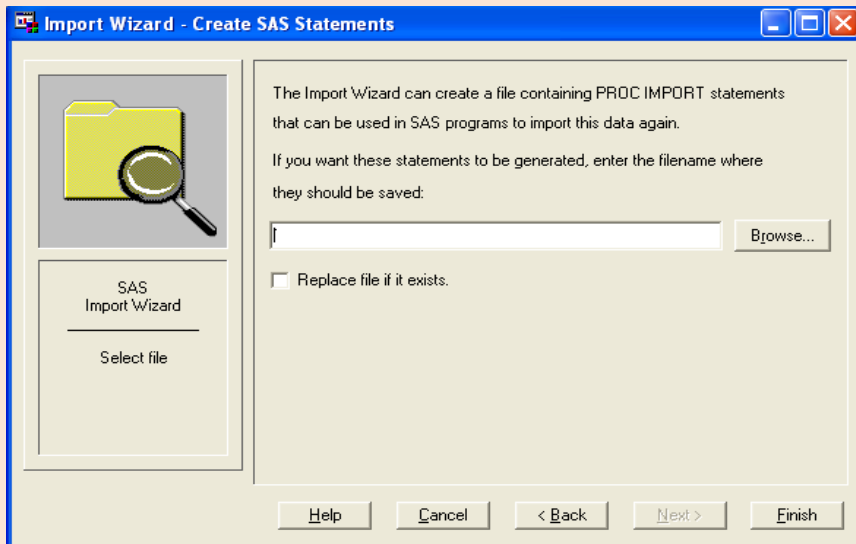
وبالنقر على Next تظهر الشاشة



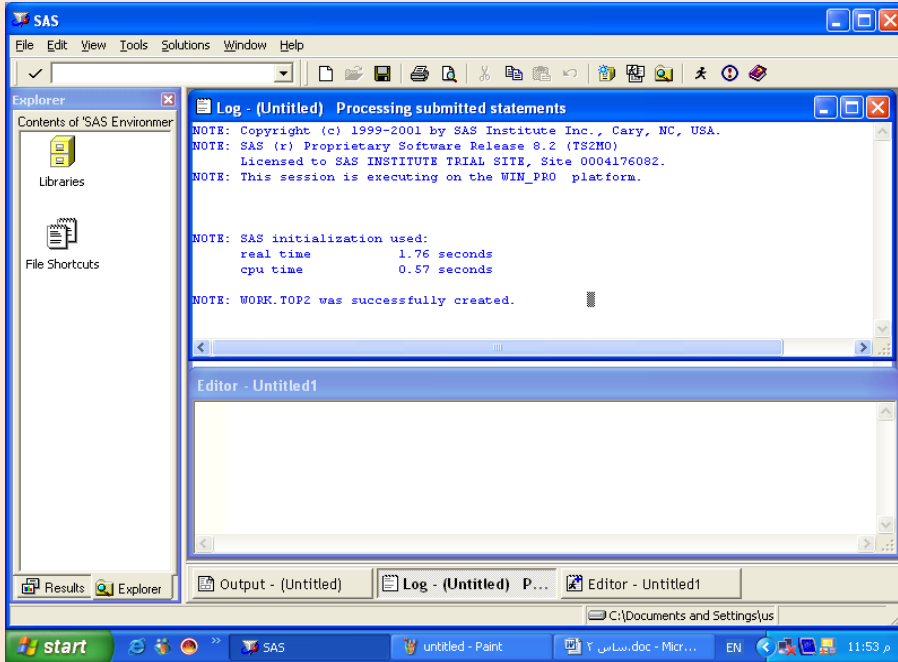
نحدد اسم member (وهو اسم الملف الداخل من اكسيل الى ساس) وليكن Top2

ونختار المكتبة أو المجلد الذي سيكون الملف الجديد بداخله وليكن work

فتظهر الشاشة التالية :



في الشاشة السابقة نقر على **finish** وبذلك أدخلنا البيانات بملف اسمه **Top2** بعد الضغط على **finish** في الشاشة الخيرة يجب أن تكون شاشة البرنامج كما في الشكل



وتبين هذه الشاشة نجاح عملية ادخال البيانات في ملف اسمه **Top2** في مكتبة **Work**

ويمكن التحقق من تحميل الملف بالنقر على **Libraries**


ثم النقر على **work**



## الطريقة الثالثة: إدخال البيانات عن طريق نافذة Editor

حيث يتم الكتابة في هذه النافذة البرمجة التالية :

```
Data ex1 ;  
    Input X1 X2 ;  
Datalines ;  
    5      47  
    3.4    32  
    1.2    12  
    8.4    17  
    4.2    36  
    4.2    35  
    3.9    41  
    3.9    36  
    7.3    32  
    4.4    46  
    3      30  
;  
run;
```

ثم نحدد أو نطلب البرنامج ثم نضغط على  ثم نضغط على Libraries  
ثم Work لنجد الملف Ex1 لنفتحه.  
وبهذا أصبح لدينا ثلاثة طرق لإدخال البيانات إلى برنامج ساس

\* تغيير القيم بعد حفظها :

إذا أردنا تغيير القيم في المثال السابق بعد أن تم حفظها work نقوم بالخطوات التالية:  
في مجلد

(1) نختار المسار التالي من شريط القوائم :

Solutions → Analysis → Analyst

(2) من قائمة File نختار امر Open by sas name

(3) من مجلدات ساس نختار المجلد work ثم الملف Ex1

(4) من القائمة Edit نختار الامر Mode ومنه الامر الفرعي Edit

(5) نذهب الى الخلية أو الخيا المراد تغييرها ونغيرها

(6) ثم File نختار الامر Save ونغلق النافذة المجدولة فقط وليس البرنامج، ستظهر نافذة

من قائمة

نختار منها Do not save .

ولاسترجاع البيانات بعد حفظها :

(1) افتح البرنامج واختر المسار

Solutions → Analysis → Analyst

(2) من قائمة File نختار الأمر open لتظهر لك نافذة الفتح.

(3) اختيار الملف ثم فتح.

\* حفظ (البرنامج) :

بعد كتابة البرنامج الذي يقوم بتحليل البيانات ، وأردت أن تحفظه فاتبع الخطوات التالي:

(1) أن تكون نافذة تحرير البرنامج Editor نشطة

(2) من قائمة file نختار امر Save as ونستكمل.

ولاسترجاع ( البرنامج ):

(1) من قائمة File اختر الأمر Open لتظهر لك نافذة الفتح .

(2) ابحث عن القرص C وانقره ، ثم ابحث عن المجلد الذي سبق وإن حفظت الملف فيه

وليكن المجلد Temp ثم انقره وابحث الملف prog1 .



## الفصل الثاني

### مقاييس وتوزيعات واختبارات إحصائية

#### \* مقاييس النزعة المركزية

سنستعرض في هذا البند المتوسطات على أنواعها مثل الوسط الحسابي والمنوال والوسيط وغيرها من مقاييس النزعة المركزية.

#### \*\* الوسط الحسابي Arithmetic mean

يعرف الوسط الحسابي على أنه حاصل جمع مجموعة من البيانات مقسوماً على عددها.

$$\bar{X} = \frac{1}{n} \sum_{i=1}^n X_i, \quad X_1, \dots, X_n$$

مثال:

احسب الوسط الحسابي لأوزان الطلاب إذا كانت أوزان ستة الطلاب بالكيلو جرام من إحدى المدارس هي

44, 40, 42, 48, 46, 44

الحل:

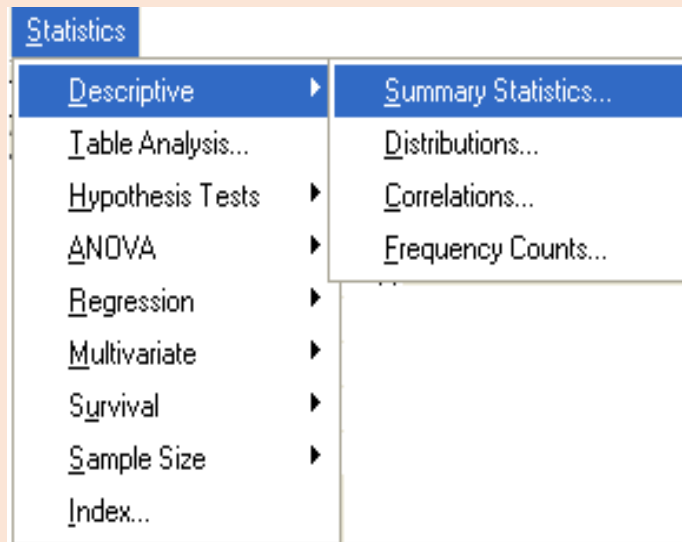
$$\bar{X} = \frac{44 + 40 + 42 + 48 + 46 + 44}{6} = \frac{264}{6} = 44$$

الحل باستخدام ساس:

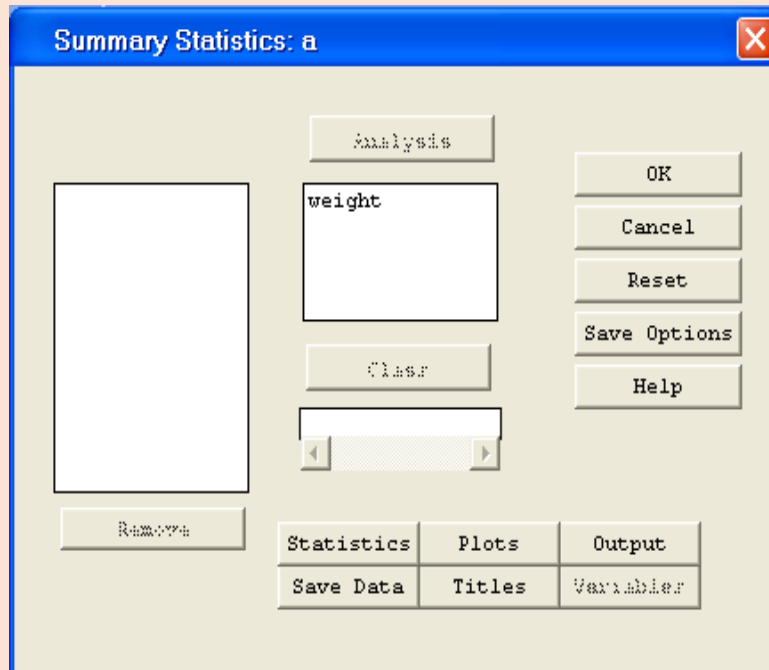
(1) ادخل أوزان الطلاب كما في الشكل التالي:

	weight
1	44
2	40
3	42
4	48
5	46
6	44

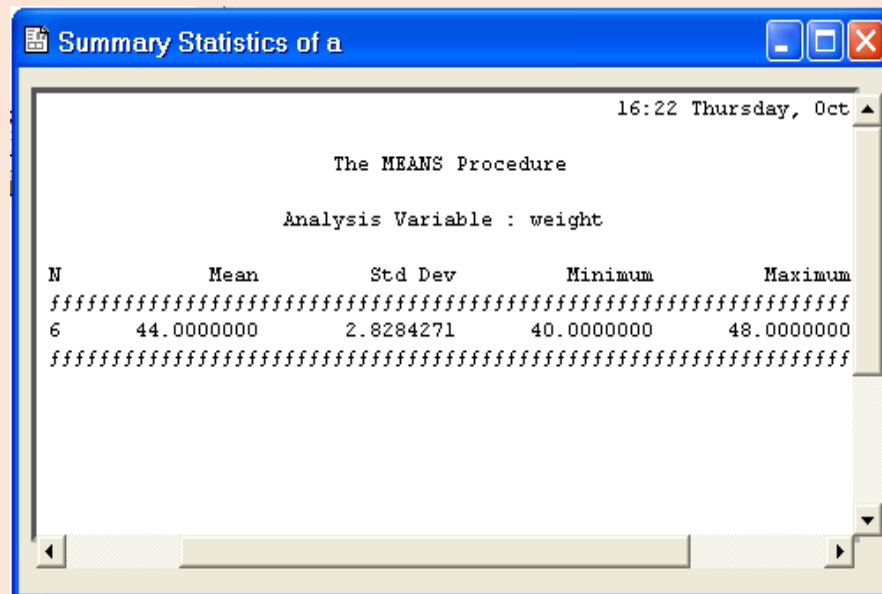
(2) اختر من القائمة **Statistics** الأمر **Descriptive** ومنه اختر **Summary Statistics** كما في الشكل التالي:



(3) يظهر مربع حوار اضغط (حدد) اسم العامود ( weight ) واضغط على المستطيل **Analysis** لتنتقل كلمة **weight** داخل مربع **Analysis** كما في الشكل التالي:



4) اضغط OK ، لتظهر النتائج ( عدد العناصر ، والمتوسط ، والانحراف المعياري ، وأصغر قيمة ، وأكبر قيمة ) . كما في الشكل التالي :



نلاحظ أن متوسط البيانات يساوي 44.

نستطيع كتابة برنامج يعطي نفس النتائج وهو كما يلي:

```
proc means data= work.a;  
var weight;  
run;
```

**\*\* المنوال Mode**

يعرف بأنه القيمة التي لها أكبر تكرار في عينة البيانات الإحصائية، وقد يكون للبيانات أكثر من منوال، وقد لا يوجد منوال على الإطلاق بين البيانات.

مثال:

احسب المنوال لأوزان عينة من الطلاب كانت أوزانهم:

44, 40, 42, 48, 46, 44

الحل:

نلاحظ في عينة القراءات السابقة بأن القيمة 44 كجم متكرر مرتين وتكرارا أكثر من باقي المشاهدات وبذلك يكون المنوال لهذه البيانات هو 44 كجم.

وسوف نحل المثال باستخدام ساس لاحقاً.



## \*\* الوسيط ( Median )

يعرف الوسيط بأنه القيمة التي يسبقها 50% من القراءات بعد ترتيبها تصاعدياً أو تنازلياً، فإذا كان عدد القراءات فردياً فإن الوسيط هو القراءة التي في المنتصف وإذا كان عدد القراءات زوجي فإن الوسيط هو الوسط الحسابي للقراءتين في المنتصف.

مثال:

احسب الوسيط لأوزان عينة من الطلاب بأحادي المدارس كانت أوزانهم هي:

44, 40, 42, 48, 46, 44

الحل:

بعد ترتيب البيانات تصاعدياً نحصل على:

48 , 46 , 44 , 44 , 42 , 40

وحيث أن حجم العينة 6 أي أن حجم العينة زوجي وبالتالي فإن رتبة الوسيط واقعة بين رتبتي  $(\frac{n}{2})$  ،  $(\frac{n}{2} + 1)$  أي أن ترتيب القراءتين هما 3 ، 4 وقيمتين القراءتين هما 46 ، 44 وبالتالي فإن الوسط الحسابي للقراءتين السابقتين وتساوي 44.

وسوف نحل المثال باستخدام ساس لاحقاً.

## \*\* الوسط الهندسي Geometric mean

فهو يعطي قيماً أدق من الوسط الحسابي في دراسة بعض الظواهر التي تزيد مفرداتها بمعدلات ثابتة ويعطى بالصيغة:

$$\bar{X}_G = \sqrt[n]{(X_1).(X_2)....(X_n)}$$

مثال:

أحسب الوسط الهندسي لأعمار عينة من 7 طلاب في المرحلة الابتدائية: أعمارهم

3, 5, 6, 6, 7, 10, 12

الحل: الوسط الهندسي لأعمار الطلاب يحسب كالتالي:

$$\bar{X}_G = \sqrt[7]{(3)(5)(6)(6)(7)(10)(12)}$$

= 6.43 سنة

الحل باستخدام ساس:

نقوم بكتابة البرنامج التالي:

```
DATA a; INPUT X @@ ; CARDS;
```

```
3 5 6 6 7 10 12
```

```
;
```

```
run;
```

ادخال البيانات في عامود x  
حفظها في ملف اسمه a

```
DATA LOGS; SET a;
```

```
IF x > 0 THEN logx = LOG(x);
```

```
ELSE logx = .;
```

```
IF (x ne 0) THEN invx = 1/x;
```

```
ELSE invx = .;
```

```
run;
```

يأخذ اللوغاريتمات لقيم x  
ومعكوساتها ويضعها في  
ملف logs

```
PROC MEANS data = LOGS ;
  VAR logX;
  OUTPUT OUT= MEANLOGS MEAN= X1;
run;
```

يسحب الوسط الحسابي لقيم اللوغاريتمات ويضعها في ملف meanlogs

```
DATA GEOMEAN; SET MEANLOGS;
  GEOMEAN= EXP(x1);
run;
```

يقوم بحساب exp للوسط الحسابي للوغاريتمات لإيجاد الوسط الهندسي .

```
PROC PRINT DATA = GEOMEAN;
  var GEOMEAN;
RUN;
```

يطبع الوسط الهندسي .



ثم نظلل البرنامج ونضغط على زر

لتخرج النتائج في مكتبة Word فنحصل على النتائج كما يلي:

	=
1	3
2	5
3	6
4	6
5	7
6	10
7	12

	x	logx	invx
1	3	1.0986122887	0.3333333333
2	5	1.6094379124	0.2
3	6	1.7917594692	0.1666666667
4	6	1.7917594692	0.1666666667
5	7	1.9459101491	0.1428571429
6	10	2.302585093	0.1
7	12	2.4849066498	0.0833333333

	_TYPE_	_FREQ_	X1
1	0	7	1.8607101473

حيث X1 هي الوسط الحسابي للقيم اللوغاريتمية لـ X (لأعمار الطلاب) وتساوى 1.8607

	_TYPE_	_FREQ_	X1	GEOMEAN
1	0	7	1.8607101473	6.4283001912

ومنه نلاحظ ان قيمة الوسط الهندسي لأعمار الطلاب تساوى 6.43

ثم نقوم بطباعة الوسط الهندسي على نافذة output

**\*\* الوسط التوافقي (Harmonic mean)**

يعتبر الوسط التوافقي من المقاييس التي تحد من تأثير القيم المتطرفة نحو الصغر أو الكبر وخاصة في حالة التطرف نحو الكبر .

ويرمز للوسط التوافقي بالرمز  $(\bar{X}_H)$  إذا كانت لدينا البيانات  $X_1, X_2, \dots, X_n$  فإن :

$$\frac{1}{\bar{X}_H} = \frac{1}{n} \left( \frac{1}{X_1} + \frac{1}{X_2} + \dots + \frac{1}{X_n} \right)$$

مثال:

احسب الوسط التوافقي لأعمار الطلاب وأعمارهم هي

3, 5, 6, 6, 7, 10, 12

الحل:

باستخدام العلاقة السابقة فإن :

$$\frac{1}{\bar{X}_H} = \frac{1}{n} \left( \frac{1}{3} + \frac{1}{5} + \frac{2}{6} + \frac{1}{7} + \frac{1}{10} + \frac{1}{12} \right) = \frac{1}{5.87}$$

أي أن الوسط التوافقي يساوي 5.87 وهو أقل من الوسط الحسابي والوسط الهندسي لنفس البيانات .

الحل باستخدام ساس:

```
TITLE 'HARMONIC MEAN';  
DATA ages; INPUT x @@; CARDS;  
3 5 6 6 7 10 12  
;  
run;
```

ادخال البيانات في عامود  
X وحفظها في ملف A

```
DATA LOGS; SET ages;  
IF x>0 THEN logx=LOG(x);  
ELSE logx=.;  
IF (x ne 0) THEN invx=1/x;  
ELSE invx=.;  
run;
```

يأخذ اللوغاريتمات لقيم x  
ومعكوساتها ويضعها في  
ملف logs

```
PROC MEANS data=LOGS ;  
VAR invx;  
OUTPUT OUT=MEAN2 mean=meaninvx ;  
run;
```

يحسب متوسط  
معكوسات القيم x

```
data harmonic; set MEAN2;
harmonic_mean=1/meaninvx;
run;
```

يحسب الوسط  
التوافقي .



ثم نطل البرنامج ونضغط على زر التنفيذ

لنحصل على النتائج التالية:

	x
1	3
2	5
3	6
4	6
5	7
6	10
7	12

	x	logx	invx
1	3	1.0986122887	0.3333333333
2	5	1.6094379124	0.2
3	6	1.7917594692	0.1666666667
4	6	1.7917594692	0.1666666667
5	7	1.9459101491	0.1428571429
6	10	2.302585093	0.1
7	12	2.4849066498	0.0833333333

	_TYPE_	_FREQ_	meaninvx
1	0	7	0.1704081633

	_TYPE_	_FREQ_	meaninx	harmonic_mean
1	0	7	0.1704081633	5.8682634731

وفي الشكل السابق تظهر نتيجة الوسط التوافقي وهو 5.868

## \* مقاييس التشتت

لقد سبق دراسة مقاييس النزعة المركزية (المتوسطات) وذلك لوصف البيانات عددياً لهذه التوزيعات المختلفة، ومقاييس التشتت هي التي تقيس درجة التجانس أو تشتت مفردات البيانات بعضها عن بعض. ومنها المدى، وانحراف المتوسط، والتباين والانحراف المعياري ومعامل الاختلاف.

## \*\* المدى Rang

يعرف المدى بأنه الفرق بين أكبر قراءة وأصغر قراءة لعينة من البيانات كالتالي:

$$\text{المدى } R = \text{أكبر قراءة} - \text{أصغر قراءة}$$

مثال:

احسب مدى الوزن لعينتين من طلاب المدرستين التاليتين، وعلق على نتيجة

الحسابات

$$42 , 40 , 46 , 44 , 48 \quad ( 1 )$$

$$30 , 52 , 44 , 35 , 59 \quad ( 2 )$$

الحل:

(1) المدى لأوزان الطلاب للمدرسة 1

$$R_1 = 48 - 40 = 8$$

(2) المدى لأوزان الطلاب للمدرسة 2

$$R_2 = 59 - 30 = 29$$



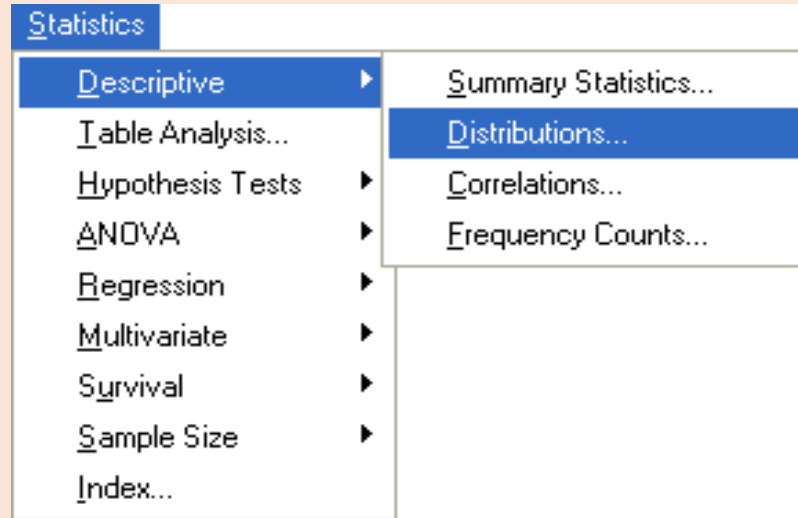
الحل باستخدام ساس:

(1) ادخل البيانات في العمودين A و B كما يلي:

	A	B
1	42	30
2	40	52
3	46	44
4	44	35
5	48	58

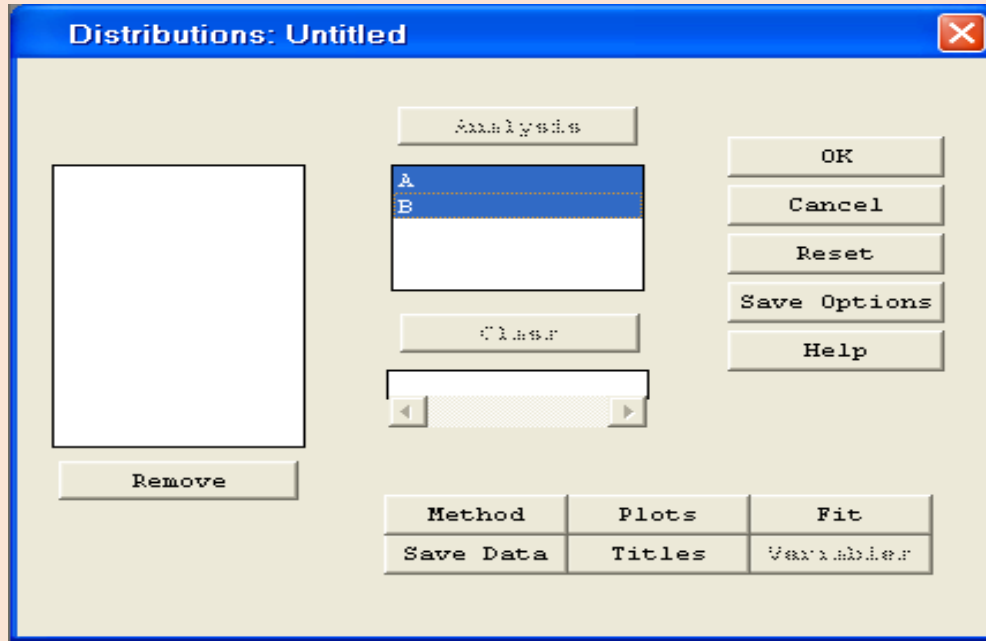
اختر الأمر **Statistics** ثم اختر الأمر الفرعي **Descriptive** ثم الأمر **Distributions**

. كما في الشكل التالي:



يظهر مربع حوار اضغط (حدد) اسم العمودين A, B ثم اضغط على المستطيل **Analysis**

لينتقل العمودين داخل مربع **Analysis** كما في الشكل التالي:



ثم اضغط OK

لتظهر النتائج

وتظهر قيمة المدى للعمود A وقيمة المدى للعمود B ويساوي 29

كما في الشكل التالي:

## Basic Statistical Measures

### Location

### Variability

Mean 44.00000 Std Deviation 3.16228

Median 44.00000 Variance 10.00000

Mode . Range 8.00000

Interquartile Range 4.00000

---

**Basic Statistical Measures**

**Location**

**Variability**

Mean 44.00000 Std Deviation 11.89538

Median 44.00000 Variance 141.50000

Mode . Range 29.00000

Interquartile Range 17.00000

**\*\* الانحراف المتوسط Deviation mean**

يعرف الانحراف المتوسط على أنه متوسط الانحرافات المطلقة للملاحظات عن وسطها

الحسابي  $\bar{X}$  ، ونرمز له بالزمر

ويعرف رياضياً كالتالي: M.D:

$$M .D = \frac{1}{n} \sum_{i=1}^k |X_i - \bar{X}|$$

مثال:

احسب الانحراف المتوسط لأوزان الطلاب

42, 40, 46, 44, 48

الحل:

يتم حساب الانحراف المتوسط كتالي وبما أن المتوسط الحسابي للقراءات

$$M.D = \frac{1}{5} [ |42 - 44| + |40 - 44| + |46 - 44| + |44 - 44| + |48 - 44| ] = 2.4$$

الحل باستخدام ساس:

```
DATA weight; INPUT W @@; CARDS;  
42 40 46 44 48  
;  
run;
```

ادخال البيانات في عامود  
w وحفظها في ملف  
اسمه . weight

```
PROC MEANS data=weight ;  
VAR W;  
OUTPUT OUT=MEAN MEAN=W1;  
run;
```

يحسب المتوسط الحسابي  
لأوزان الطلاب ويحفظه في  
W1. خانة

```
data dmean; set weight;  
dabs=abs(W-44);  
run;
```

يقوم بطرح كل القيم من  
متوسط الحسابي ويحفظها  
ملف اسمه . dmean في

```
PROC MEANS data=dmean ;  
VAR dabs;  
OUTPUT OUT=MEAN1 mean=dvmean ;  
run;
```

يحسب المتوسط الحسابي  
القيم الجديد الظاهر في  
dmean ونحفظه في  
ملف dvmean



ثم نظل البرنامج ونضغط على زر التنفيذ

حيث ينتج كلا من:

	W
1	42
2	40
3	46
4	44
5	48

	_TYPE_	_FREQ_	W1
1	0	5	44

	W	dabs
1	42	2
2	40	4
3	46	2
4	44	0
5	48	4

	_TYPE_	_FREQ_	dvmean
1	0	5	2.4

ومن الشكل السابق نحصل على انحراف المتوسط والذي يساوي 2.4

**\*\* التباين والانحراف المعياري Variance and standard Deviation**

يعتبر التباين ويرمز له بالرمز  $\sigma^2$  والانحراف المعياري هو الجذر التربيعي للتباين ويرمز له بالرمز  $\sigma$  (سيجما) من أهم مقاييس التشتت، وتتلخص فكرة حساب التباين بأنه

متوسط مجموع مربعات الانحرافات للقيم عن وسطها الحسابي  $\bar{X}$  والانحراف المعياري هو الجذر التربيعي للتباين. ويعتبر الانحراف المعياري  $\sigma$  من أهم وأدق وأفضل مقاييس التشتت.

حيث  $\bar{X}$  تمثل الوسط الحسابي للبيانات يتم حساب التباين في حالة العينات حيث يرمز له بالرمز  $S^2$  بدلاً من  $\sigma^2$  لتصبح كالتالي:

$$S^2 = \frac{1}{(n-1)} \sum_{i=1}^n (X_i - \bar{X})^2$$

والانحراف المعياري من العلاقة السابقة هو:

$$S = \sqrt{\frac{1}{(n-1)} \sum_{i=1}^n (X_i - \bar{X})^2}$$

ملاحظة: التباين والانحراف المعياري للعينات دون المجتمع.

مثال:

احسب المنوال والوسيط والتباين والانحراف المعياري لأوزان الطلاب

42, 40, 46, 44, 48, 44

الحل:

لقد تم إيجاد المنوال والوسيط يدوياً وسنجده باستخدام ساس أما التباين والانحراف المعياري فإيجادهما يدوياً كالتالي:

لحساب التباين حيث أن المتوسط الحسابي  $\bar{X} = 44$  هو :

$$S^2 = \frac{1}{5} [(42-44)^2 + (40-44)^2 + (44-44)^2 + (48-44)^2 + (46-44)^2 + (44-44)^2] = \frac{40}{5} = 8$$

ولحساب الانحراف المعياري:

$$S = \sqrt{\frac{1}{5}[(42-44)^2 + (40-44)^2 + (44-44)^2 + (48-44)^2 + (46-44)^2 + (44-44)^2]} = 2.83$$

الحل باستخدام ساس:

لإيجاد المنوال والوسيط والتباين والانحراف المعياري قم بالخطوات التالية:

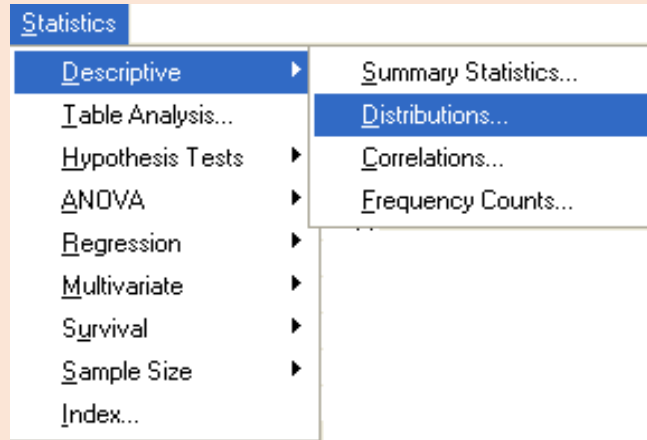
الحل باستخدام ساس:

(1) ادخل أوزان الطلاب كما في الشكل التالي:

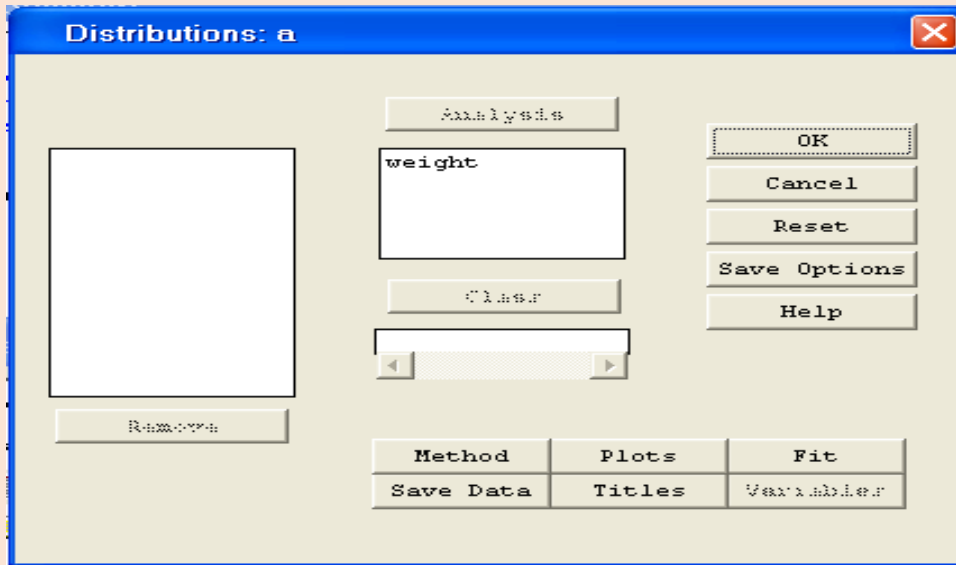
	weight
1	44
2	40
3	42
4	48
5	46
6	44

(2) اختر من القائمة **Descriptive Statistics** الأمر **ومنه اختر**

**Distributions** كما في الشكل التالي:



3) يظهر مربع حوار اضغط أو(حدد) اسم العامود ( weight ) واضغط على المستطيل Analysis لتنتقل كلمة weight داخل مربع Analysis كما في الشكل التالي :



4) اضغط OK ، لتظهر النتائج:



وس يظهر في حزمة ساس نتائج كثيرة منها

## Basic Statistical Measures

### Location

### Variability

**Mean** 44.00000 **Std Deviation** 2.82843

**Median** 44.00000 **Variance** 8.00000

**Mode** 44.00000 **Range** 8.00000

**Interquartile Range** 4.00000

## \* التوزيعات الإحصائية

### \*\* بعض التوزيعات الاحتمالية المتقطعة Discrete Probability Distributions

سوف نقوم بدراسة بعض التوزيعات المتقطعة الخاصة لأهميتها في الحياة العملية لتفسير كثير من الظواهر التطبيقية.

### \*\* توزيع ذي الحدين BINOMIL Distribution

في كثير من التجارب تكون النتيجة فيها أحد أمرين إما نجاح أو فشل وتتألف هذه التجارب من تكرار وإعادة المحاولات المستقلة عن بعضها البعض، فمثلاً عند رمي زهرة النرد فإن النتيجة تكون إما عدداً فردياً أو عدداً زوجياً وتكون نتيجة كل محاولة مستقلة عن نتيجة مقدار ثابت في كل محاولة.  $p$  أي محاولة أخرى واحتمال النجاح

ودالة الكثافة الاحتمالية لتوزيع ذي الحدين كالتالي:

$$f(x) = p(X = x) = \binom{n}{x} p^x (1-p)^{n-x}$$

,  $x = 0, 1, 2, \dots, n$

$$0 < P < 1$$

من خواصه:

np	التوقع
npq	التباين

حساب الاحتمالات باستخدام توزيع ذي الحدين:

$$P( X \leq a ) = f( 0 ) + f( 1 ) + \dots + f( a )$$

$$P( X \leq a ) = f( a ) + f( a + 1 ) + f( a + 2 ) + \dots + f( n )$$

مثال:

في أحد لمدن كانت نسبة المصابين من كبار السن بمرض السكر هي 30%  
فخضعت عينة مكونة من 5 أفراد من كبار السن للتأكد من أصابتهم بمرض السكر، وبفرض  
أن المتغير العشوائي  $X$  يمثل عدد المصابين في العينة  
ما هو احتمال وجود شخص واحد على الأقل مصاب؟

الحل باستخدام ساس:

الاحتمال هو:

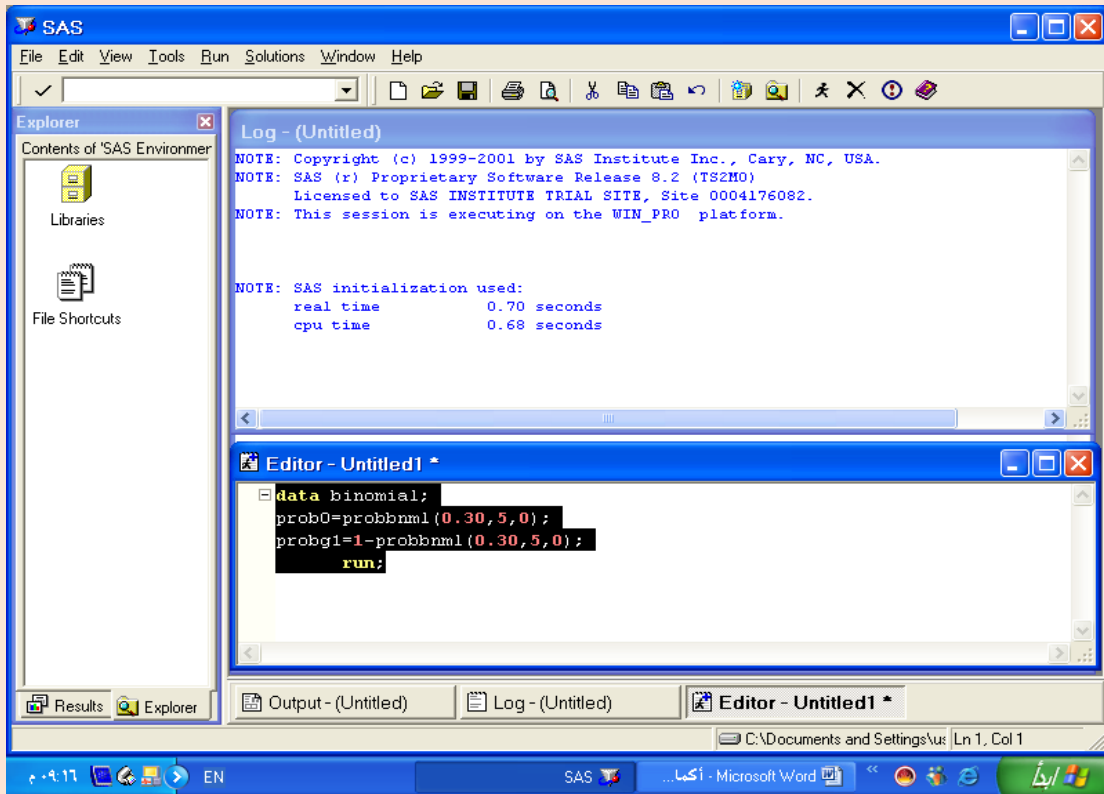
$$\begin{aligned} P(X \geq 1) &= f( 1 ) + f( 2 ) + f( 3 ) + f( 4 ) + f( 5 ) \\ &= 1 - f( 0 ) \end{aligned}$$

(1) سوف نقوم بحساب  $f( 0 )$

بكتابة البرنامج التالي:

```
data binomial;  
prob0=probbnml(0.30,5,0);  
prob1=1-probbnml(0.30,5,0);  
run;
```

نظّل البرنامج كما في الشكل التالي: 2)



3) ثم نضغط على زر التنفيذ

4) نفتح المكتبات **libraries** نفتح المكتبة **work** فنجد الملف **binomial**،

الذي توجد به نتائج البرنامج ثم ننقر عليه بالنقر المزدوج لنجد النتائج التالية:

	prob0	probg1
1	0.16807	0.83193

حيث أن قيمة  $f(0)$  هي:

$$f(0) = 0.16807$$

ويكون احتمال إصابة شخص واحد على الأقل هي:

$$P(X \geq 1) = 1 - 0.16807 = 0.83193$$

\* توليد الأرقام العشوائية من توزيع ذي الحدين باستخدام ساس:

يمكننا توليد أرقام عشوائية من التوزيع ذي الحدين من خلال حزمة ساس وذلك بكتابة

البرنامج في نافذة تحرير البرنامج كما يلي:

مثال: ولد من 1 الى 100 رقما عشوائيا من توزيع ذي الحدين باحتمال

$$p=0.6 \text{ و } n = 5 \text{ و } q = 3.$$

الحل باستخدام ساس:

```
data ranbinom;  
do _i_ = 1 to 100;  
Binom1 = ranbin(3,5,0.6);  
output;  
end;  
run;
```

## \*\* توزيع بواسون Poisson Distribution

إن التجارب التي تعطينا عدد النجاحات في فترة زمنية أو منطقة محددة تسمى تجارب بواسون ، فعدد حوادث السيارات على طريق في الأسبوع وعدد المكالمات الهاتفية التي تصل إلى مكتب كل عشرة دقائق جميعها أمثلة على تجارب بواسون .

ودالة الكثافة الاحتمالية لتوزيع بواسون كالتالي:

$$p(x; \lambda) = f(x) = p(X = x) = \frac{\lambda^x}{x!} e^{-\lambda}, \quad x = 0, 1, 2, \dots; \quad \lambda > 0$$

ومن خواصه:

$\lambda$	التوقع
$\lambda$	التباين

حساب الاحتمالات باستخدام توزيع بواسون :

$$P(X \leq a) = f(0) + f(1) + \dots + f(a)$$

$$P(X \geq a) = f(a) + f(a+1) + f(a+2) + \dots$$

$$= 1 - f(0)$$

مثال:

إذا كان المعدل اليومي للحوادث المرورية في احدى المدن يساوى 24 حادث

ما احتمال وقوع حادثين في الساعة القادمة؟

الحل باستخدام ساس:

نريد حساب احتمال  $P(X = 2) = f(2)$  ويكون المتوسط  $\lambda = \frac{24}{24} = 1$  نقوم

بالخطوات التالية:

(1) كتابة البرنامج في نافذة تحرير البرنامج كما يلي:

```
data poisson;
```

```
x= pdf('POISSON',2,1);
```

```
run;
```



(2) نظل البرنامج ونضغط على زر التنفيذ

(3) نفتح المكتبات **libraries** ثم نفتح المكتبة **work** فنجد الملف **poisson**

الذي توجد به نتائج البرنامج ثم ننقر عليه بالنقر المزدوج لتكون النتائج:

	x
1	0.1839397206

تظهر قيمة الاحتمال تساوى 0.18 أي أن

$$f(2) = 0.18$$

توليد أرقام عشوائية من توزيع بواسون :

مثال: ولد من 1 الى 100 رقما عشوائيا بمتوسط  $M=1$  وعدد صحيح عشوائي  $N=5$

الحل باستخدام ساس:

```
data Pran;  
do i=1 to 10;  
X1=ranpoi(1,5);  
output;  
end;  
run;
```

نحصل على ملف موجود في مكتبة work اسمه pran يحتوى على الأرقام العشوائية

	i	X1
1	1	3
2	2	10
3	3	4
4	4	3
5	5	8
6	6	10
7	7	5
8	8	5
9	9	2
10	10	2



\* بعض التوزيعات الاحتمالية المتصلة

## Continuous Probability Distributions

سوف نعرض في هذا الجزء بعض من التوزيعات المتصلة التي لها أهمية في التطبيقات

العملية:

### \*\* التوزيع الطبيعي Normal Distribution

كثير من الظواهر العشوائية التي تنتج عن تجمع متغيرات كثيرة تعطي توزيع تكراري، فأوجد جاوس صيغة رياضية لدالة كثافة احتمالية تسمى دالة التوزيع الطبيعي تنطبق على أي توزيع تكراري معطى لكثير من القياسات المختلفة وتمثل توزيعاً مثالياً لها.

ودالة الكثافة الاحتمالية للتوزيع الطبيعي كالتالي:

$$f(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}, \quad -\infty < x < \infty ; \quad -\infty < \mu < \infty ,$$

$$\sigma > 0$$

من خواصه:

$\mu$	التوقع
$\sigma^2$	التباين

المنحنى الاحتمالي للتوزيع الطبيعي يشبه الناقوس من حيث الشكل، ومن خصائصه أنه متماثل حول المحور الرأسي الذي يمر بقمته، أي يقسم المنحنى الطبيعي إلى قسمين متناظرين (متماثلين) في الشكل والمساحة، كما أن المنحنى له نقطة انقلاب عند

$$X = \mu \pm \sigma$$

ونلاحظ أن 99% من قيم المتغير العشوائي  $x$  تقع داخل الفترة

$$[\mu - 2.58\sigma, \mu + 2.58\sigma]$$

أي أنه نادراً ان نجد قيمة من قيم  $x$  خارج هذه المنطقة، كم ان 95% من قيم المتغير  $x$  تقع داخل الفترة

$$[\mu - 1.96\sigma, \mu + 1.96\sigma]$$

وان 68% منها تقع داخل الفترة

$$[\mu - \sigma, \mu + \sigma]$$

ويمكن حساب احتمال  $x$  في أي فترة نريدها، وذلك بإيجاد قيمة تكامل الدالة الاحتمالية

$f(X)$  داخل هذه الفترة، أي إيجاد المساحة المحصورة تحت منحنى هذه الدالة داخل هذه

الفترة باستخدام التكامل، ولكن باستخدام حزمة ساس لن نحتاج الى حساب ذلك حيث نستطيع الوصول الى ذلك مباشرة.

مثال:


إذا كان  $X \sim N(2, 25)$  أوجد قيمة  $a$  لكلاً مما يلي:

$$(1) P(X < a) = 0.421$$

$$(2) P(X < 1) = a$$

الحل باستخدام ساس:

```
data normal;  
z=probit(0.421);  
a1=5*z+2;  
znor=(1-2)/5;  
xnor_a2=probnorm(znor);  
run;
```

نظّل البرنامج ونضغط على زر التنفيذ  وتكون النتائج:

	z	a1	znor	xnor_a2
1	-0.199335898	1.0033205097	-0.2	0.4207402906

حيث أن:

$$(1) P(X < a) = 0.241 \longrightarrow a = 1.00332051$$

$$(2) P(X < 1) = a \longrightarrow a = 0.241$$

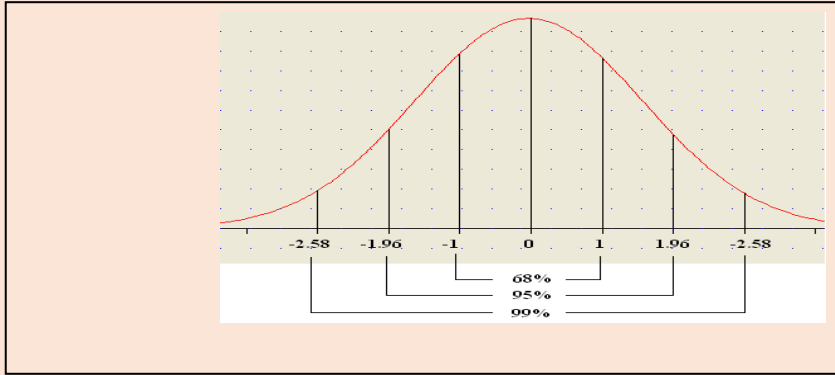
## \*\* التوزيع الطبيعي القياسي Standard Normal Distribution

و  $\sigma^2$  ( $\mu$ ) : في السابق كان دالة التوزيع الاحتمالي للتوزيع الطبيعي يعتمد على قيم  $\mu$  وتتغير قيمته بتغيرهما ، لهذا كان من الضروري إيجاد طريقة لا تعتمد على قيم  $\mu$  و  $\sigma^2$  التالي: لهذا الغرض يستخدم المتغير العشوائي

له توقع  $E(X) = \mu$  و تباين  $V(X) = \sigma^2$  فإن المتغير  $X$  إذا كان المتغير العشوائي العشوائي  $Z = \frac{X - \mu}{\sigma}$  ويكون له توقع  $E(Z) = 0$  وتباين  $V(Z) = 1$  وتكون دالة الكثافة كالتالي:  $Z$ : الاحتمالية للمتغير العشوائي

$$f(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}}, \quad -\infty < z < \infty$$

الشكل التالي يوضح المنحنى الاحتمالي للتوزيع الطبيعي القياسي، والنسب للاحتمالات تحت المنحنى:



من خواصه:

0	التوقع
1	التباين

والدالة التراكمية للتوزيع الطبيعي القياسي كالتالي:

$$F(z) = p(Z \leq z) = \int_{-\infty}^z f(t)dt, \quad -\infty < z < \infty$$

حساب الاحتمالات باستخدام التوزيع الطبيعي القياسي:

قام الإحصائيون بعمل جداول لحساب المساحات تحت المنحنى الطبيعي القياسي ووضعها في جدول، بحيث تتمكن من حساب الاحتمالات بالرجوع إلى الجدول ولكن باستخدام حزمة ساس لن نحتاج الرجوع إلى الجدول بل سنحصل على القيم مباشرة.

مثال أوجد قيم  $a$  لكلا من:

$$(1) \quad P(Z \leq -0.34) = a$$

$$(2) \quad P(Z \leq a) = 0.3669$$

الحل باستخدام ساس:

```
data stanorm;
```

```
a1=probnorm(-0.34);
```

```
z=probit(0.3669);
```

```
run;
```

الملف **Work**. بعد ذلك نبحث في مكتبة  نظل البرنامج ونضغط على زر التنفيذ

ونفتحه لنجد النتائج كما في الشكل التالي: **stanorm:**

	a1	a2
1	0.366928264	-0.340075064

حيث أن :

- ( تساوي 2 في الفقرة a ) . بينما قيمة 0.3669 ( تساوي 1 في الفقرة a ) قيمة  
0.34.

\*\* توزيع t Distribution

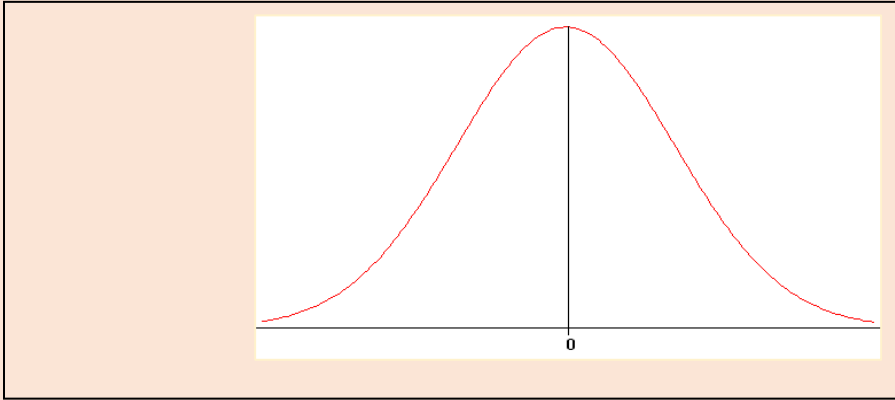
ودالة كثافته الاحتمالية كتالي :

$$f(x) = \frac{1}{\sqrt{\nu} B(1/2, \nu/2)} \left(1 + \frac{t^2}{\nu}\right)^{\left(\frac{-\nu-1}{2}\right)}, \quad -\infty < t < \infty$$

من خواصه :

0	التوقع
$\frac{\nu}{\nu-2}$	التباين

منحنى التوزيع يشبه الناقوس ومتناظر حول المحور  $T=0$  كالتالي:



حساب الاحتمالات باستخدام توزيع t

ولحساب أي احتمالات حول المتغير T

يلزم وجود جدول يبين المساحات المختلفة تحت منحنى دالة الكثافة الاحتمالية لكل من قيم  $V$  المختلفة ، ولكن يمكننا ساس من الحصول على الاحتمالات دون الحاجة للرجوع للجدول .

مثال:

أوجد قيمة  $a$  في كلا من ما يلي :

علماً بأن

$$25 = V$$

$$(1) \quad P(T_v \geq 1.7) = a$$

$$(2) \quad P(T_v \geq a) = 0.05$$

الحل باستخدام ساس :

```
data t1;
```

```
cp=probt(1.7,25);
```

```
a=1-cp;
```

```
Tinvx2=tinv(0.95,25);
```

```
run;
```

نظل البرنامج ثم نضغط على زر التنفيذ فنجد قيمة

في الفقرة (1) و

$$0.0507705852 = a = a$$

$$1.708 = a = \text{Tinvx2} \text{ في الفقرة (2)}$$

اما cp فهي العدد التراكمي كما في الشكل التالي:

	cp	a	Tinvx2
1	0.9492294148	0.0507705852	1.7081407613

### \*\* توزيع كاي تربيع Chi-square distribution

توزيع مربع كاي ( $\chi^2$ ) اكتشفه العالم كارل بيرسون ويعتبر من التوزيعات الاحتمالية المهمة في كثير من التطبيقات العملية في الدراسات الإحصائية ودالة كثافة الاحتمالية تعتمد على قيمة المعلمة  $\nu$  والتي تمثل درجات الحرية  $\nu = n - 1$  كالتالي :

$$f(\chi^2_\nu) = \frac{1}{2\Gamma(\nu/2)} \left( \frac{\chi^2_\nu}{2} \right)^{\left(\frac{\nu}{2}-1\right)} e^{-\left(\frac{\chi^2_\nu}{2}\right)} \quad 0 < \chi^2_\nu < \infty$$

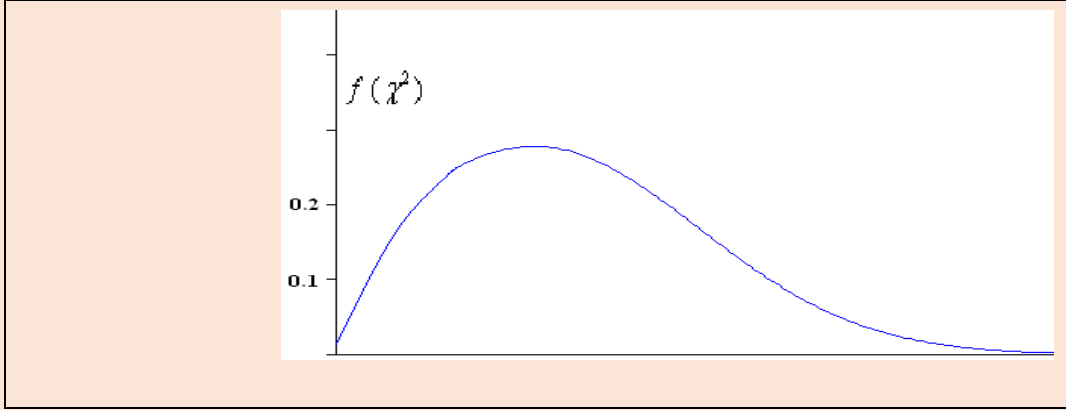
ومن خواصه :

$\nu$	التوقع
$2\nu$	التباين

منحنى توزيع كاي تربيع يقع في الجهة اليمنى للمحور الرأسي ، الممثل لدالة الكثافة

الاحتمالية ، ومنحنى التوزيع التكراري له غير متناظر ، وملتو نحو اليمين كالتالي :





حساب الاحتمالات باستخدام توزيع كاي تربيع :

لحساب الاحتمالات في السابق يلزم وجود جدول يبين المساحات المختلفة تحت المنحنى دالة الكثافة الاحتمالية يوضح قيم  $\chi^2$  المختلفة ، ولكن لن نحتاج لهذا الجدول بفضل توافر إمكانية الحصول على الاحتمالات مباشرة باستخدام حزمة ساس .

مثال:

أوجد قيمة  $a$  في كلا من ما يلي :

$$(1) \quad P(\chi_{15}^2 \geq 32.8) = a$$

$$(2) \quad P(\chi_{15}^2 \geq a) = 0.005$$

الحل باستخدام ساس :

```
data chi;  
x=probchi(32.8,15);  
a1=1-x;  
q1=cinv(1-0.005,15);  
run;
```

نظّل البرنامج ثم نضغط على زر التنفيذ لتظهر النتائج في مكتبة work

	x	a1	q1
1	0.994997903	0.005002097	32.801320646

حيث أن :

$$(1) P(\chi_{15}^2 \geq 32.8) = a \longrightarrow a1 = 0.005$$

$$(2) P(\chi_{15}^2 \geq a) = 0.005 \longrightarrow q1 = 32.8$$

X هو العدد التراكمي ويساوى 0.995

\*\* توزيع F (F distribution)

ويعتبر من التوزيعات الاحتمالية الهامة في علم الإحصاء وتطبيقاته، ودالة كثافة

الاحتمال هي:

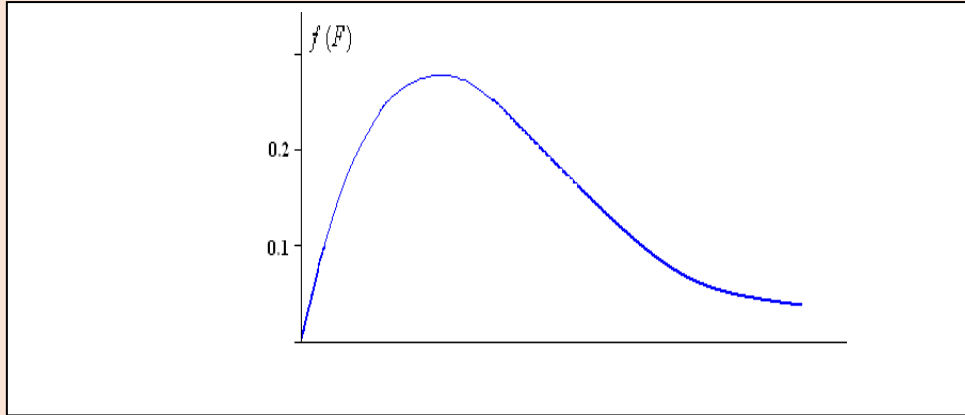
$$f(F) = \frac{\Gamma(v_1/2 + v_2/2)}{\Gamma(v_1/2)\Gamma(v_2/2)} \left(\frac{v_1}{v_2}\right)^{(v_1/2)} F^{v_1/2-1} \left(1 + \frac{v_1}{v_2} F\right)^{-(v_1+v_2)/2}, \quad 0 < F < \infty$$

حيث درجات الحرية لتوزيع  $v_1$  و  $v_2$

من خواصه :

$\frac{v_1}{v_2 - 2} , v_1, v_2 > 2$	التوقع
$\frac{2v_2^2(v_1 + v_2 - 2)}{v_1(v_2 - 2)^2(v_2 - 4)} , v_2 > 4$	التباين

ومنحنى التوزيع يظهر بالشكل التالي:



حساب الاحتمالات باستخدام التوزيع F

بدرجات حرية  $(v_1, v_2)$  التي توجد على يمينها مساحة F يعطي قيم F هناك جدول لتوزيع قدرها  $a$  . ويمكننا الحصول على الاحتمالات عن طريق حزمة ساس ، ويتضح ذلك بالمثل التالي :

مثال:

أوجد قيمة  $a$  في كلا مما يلي :

$$(1) \quad P(F_{8,6} \geq 4.14) = a$$

$$(2) \quad P(F_{8,6} \geq a) = 0.05$$

الحل باستخدام ساس :

data F;

x=probf(4.14,8,6);

a1=1-x;

q1=finv(.95,8,6);

run;

نظل البرنامج ونضغط على زر التنفيذ لتظهر النتائج في مكتبة Work

كما في الشكل التالي :

	x	a1	q1
1	0.9498176503	0.0501823497	4.1468041623

حيث أن :

$$(1) \quad P(F_{8,6} \geq 4.14) = a \longrightarrow a = a1 = 0.05$$

$$(2) \quad P(F_{8,6} \geq a) = 0.05 \longrightarrow a = q1 = 4.148$$

. 0.9498 هو العدد التراكمي ويساوي x و

\*\* الربيعات والعشيرات والمئينات والحد الأعلى والحد الأدنى :

الربيعات في البيانات المرتبة تصاعدياً هي :

- الربيع الأول  $Q_1$  هو القيمة التي يسبقها ربع البيانات ويليهما ثلاثة أرباع البيانات .
- الربيع الثاني  $Q_2$  هو الوسيط وهو القيمة التي يسبقها نصف البيانات ويليهما النصف الآخر .
- الربيع الثالث  $Q_3$  وهو القيمة التي يسبقها ثلاثة أرباع البيانات ويليهما ربع البيانات .

أما نقاط التقسيم إلى عشرة أقسام متساوية في المساحة فتسمى العشيرات . فالعشير الأول هو القيمة التي يسبقها  $\frac{1}{10}$  من البيانات ويتبعها  $\frac{9}{10}$  من البيانات على فرض أن القيم مرتبة تصاعدياً . وكذلك العشيرات الأخرى وبنفس الطريقة . ونرمز للعشيرات بالرموز  $D_1, D_2, \dots, D_9$  .

وعند التقسيم إلى مائة قسم متساو في المساحة فتسمى المئينات ، فالمئين الأول ورمزه  $P_1$  هو القيمة التي يسبقها  $\frac{1}{100}$  من البيانات ويليهما  $\frac{99}{100}$  من البيانات على فرض أن القيم مرتبة تصاعدياً . وكذلك المئينات الأخرى ، ونلاحظ أن المئين الخمسين هو الوسيط وهو الربيع الثاني وهو العشير الخامس أي أن :

$$P_{50} = \mu = Q_2 = D_5$$

أما الحد الأدنى هي أصغر قيمة لمجموعة من القيم ، والحد الأعلى هي أكبر قيمة لمجموعة من القيم .

مثال:

أوجد الربيعات الثلاثة والمئين الخامس والسبعون , ثم أوجد الحد الأدنى والحد الأعلى

للأرقام التالية:

42, 40, 46, 44, 48

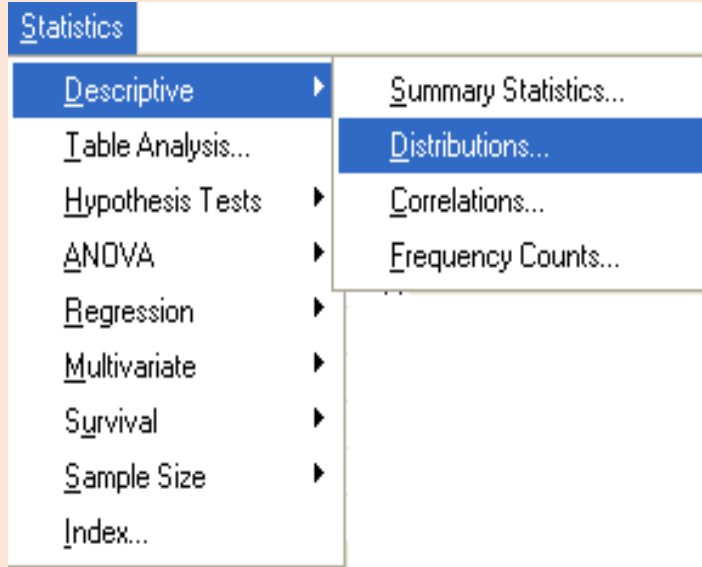
الحل باستخدام ساس :

(1) ادخل البيانات واحفظها كما يلي:

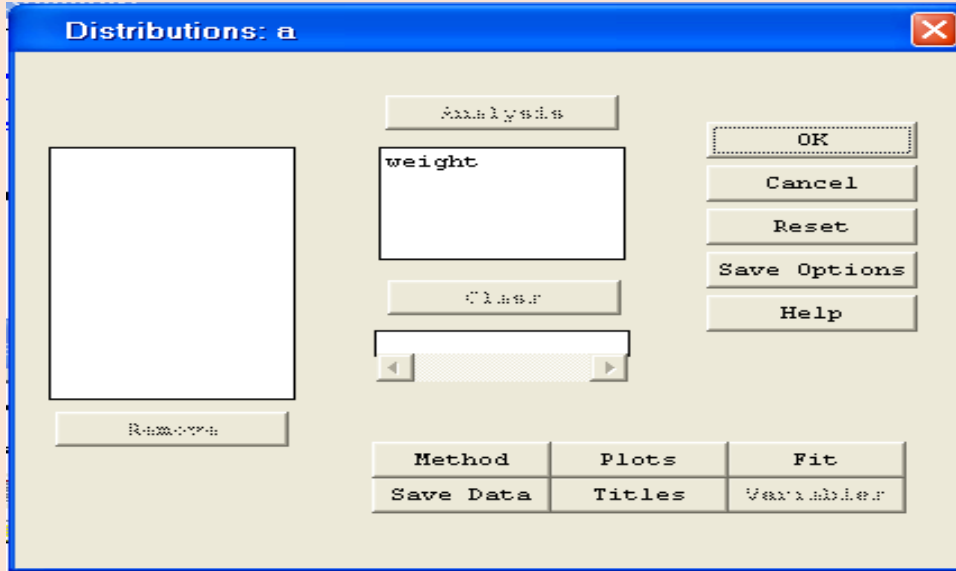
	Weight
1	42
2	40
3	46
4	44
5	48

(2) اختر من القائمة **Statistics** ومنه أختار **Descriptive** ثم **Distributions**

كما في الشكل التالي :



(3) يظهر مربع حوار اضغط او حدد اسم العامود weight واضغط على المستطيل Analysis لتنتقل كلمة Weight داخل مربع Analysis كما يلي:



(4) اضغط OK لتظهر النتائج التالية:

### Quantiles (Definition 5)

	Quantile	Estimate
48	100%	Max
48	99%	
48	95%	
48	90%	
46	75%	Q3
44	50%	Median
25%	Q1	
10%	40	
5%	40	
1%		
40		0%
Min	40	

ومنها نجد أن

الربيع الأول  $Q_1 = 42$  والربيع الثاني  $Q_2 = 44$   
والربيع الثالث  $Q_3 = 46$  كما أن المئين الخامس والسبعون  $P_{75} = 46$   
والحد الأدنى  $= 40$  والحد الأعلى  $= 48$ .



\*مقاييس الالتواء والتفرطح :

تمكننا حزمة ساس من إيجاد مقاييس الالتواء والتفرطح .

\* مقاييس الالتواء : Skewness

يكون التوزيع التكراري ملتوياً نحو اليمين أو موجب الالتواء إذا كان ممتداً أكثر نحو اليمين أما إذا كان له طرف ممتد أكثر نحو اليسار فيقال إنه سالب الالتواء أو ملتو نحو اليسار ويرمز له بالرمز  $sk$  كالتالي:

$$sk = \frac{n}{(n-1)(n-2)} \sum \left( \frac{x_j - \bar{x}}{s} \right)^3$$

\* التفرطح: kurtosis

وهو يقيس مقدار التدبب لقمة منحنيات التوزيعات التكرارية بالنسبة لقمة منحنى التوزيع الطبيعي ارتفاعاً أو انخفاضاً ، وتسمى قمة منحنى التوزيع الطبيعي متوسطة التفرطح ومعامل تفرطحه يساوى 3 ويعطى بالعلاقة التالية:

$$\left[ \frac{n(n+1)}{(n-1)(n-2)(2-3)} \sum \left( \frac{x_j - \bar{x}}{s} \right)^4 \right] - \left[ \frac{3(n-1)^2}{(n-2)(n-3)} \right]$$

مثال: احسب مقدار الالتواء لاعداد لعينة من 7 طلاب أعمارهم:

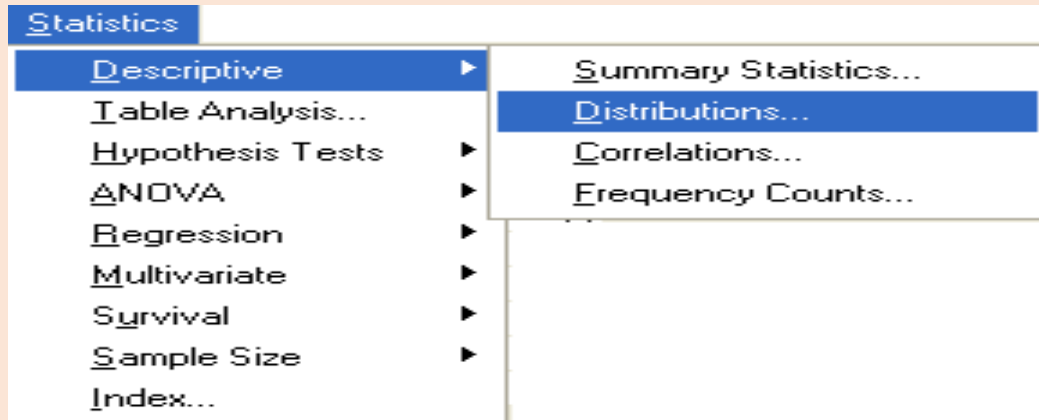
3 , 5 , 6 , 6 , 7 , 10 , 12

الحل باستخدام ساس:

(1) ادخل البيانات واحفظها في مكتبة Work وليكن باسم ages كما يلي

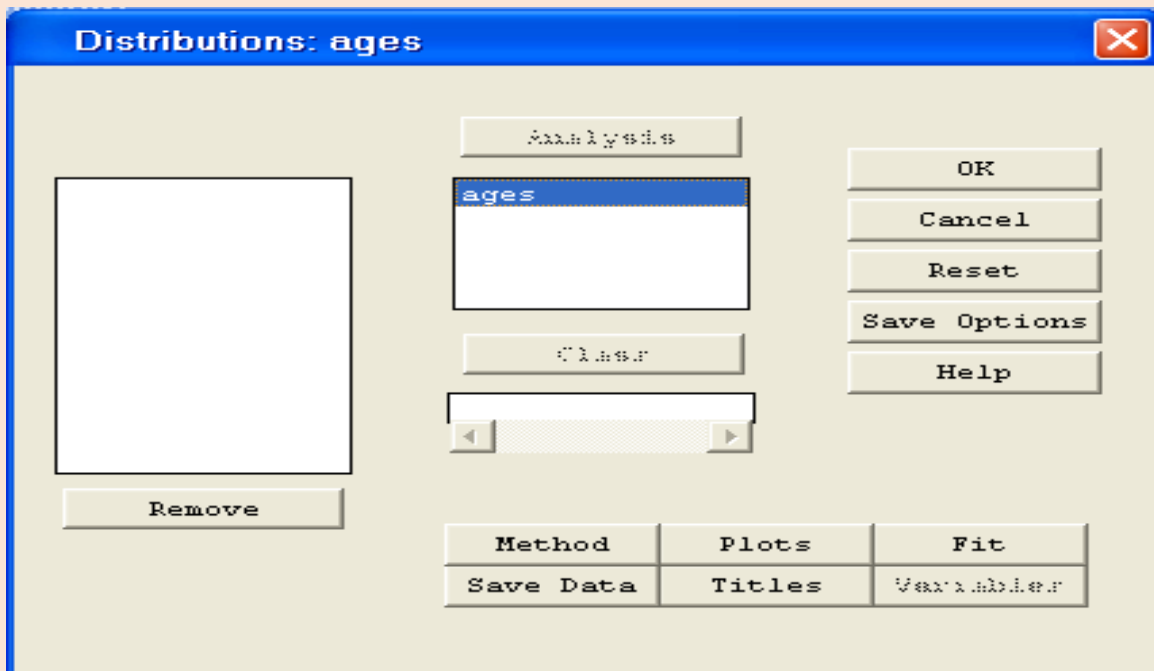
	ages
1	3
2	5
3	6
4	6
5	7
6	10
7	12

(2) اختار من القائمة Statistics الأمر Descriptive ومنه نختار Distributions



(3) يظهر مربع حوار اضغط أو حدد اسم العمود ages واضغط على المستطيل Analysis

لتنقل كلمة ages داخل مربع Analysis كما في الشكل:



(4) اضغط OK لتظهر النتائج نبحث عن النتائج التي على الشكل التالي

### The UNIVARIATE Procedure

Variable: ages

#### Moments

N	7	Sum Weights	7
Mean	7	Sum Observations	49
Std Deviation	3.05505046	Variance	9.333333333
Skewness	0.63828733	Kurtosis	-0.15
Uncorrected SS	399	Corrected SS	56
Coeff Variation	43.643578	Std Error Mean	1.15470054

ومنه نجد أن :

الالتواء = skewness = 0.638 والتفرطح = kurtosis = -0.15



## الفصل الثالث

### أجراء الاختبارات الإحصائية باستخدام ساس

يتعرض الإنسان في كثير من الحالات وفي مجالات العمل المختلفة إلى مواقف معينة تتطلب منه اتخاذ قرار بناء على معلومات محسوبة من عينة ، وطبيعي أن يتخذ هذه القرار بشيء من الحكمة وبأقل قدر ممكن من المخاطر . سوف نوضح كيفية اتخاذ قرار سليم ابتداء بصياغة القرار : أن المطلوب هو اختبار فرض العدم  $H_0$  حول أحد المعالم لتكن  $\theta$  ، لذلك فإن فرض العدم لاختبار المعلمة  $\theta$  يمكن صياغته على الصورة :

$$H_0 : \theta = \theta_0$$

والفرض البديل هو أحد الحالات التالية :

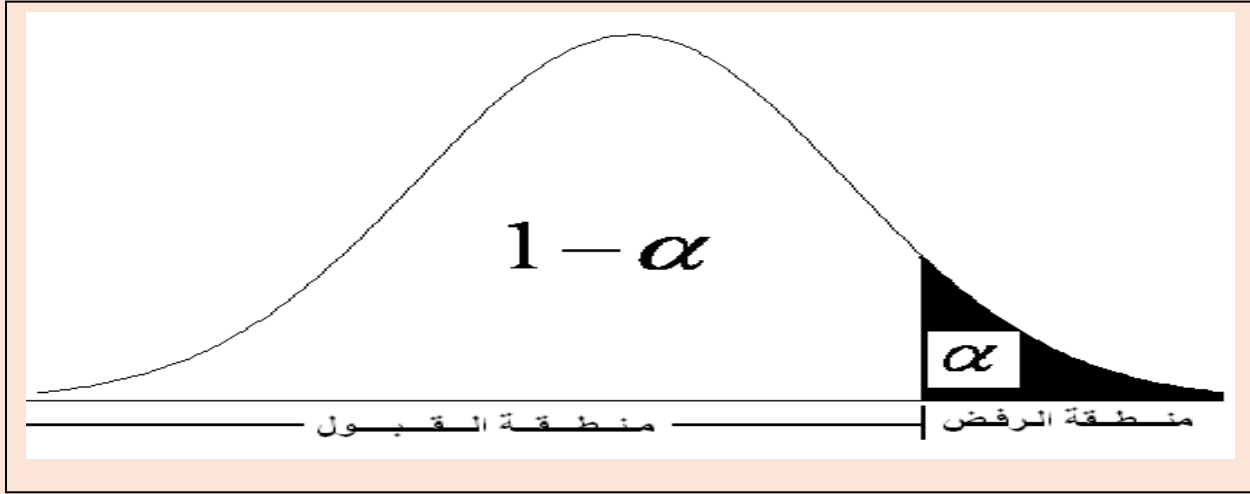
$$H_1 : \theta > \theta_0 \quad \text{اختبار من طرف واحد من الجهة اليمنى}$$

$$H_1 : \theta < \theta_0 \quad \text{اختبار من طرف واحد من الجهة اليسرى}$$

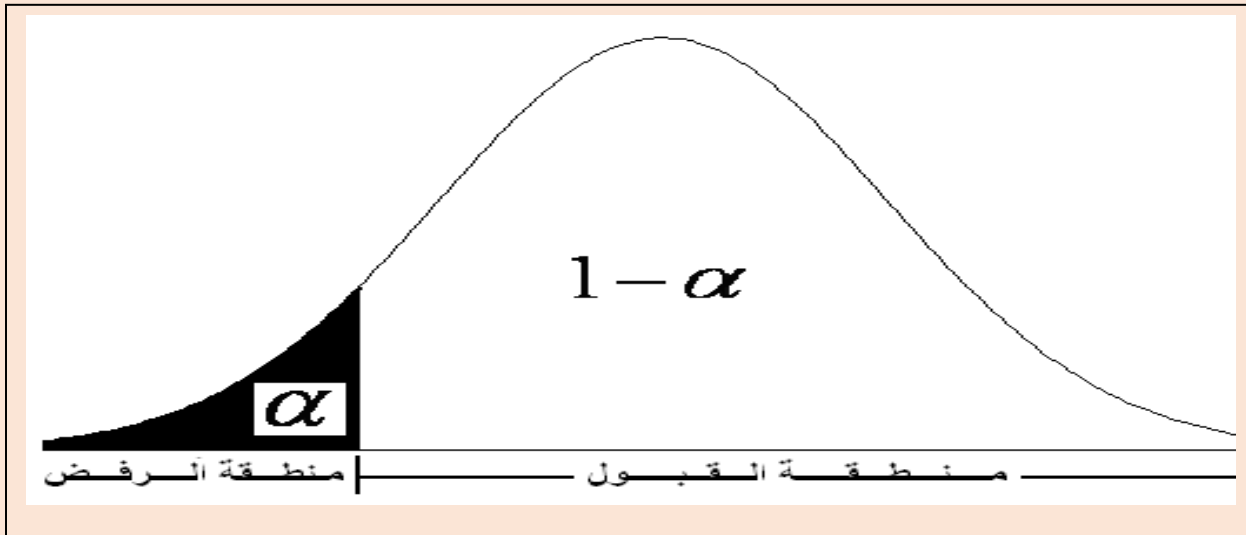
$$H_1 : \theta \neq \theta_0 \quad \text{اختبار من طرفين اليمنى واليسرى}$$

وبناء على الفرض البديل  $H_1$  وعلى مستوى المعنوية  $a$  يمكن تقسيم المحور الأفقي لتوزيع إحصاءه الاختبار إلى منطقتين إحداهما تسمى منطقة القبول والأخرى تسمى منطقة الرفض ، وهذه المناطق تقسم المساحة أسفل منحنى التوزيع المستخدم بحيث أن منطقة ، ويمكن تصنيف عملية - 1 الرفض تمثل بالمساحة  $a$  أما مساحة منطقة القبول فإنها  $a$  قبول أو رفض القرار إلى ثلاث حالات هي كالتالي :

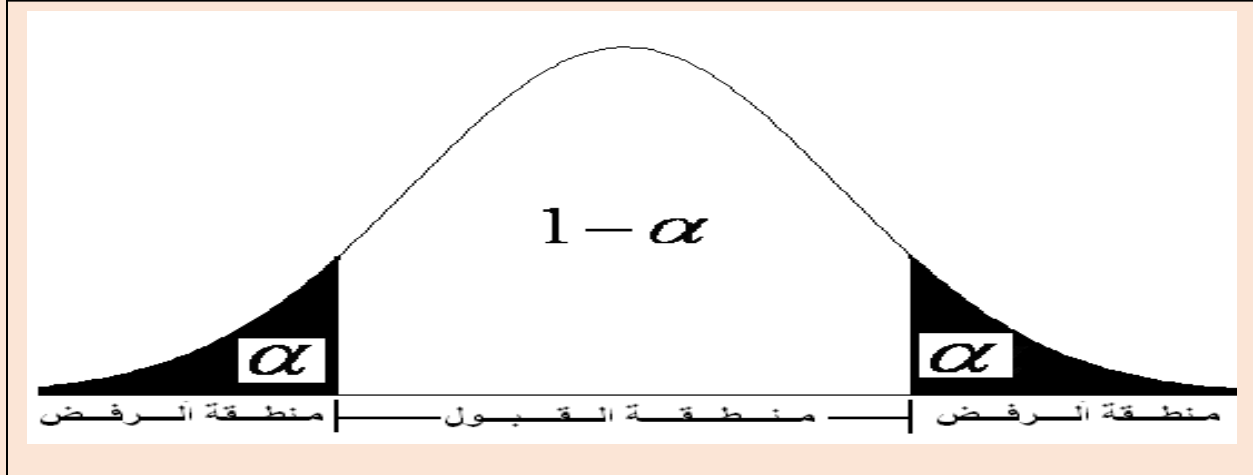
(1) اختبار من طرف واحد من الجهة اليمنى  $H_1: \theta > \theta_0$



(2) اختبار من طرف واحد من الجهة اليسرى  $H_1: \theta < \theta_0$



(3) اختبار من طرفين ( من الجهة اليمنى واليسرى )  $H_1: \theta \neq \theta_0$



حزمة ساس تعطيك قيمة تمكّنك من اتخاذ قرار إما بالقبول أو الرفض ودرجة القبول والرفض حيث تبين أقل مستوى معنوي يتم رفض فرض العدم عنده ، تسمى غالباً ويتم مقارنتها مع مستوى المعنوية كالتالي: P\_value:

نرفض فرض العدم  $H_0$  عندما :

$$P\_value < a$$

وهذه القيمة متوافرة في حزمة ساس .

## \*\* اختبار Z

يمكن استخدامه في اختبارات حول متوسط المجتمع الطبيعي ( مقارنة متوسط مجتمع بقيمة معينة ) سواء حجم العينة أو كبير والتباين معلوم أو مجهول وتكون الفروض كالتالي

$$H_0 : \mu = \mu_0$$

ضد أحد الفروض البديلة :

$$H_1 : \mu > \mu_0 , \quad H_1 : \mu < \mu_0 , \quad H_1 : \mu \neq \mu_0$$

مثال: يدعي أحد الموردين للقماش لمحل خياطة بأن القماش يتصف بقوة تحمله للشد

في المتوسط تزيد عن 200 وكانت الدراسات والتجارب السابقة تشير الى ان تباين هذا القماش يمكن اعتباره 100 قام صاحب المحل بأخذ عينة عشوائية من أربعة قطع من هذا القماش وتم قياس قوة تحمله (بوحدة مناسبة) كالتالي:

$$204 , 205 , 215 , 208$$

المطلوب اتخاذ القرار بالتعامل أو عدم التعامل مع هذا المورد عند مستوى معنوية

$$\alpha = 0.05 .$$

الحل باستخدام ساس :

بما أن المورد يدعي أن قوة تحمل القماش للشد تزيد عن 200 أي نختبر عكس ادعاء

المورد (من الجهة اليسرى) كالتالي:



$$H_0 : \mu = 200$$

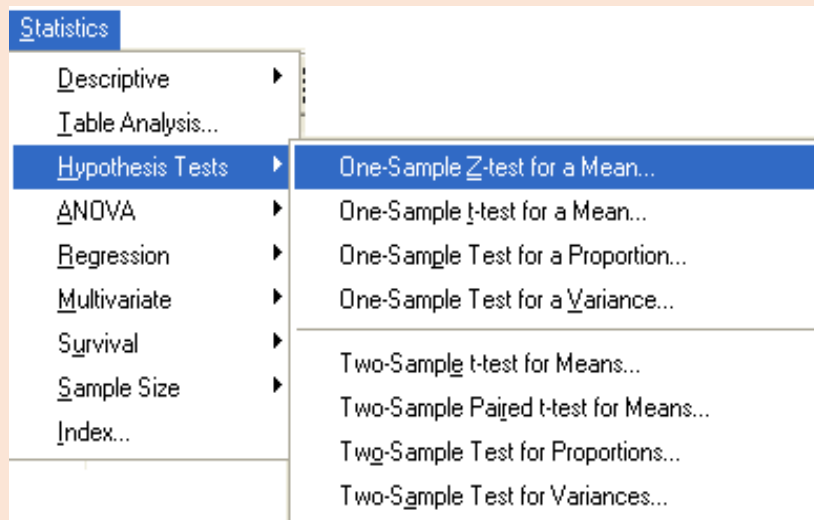
$$H_1 : \mu > 200$$

1) ادخل البيانات وأحفظها في مكتبة Work باسم FAB

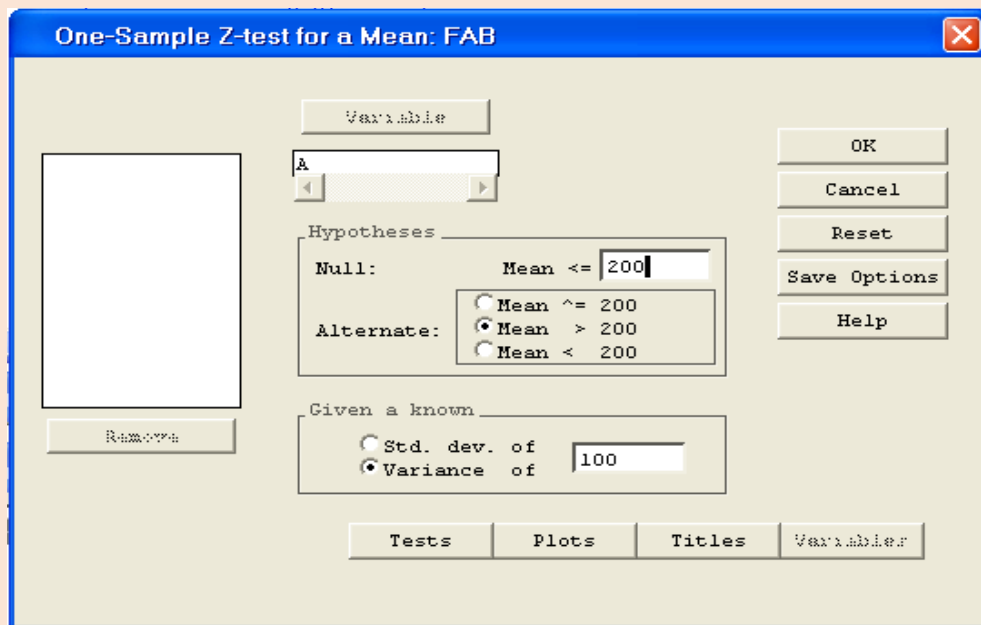
	A
1	204
2	205
3	215
4	208

2) اختر من القائمة Statistics الامر Hypothesis tests ومنه اختار

One-Sample Z-test for a mean



( يظهر لك مربع حوار حدد فيه العمود الذي يحوي البيانات و قيمة الادعاء وقيمة 3) التباين أو الانحراف المعياري ويمكنك تحديد قيمة مستوي المعنوية ، كما في الشكل التالي :



لتظهر النتائج كما في الشكل التالي ( OK: اضغط 4 )

## One Sample Z Test for a Mean

### Sample Statistics for A

N	Mean	Std. Dev.	Std. Error
---	------	-----------	------------

4	208.00	4.97	2.48
---	--------	------	------

### Hypothesis Test

Null hypothesis: Mean of A  $\leq$  200

Alternative: Mean of A  $>$  200

With a specified known variance of 100

Z Statistic	Prob > Z
-------------	----------

1.600	0.0548
-------	--------

4.97 ، وانحرافها المعياري = 208 و متوسطها = 4 ومنه نلاحظ عدد البيانات =  
2.48. والخطأ المعياري =

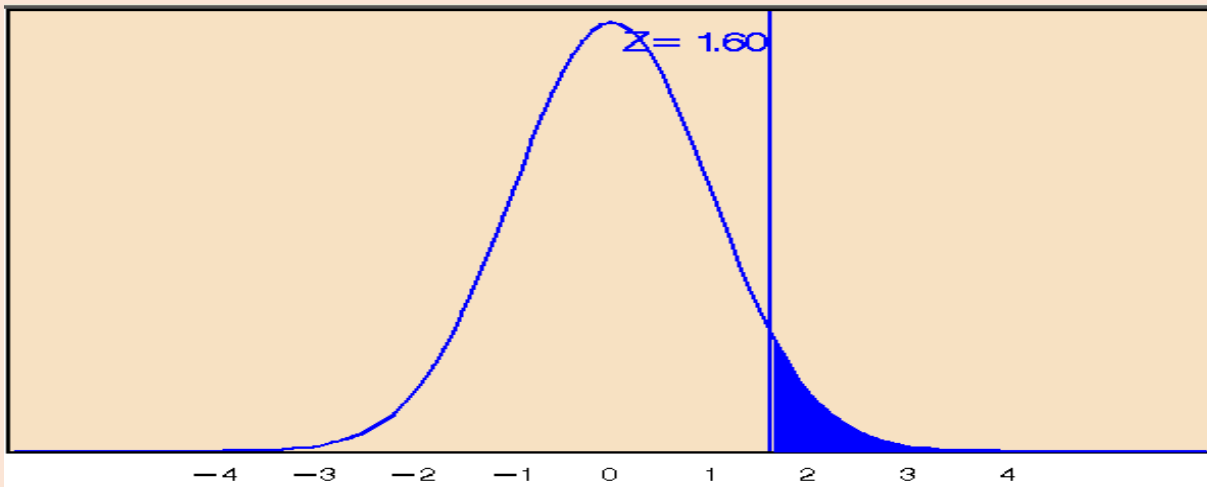
$Z = 1.6$ . وصياغة فرض عدم والفرض البديل . وقيمة الإحصاءة

( المهمة في اتخاذ القرار Prob ) وتكتب في ساس على الصورة  $P\_value$  وقيمة  
وهذا يعني : 0.0548 والتي تساوي في مثالنا

$$P\_value = 0.0548 > 0.05 = \alpha$$

ويعني ذلك قبول فرض عدم  $H_0$  ( أي لا نستطيع رفض  $H_0$  )

( Plots ) : يمكن لساس رسم منطقة الرفض والقبول . وذلك بالضغط على زر 1 ملاحظة )  
ومن ثم Normal distribution plot الموجود في مربع الحوار السابق ، واختيار  
مرتين . وسيظهر الرسم كما في الشكل التالي : OK الضغط على



( : يجب ملاحظة أن القيمة الافتراضية لمستوى المعنوي في حزمة ساس هي 2 ملاحظة )  
في مربع الحوار Tests ، ويمكننا تغييرها لأي قيمة أخرى وذلك بالضغط على زر 0.05  
ثم غير قيمة مستوى المعنوية Power Analysis السابق ، ليظهر مربع آخر نختار منه  
كما يتضح في الرسم التالي : Values إلى أي قيمة مطلوبة في مستطيل

**One-Sample Z-test for a Mean: Tests**

Confidence Intervals | Power Analysis

Select if you want to perform retrospective power analysis.

Perform power analysis

Alphas

Values: .05

Sample sizes

Values:

From:  To:  By:

OK  
Cancel  
Reset  
Help

T\* اختبار

في اختبارات الفروض حول متوسط مجتمعين طبيعيين ( مقارنة متوسطين مجتمعين مع بعضهما البعض ) وحجم العينة صغير والتباين مجهول ولكن بمعرفة تساوي تباينات أو عدم تساويهما وإمكانية اختبار الفروق بين المجتمعين ، هذا الخيار يمكنك من الحصول على قيمة تساعدك في اتخاذ القرار برفض أو قبول فرض العدم الذي يكون كالتالي :

$$H_0 : \mu_1 = \mu_2$$

ضد أحد الفروض البديلة التالية :

$$H_1 : \mu_1 > \mu_2$$

$$H_0 : \mu_1 < \mu_2$$

$$H_1 : \mu_1 \neq \mu_2$$

مثال:

فكانت مدة إضاءة كل A لإنتاج المصابيح أخذت عينة من إنتاج المصنع B و Aمصنع مصباح بالشهور كتالي :

10, 24, 32, 14, 36, 40, 41, 38

كتالي B:والمصنع

18, 9, 33, 11, 33, 1, 12, 9, 13, 17, 17

وكان تباين المصنعين غير معروف ولكن يمكن افتراض تساويهما ، هل يوجد اختلاف بين 0.05؟إنتاج المصنعين عند مستوى معنوي

الحل باستخدام ساس :

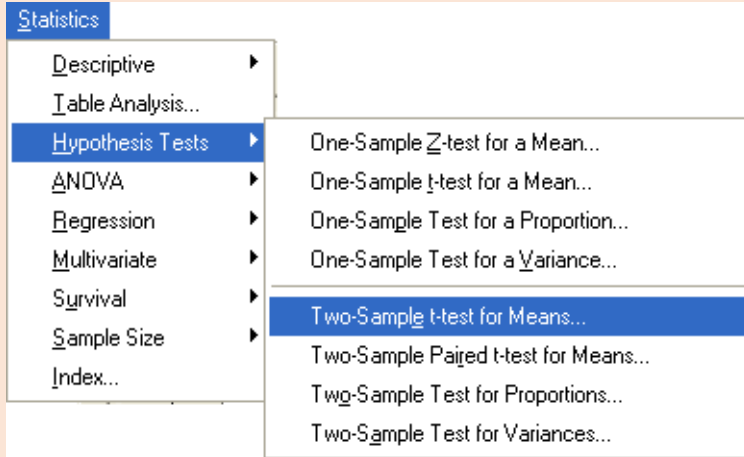
صياغة الفروض :

$$H_0 : \mu_1 = \mu_2 \quad \text{VS} \quad H_1 : \mu_1 \neq \mu_2$$

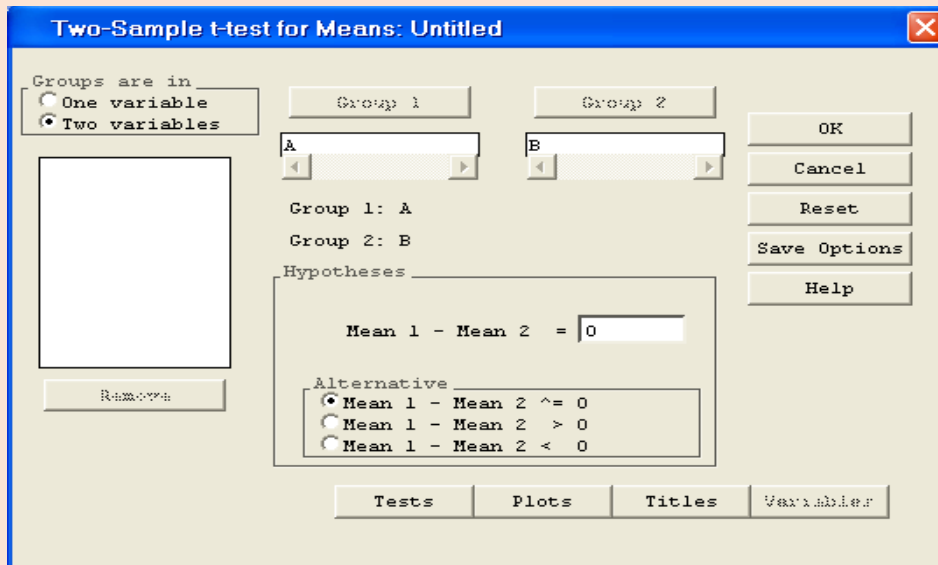
( ادخل البيانات كما في الشكل التالي :1)

	A	B
1	10	18
2	24	9
3	32	33
4	14	11
5	36	33
6	40	1
7	41	12
8	38	9
9		13
10		17
11		17

Two- ومنه اختر Hypothesis tests الأمر Statistics ) اختر من القائمة 2)  
كما في الشكل التالي : Sample t- test for means



وهو يعني أن لدينا Two variables ) يظهر لك مربع حوار اختر الخيار 3)  
واضغط B واخر Group 1 واضغط على المستطيل A . اختر B و A مجموعتين وهي  
وحدد الفرض البديل كما في الشكل التالي : Group 2 على المستطيل



لتظهر النتائج كما يلي : OK ثم أضغط

## Two Sample t-test for the Means of A and B

### Sample Statistics

Group	N	Mean	Std. Dev.	Std. Error
-------	---	------	-----------	------------

A	8	29.375	12.035	4.255
B	11	15.72727	9.7785	2.9483

### Hypothesis Test

Null hypothesis: Mean 1 – Mean 2 = 0

Alternative: Mean 1 – Mean 2  $\neq$  0

If Variances Are	t statistic	Df	Pr > t
------------------	-------------	----	--------

Equal	2.728	17	0.0143
Not Equal	2.636	13.20	0.0203

( تحليل النتائج . 4 )

A يعطي الجدول الأول بعض الإحصاءات الوصفية لكل مجموعة فعدد عناصر المجموعة وتباينها = 12.035 وانحرافها المعياري mean = 29.378 ومتوسطها N = 8 هي B. وهكذا بالنسبة للمجموعة Variance = 144.8393

( فيتضح أن Two-tail: أما الجدول الثاني حيث أن الاختبار في المثال من طرفين )

في حالة تساوي التباين :

والتي منها نتخذ القرار P-value بقيمة  $t_0 = 17$  . ودرجات حرية تساوي 2.728  
حيث أن :

$$P -value = 0.0143 < 0.05 = \alpha$$

فنقول لا نستطيع قبول  $H_0$  ( رفض  $H_0$  ) أي أن متوسطات المجتمعين مختلفة .  
أما في حالة عدم تساوي التباين :

P-value بقيمة  $t_0 = 13.20$  ودرجات حرية تساوي 2.636

والتي منها نتخذ القرار حيث أن :

$$P -value = 0.0203 < 0.05 = \alpha$$

فنقول لا نستطيع قبول  $H_0$  ( رفض  $H_0$  ) أي أن متوسطات المجتمعين مختلفة .

F\* اختبار

في اختبارات فروض حول تساوي تباين مجتمعين طبيعيين حيث يكون الفرض كتالي :

$$H_0 : \sigma_1^2 = \sigma_2^2$$

$$H_1 : \sigma_1^2 \neq \sigma_2^2$$

مثال:

في المثال السابق اختبر تساوي تباين المجتمعين .



الحل باستخدام ساس :

( صياغة الفروض : 1 )

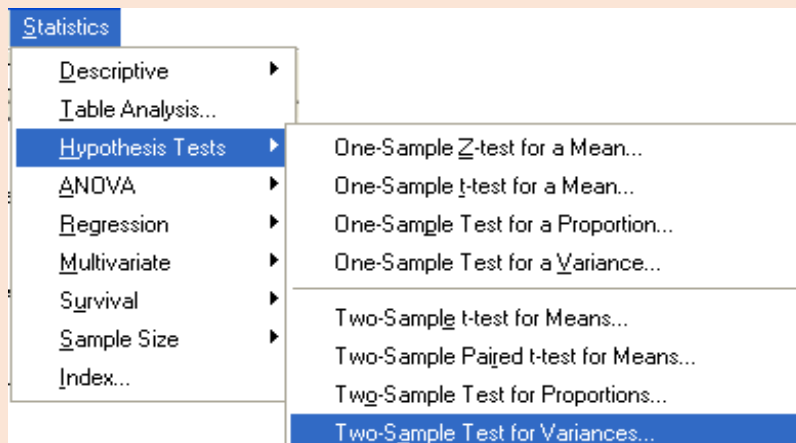
$$H_0 : \sigma_1^2 = \sigma_2^2$$

$$H_1 : \sigma_1^2 \neq \sigma_2^2$$

( ادخل البيانات كما في الشكل التالي:2)

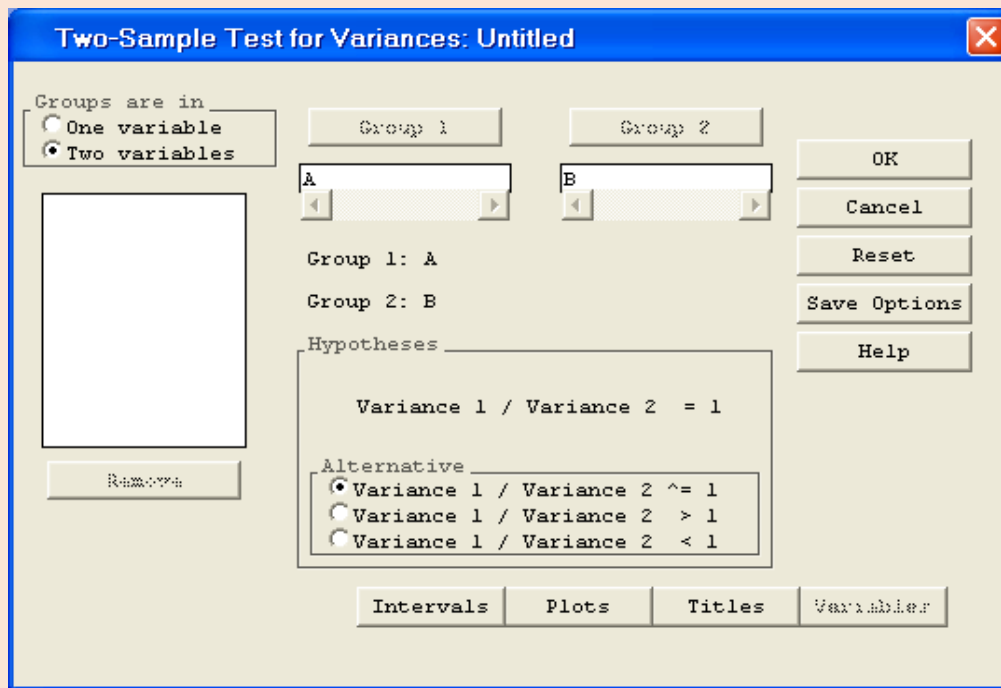
	A	B
1	10	18
2	24	9
3	32	33
4	14	11
5	36	33
6	40	1
7	41	12
8	38	9
9		13
10		17
11		17

Two- ومنه اختر Hypothesis tests الأمر Statistics ( اختر من القائمة 2)  
Sample test for variances : كما في الشكل التالي :



(3)

وهو يعني أن لدينا **Two variables** ( يظهر لك مربع حوار اختر الخيار 3 )  
واضغط B واختر **Group 1** واضغط على المستطيل A . اختر B و A مجموعتين وهي  
وحدد الفرض البديل كما في الشكل التالي: **Group 2** على المستطيل



لتظهر النتائج كما يلي : OK ثم اضغط

### Sample Statistics

Group      N      Mean      Std. Dev.      Variance

---

A            8      29.375      12.035      144.8393

B            11   15.72727      9.7785    95.61818

### Hypothesis Test

Null hypothesis:    Variance 1 / Variance 2 = 1

Alternative:        Variance 1 / Variance 2  $\neq$  1

– Degrees of Freedom –

F	Numer.	Denom.	Pr > F
---	--------	--------	--------

---

1.51	7	10	0.5320
------	---	----	--------

(4) تحليل النتائج.

يعطي الجدول الأول بعض الإحصاءات الوصفية لكل مجموعة مثل

عدد عناصر المجموعة A = N

متوسطها = Mean = 29.378

انحرافها المعياري = 12.035

تباينها = Variance = 144.8393

وهكذا بالنسبة للمجموعة B

أما الجدول الثاني حيث أن الاختبار في المثال من طرفين Two-tail    فيتضح ان

$$F_0 = 1.51$$

ودرجات الحرية

$$v_1 = 7, v_2 = 10$$

وقيمة P-value والتي منها نتخذ القرار حيث ان

$$P - value = 0.5320 > 0.05 = \alpha$$

فنقول لا نستطيع رفض  $H_0$  ( قبول  $H_0$  ) أي أن تباينات المجتمعين متساوية .

\* اختبار مربع كاي Chi test

يستخدم اختبار كاي في اختبارات الفروض للاستقلال ( دراسة العلاقة بين عاملين أو أكثر ) واختبارات التجانس . فعند اختبارات الفروض للاستقلال تكون صياغة الفروض كما يلي :

$H_0$  البيانات مستقلة .

$H_1$  البيانات غير مستقلة .

مثال:

اخذت عينة من 91 طالب تم تقسيمهم حسب اوزانهم وتقديراتهم كما يلي:

المجموع	متوسط	جيد	ممتاز	
35	10	11	14	نحيف
42	16	16	10	متوسط الوزن
14	7	4	3	بدين
91	33	31	27	المجموع

استخدم مربع كاي لاختبار الفرض القائل أن التحصيل العلمي مستقل عن الوزن بمستوى معنوية  $\alpha = 0.05$  .

الحل باستخدام ساس :

$H_0$  مقدرة الطالب في التحصيل العلمي مستقلة عن وزنه

$H_1$  مقدرة الطالب في التحصيل العلمي غير مستقلة عن وزنه.

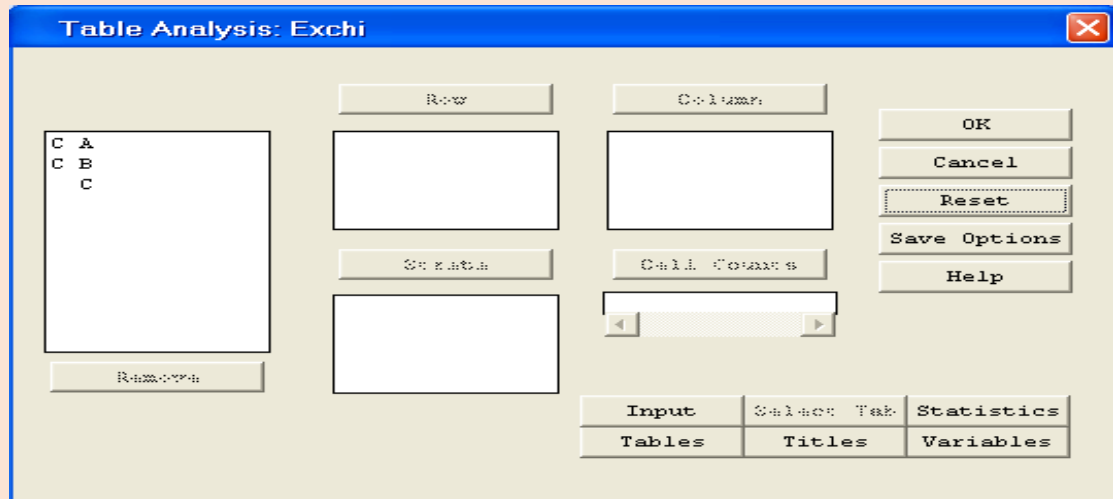
(1) ندخل البيانات بحيث يكون العمود A للتقدير والعمود B للوزن والعمود C للبيانات

ويتم ادخال التقديرات ممتاز Excellent وجيد Good ومتوسط Mean ونحيف Thin

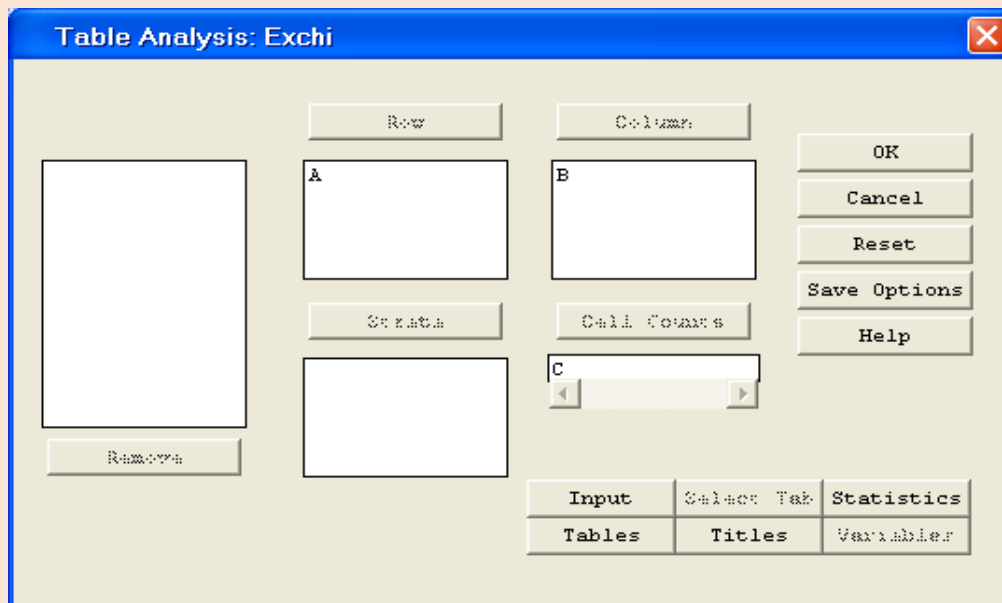
ومتوسط الوزن M-Weigth وبيدين obese كما في الشكل:

	A	B	C
1	Excellen	Thin	14
2	Excellen	M-Weight	10
3	Excellen	Obese	3
4	Good	Thin	11
5	Good	M-Weight	16
6	Good	Obese	4
7	Mean	Thin	10
8	Mean	M-Weight	16
9	Mean	Obese	7

(2) من القائمة Statistics نختار الامر Table Analysis ليظهر لنا المربع التالي:

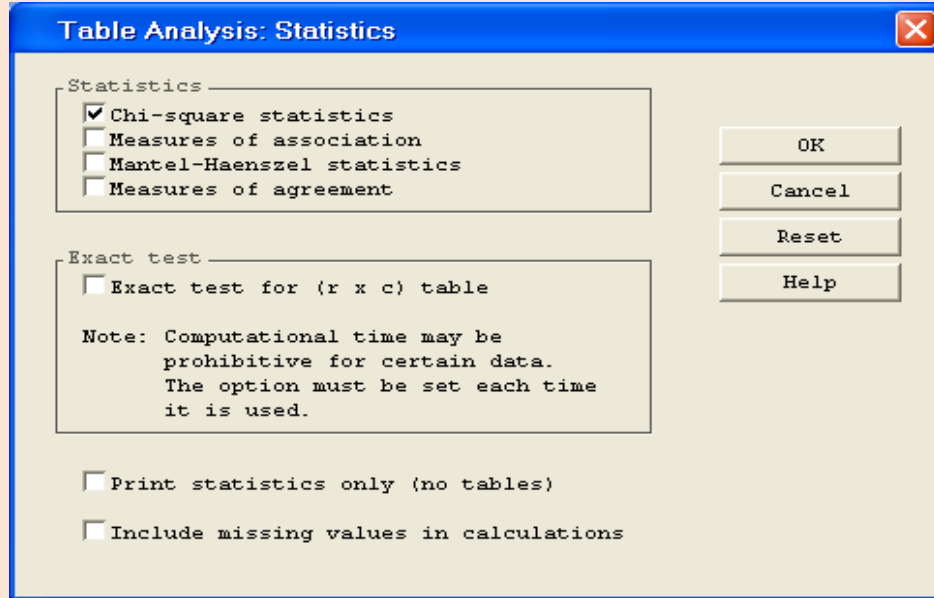


(3) نحدد A ونضغط على Row ليننتقل تحته، ونحدد B ونضغط على Column ليننتقل تحته، ونحدد C ونضغط على Cell Counts ليننتقل تحته كما في الشكل:



(4) نضغط على Statistics وسيظهر مربع آخر يحتوي على بعض الخيارات، ومنه نختار

## Chi-Square Statistics كما في الشكل التالي:



ثم نضغط OK لنعود الى المربع السابق ونضغط OK لتظهر النتائج التالية:

### Statistics for Table of A by B

Statistic	DF	Value	Prob
<i>ffffffffffffffffffffffffffffffffffffffff</i>			
Chi-Square	4	3.7995	0.4338
Likelihood Ratio Chi-Square	4	3.7135	0.4462
Mantel-Haenszel Chi-Square	1	1.7967	0.1801
Phi Coefficient			0.2043
Contingency Coefficient			0.2002
Cramer's V			0.1445

يظهر لنا من هذا الجدول أن قيمة كاي تربيع تساوي 3.7995 بـدرجات حرية 4،

كما ان قيمة P-value تساوى 0.4338 التي سنقارنها مع مستوى المعنوية

$\alpha$  نجد أن:

$$P -value = 0.4338 > 0.05 = \alpha$$

وهذا يعني أننا لا نستطيع رفض فرض العدم  $H_0$  أي أن مقدرة الطالب مستقلة عن وزنه.



## الفصل الرابع

### التحليل الإحصائي باستخدام قوائم ساس والبرمجة

#### \* الإحصاء الوصفي Descriptive Statistics

يعرف الإحصاء الوصفي بأنه مجموعة من مقاييس النزعة المركزية ومقاييس التشتت وبعض المقاييس الأخرى التي يتم حسابها تحت مسمى الوصف الإحصائي حيث سبق شرح هذه المقاييس في فصل سابق وهي كالتالي :

Mean	الوسط الحسابي
Standard Error	الخطأ المعياري
Median	الوسيط
Mode	المنوال
Standard Deviation	الانحراف المعياري
Variance	التباين
Kurtosis	التفرطح
Skewness	الالتواء
Rang	المدى
Maximum	القيمة العظمى
Minimum	القيمة الصغرى
Sum Observations	مجموع المشاهدات
Sum Weights	عدد الخلايا
Quartiles	الربيعات

مثال

لتكن لدينا درجات الطلاب في إحدى المواد وهي كالتالي :

68, 78, 87, 90, 80, 80, 98, 64, 77

ناقش المقاييس الوصفية لهذه الدرجات .

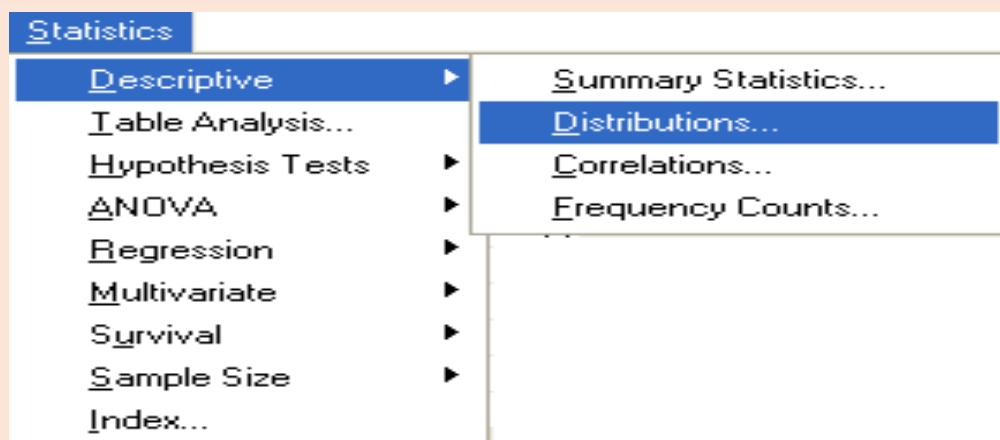
الحل باستخدام ساس :

(1) ادخل البيانات كما في الشكل التالي

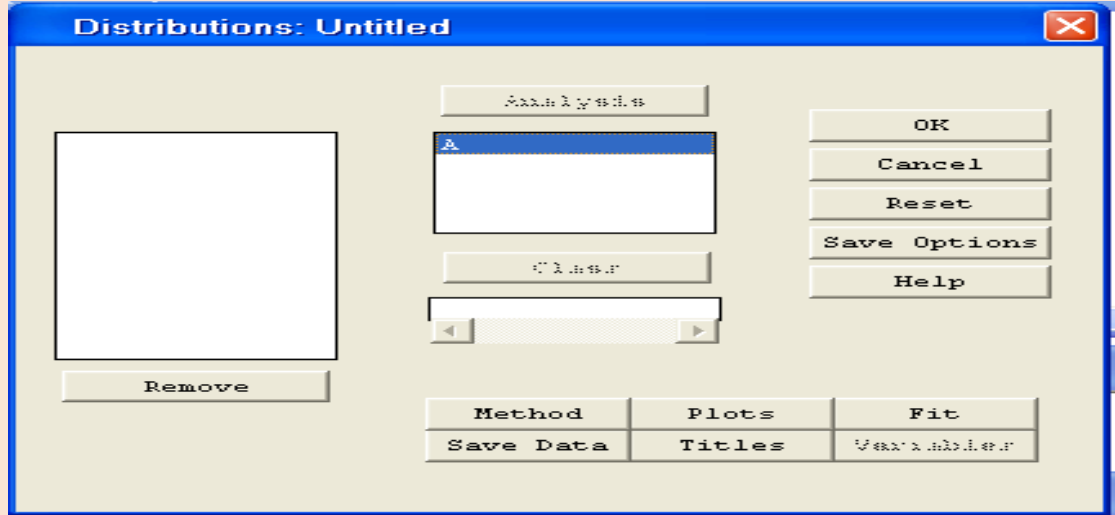
	الدرجة
1	68
2	78
3	87
4	90
5	80
6	80
7	98
8	64
9	77

( 2 ) اختر من القائمة الامر **Statistics** ومنه اختر الامر **Descriptive** ومنه اختر

**Distributions**. كما في الشكل التالي:



(3) يظهر مربع حوار نضغط أو نحدد اسم العمود A ثم نضغط على المستطيل Analysis لينتقل العمود A داخل مربع Analysis كما يلي:



(4) نضغط على OK لتظهر النتائج التالية

### The UNIVARIATE Procedure

Variable: A

#### Moments

<b>N</b>	<b>9</b>	<b>Sum Weights</b>	<b>9</b>
<b>Mean</b>	<b>80.2222222</b>	<b>Sum Observations</b>	<b>722</b>
<b>Std Deviation</b>	<b>10.5211427</b>	<b>Variance</b>	<b>110.694444</b>
<b>Skewness</b>	<b>0.10012018</b>	<b>Kurtosis</b>	<b>-0.1373606</b>
<b>Uncorrected SS</b>	<b>58806</b>	<b>Corrected SS</b>	<b>885.555556</b>
<b>Coeff Variation</b>	<b>13.1149979</b>	<b>Std Error Mean</b>	<b>3.50704758</b>

#### Basic Statistical Measures

	Location		Variability	
<b>Mean</b>	<b>80.22222</b>	<b>Std Deviation</b>	<b>10.52114</b>	

Median	80.00000	Variance	110.69444
Mode	80.00000	Range	34.00000
		Interquartile Range	10.00000

**Tests for Location:  $\mu_0=0$**

Test	-Statistic-	-----p Value-----
Student's t	t 22.87457	Pr >  t  <.0001
Sign	M 4.5	Pr >=  M  0.0039
Signed Rank	S 22.5	Pr >=  S  0.0039

**Quantiles (Definition 5)**

Quantile	Estimate
100% Max	98
99%	98
95%	98
90%	98
75% Q3	87
50% Median	80
25% Q1	77
10%	64
5%	64
1%	64
0% Min	64

ومن النتائج السابقة نجد أن :

الوسط الحسابي = 80.22 والوسيط = 80

وبما أن الوسط الحسابي والوسيط متقاربين ، لذلك فإن توزيع البيانات يكون إلى حد ما متماثل .

ومن النتائج :

المنوال=80 أي القيمة التي لها أكبر تكرار في البيانات تساوي 80  
التباين=110.6944 والانحراف المعياري = 10.52114 وهذا يبين درجة تجانس البيانات

$$\frac{S}{\sqrt{n}} = 3.507 = \text{والخطأ المعياري}$$

وهو مقياس لتشتت بيانات العينة بالنسبة للمجتمع

والتفريط=-0.137

وهذا يعني أن منحنى التوزيع يقل معامل تفريطه عن معامل تفريط التوزيع الطبيعي

بمقدار 0.137

الالتواء=0.1 وهذا يعني أن منحنى التوزيع موجب الالتواء .

القيمة العظمى = 98 أي أكبر قيمة في البيانات

القيمة الصغرى =64 أي أصغر قيمة في البيانات

المدى = 64-98 = 34 وهو الفرق بين أكبر وأصغر قيمة في البيانات

مجموع المشاهدات = 722 والرابع الأول=77 والرابع الثاني = الوسيط =80

والرابع الثالث=87

## \* تحليل الانحدار Regression

\*\* الانحدار البسيط :

يمكن تمثيل معادلة الانحدار من الدرجة الأولى على الصورة التالية :

$$Y = a + bX$$

حيث :

المتغير التابع الذي نريد تقديره ( التنبؤ ) Y

المتغير المستقل X

مقادير ثابتة يمكن حسابها من مشاهدات العينة . a , b

وعند معرفة قيم b , a يمكن التنبؤ بقيم Y عندما تأخذ X قيما معينة

وتستخدم طريقة المربعات الصغرى لإيجاد قيم a,b

مثال

اوجد معادلة انحدار نمو النبات Y على عمر النبات X الموضحة في

X	1	2	3	4	5
Y	2	3	5	6	8

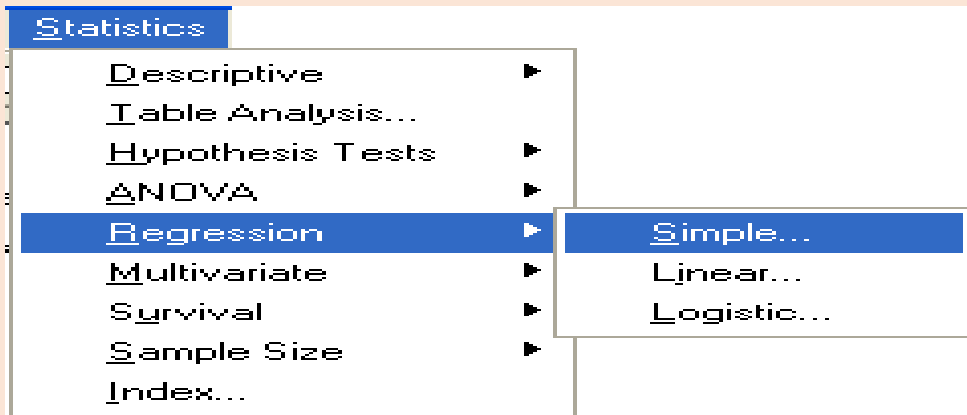
الحل باستخدام ساس :

(1) ادخل البيانات كما في الشكل التالي

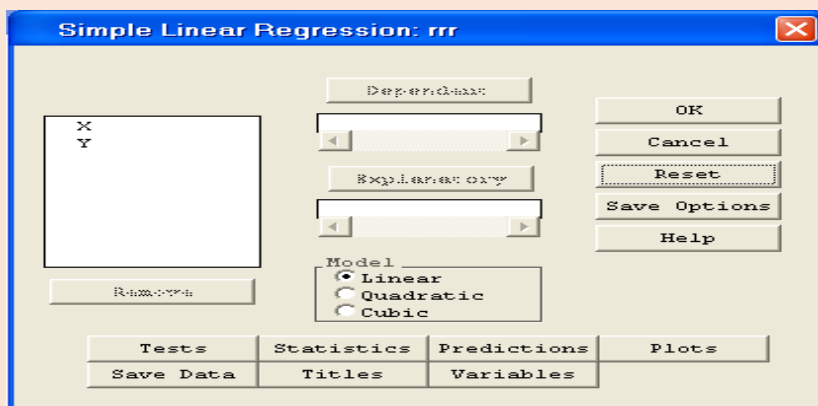
	X	Y
1	1	2
2	2	3
3	3	5
4	4	6
5	5	8

(2) من القائمة **Statistics** اختر الأمر **Regression** ومنه اختر الامر الفرعى

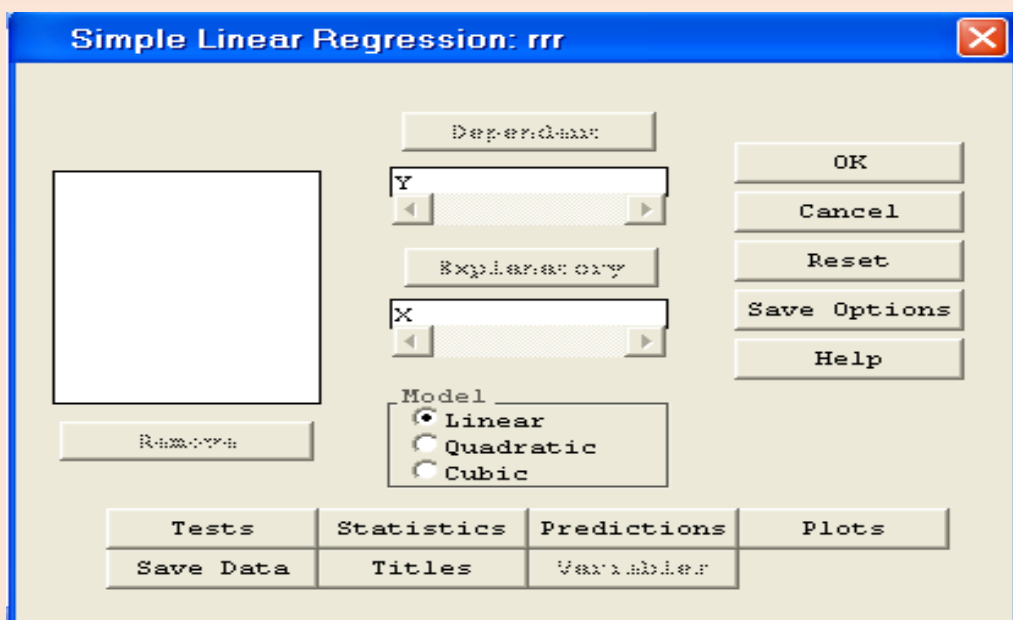
**Simple....** كما في الشكل:



يظهر لك مربع حوار كما في الشكل التالي :



4) في المربع الحوارى السابق حدد  $Y$  واضغط على المستطيل **Dependency** وهو الذي نريد تقديره وهو دائما غير مستقل. ثم حدد  $X$  وهو المتغير المستقل واضغط على المستطيل **Explanatory** كما في الشكل.



ثم نضغط **OK** لتظهر النتائج التالية:



**The REG Procedure**

**Model: MODEL1**

**Dependent Variable: Y**

**Analysis of Variance**

<b>Sum of</b>	<b>Mean</b>	<b>Source</b>	<b>DF</b>	<b>Squares</b>		
<b>Square</b>	<b>F Value</b>	<b>Pr &gt; F</b>				
<b>Model</b>	<b>1</b>	<b>22.50000</b>	<b>22.50000</b>	<b>225.00</b>	<b>0.0006</b>	
<b>Error</b>	<b>3</b>	<b>0.30000</b>	<b>0.10000</b>			
<b>Corrected Total</b>	<b>4</b>	<b>22.80000</b>				
<b>Root MSE</b>		<b>0.31623</b>	<b>R-Square</b>	<b>0.9868</b>		
<b>Dependent Mean</b>		<b>4.80000</b>	<b>Adj R-Sq</b>	<b>0.9825</b>		
<b>Coeff Var</b>		<b>6.58808</b>				

**Parameter Estimates**

<b>Variable</b>	<b>DF</b>	<b>Parameter Estimate</b>	<b>Standard Error</b>	<b>t Value</b>	<b>Pr &gt;  t </b>
<b>Intercept</b>	<b>1</b>	<b>0.30000</b>	<b>0.33166</b>	<b>0.90</b>	<b>0.4324</b>
<b>X</b>	<b>1</b>	<b>1.50000</b>			

0.10000      15.00      0.0006

من الجدول الأول

نحلل النتائج كما يلي : Analysis of Variance من جدول تحليل التباين

• مجموع المربعات ( SS ) :

( 22.5 ) هو Model نموذج انحدار )

( 0.3 ) هو Error البواقي أو الخطأ )

( 22.8 ) هو Corrected Total المجموع )

• درجات الحرية ( df ) :

( 1 ) يساوي Model نموذج انحدار )

( 3 ) يساوي Error البواقي أو الخطأ )

( 4 ) يساوي Corrected Total المجموع )

• متوسط المربعات ( MS ) :

( 22.5 ) هو Model نموذج انحدار )

( 0.1 ) هو Error البواقي أو الخطأ )

• أدوات الاختبار :

0.0006 و تساوي P-value أو Significance F 225 المحسوبة هي F قيمة

من الجدول الثاني

سنأخذ منه بعض الاحصاءات المهمة مثل :

• الخطأ المعياري ( Root MSE ) = 0.31623

• معامل التحديد R-Square أو  $R^2 = 0.9868$  ويمكن إيجاد معامل الارتباط المتعدد

R وذلك بأخذ جذر معامل التحديد كما يلي :

$$R = \sqrt{R^2} = \sqrt{0.9868} = 0.9934$$

- معامل التحديد المعدل ( Adj R-Sq ) أو  $R^2$  ويساوي 0.9825 من المقاييس السابقة نجد أن النموذج ملائم للبيانات المعطاة حيث أن معامل التحديد وهذا يدل أن نمو النباتات يعتمد اعتماداً كبيراً على عمر النبات ، وهذا  $R^2 = 0.9868$  الاسلوب هو أحد أساليب الحكم على معنوية النموذج .  
من الجدول الثالث

( نجد أن معاملات معادلة الانحدار هي : Parameter Estimates أما من الجدول )  
 $b = 1.5$  وهو  $X$  ومعامل  $a = 0.3$  الحد الثابت

لذا يمكن كتابة معادلة الانحدار بالشكل التالي :

$$Y = 0.3 + 1.5 X$$

$X$  عند أي قيم للمتغير  $Y$  وهذه المعادلة تمكننا من التنبؤ لأي قيمة للمتغير .  
هو Error Standard وأيضاً نجد أن الخطأ المعياري لتقدير المعاملات

$$S.E(\hat{a}) = 0.33166$$

$$S.E(\hat{b}) = 0.1$$

التي تستخدم في اختبار الفرضيات حول معاملات الانحدار هي  $t$  ونجد أن قيم الاحصاءات كالتالي :

$$t = 0.90 \Rightarrow P\text{-value} = 0.4324$$

$$t = 15 \Rightarrow P\text{-value} = 0.0006$$

وعدم معنوية الحد الثابت في النموذج المقدر  $X$ . وهذا يدل على أن معنوية معامل  
\*ولإيجاد فترات الثقة نقوم بالخطوات التالية  
\*\* الانحدار المتعدد :

في كثير من الدراسات الإحصائية نضطر إلى استخدام أكثر من متغير للتنبؤ بقيمة المتغير Y التابع يسمى هذا الأسلوب بأسلوب الانحدار المتعدد وسنقتصر على أسلوب لتنبؤ قيمة من كل من المتغيرين  $X_2, X_1$  من العلاقة الخطية التالية :

$$\hat{Y} = a_0 + a_1X_1 + a_2X_2 + \dots + a_kX_k$$

وبحل هذه المعادلة نحصل على قيم  $a_0, a_1, a_2$  وتوضيح ذلك نأخذ المثال التالي :

مثال

يمثل الجدول التالي مساحة وعمر وسعر 20 بيتا في بلد ما، اوجد معادلة الانحدار للتنبؤ عن سعر Y من كل من المتغيرين  $X_1, X_2$  من العلاقة الخطية التالية:

$$\hat{Y} = a_0 + a_1X_1 + a_2X_2$$

الى  $a_k X_k$

السعر بالآلاف الريالات	العمر بالسنوات $X_2$	المساحة بالآلاف الأقدام المربعة $X_1$
------------------------	-------------------------	------------------------------------------

32	30	1.8
24	33	1.0
27	25	1.7
47	12	2.8
35	26	2.2
17	25	0.8
52	28	3.6
20	29	1.1
38	25	2.0
45	2	2.6
44	30	2.3
19	23	0.9
	12	1.2
<b>690</b>	<b>430</b>	<b>40 المجموع =</b>

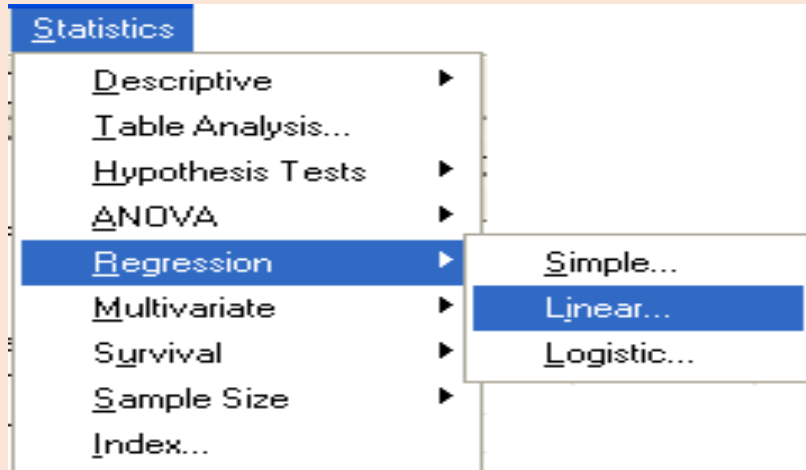
الحل باستخدام ساس :

(1) ادخل البيانات في حزمة ساس من 1 الى 20 كما يلي:

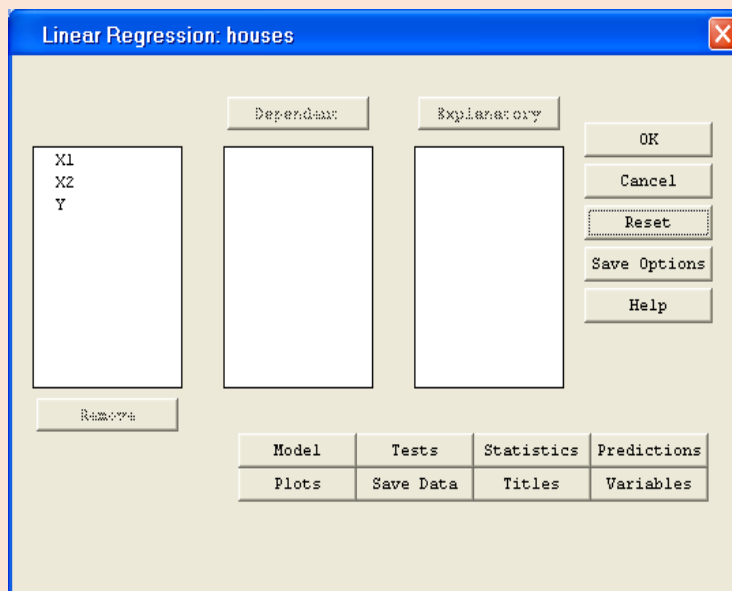
	X1	X2	Y
2	1	33	24
3	1.7	25	27
4	2.8	12	47
5	2.2	26	35
6	0.8	25	17
7	3.6	28	52
8	1.1	29	20
9	2	25	38
10	2.6	2	45
11	2.3	30	44

(2) من القائمة Statistic اختر الامر Regression ومنه اختار الامر الفرعى

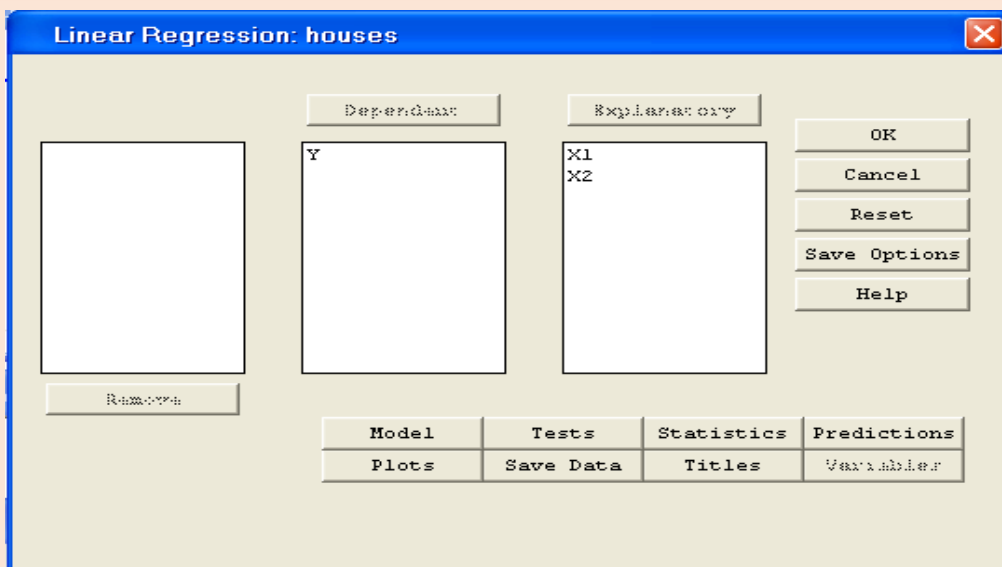
Linear.... كما يلي:



سيظهر لك مربع حوار كما في الشكل التالي :



4) في المربع الحوارى السابق حدد Y واضغط على المستطيل Dependent وهو الذي نريد تقديره وهو دائما غير مستقل. ثم حدد X1, X2 وهما المتغيران المستقلان واضغط على المستطيل Explanatory كما في الشكل.



ثم نضغط على OK ثم OK لتظهر النتائج:

### Analysis of Variance

Sum of	Mean	Source	DF	Squares	Square	F Value
Pr	>		F			
Model	2			2261.42033		
				1130.71016	172.27	<.0001
Error	17			111.57967		
						6.56351

Corrected Total 19 2373.00000  
 Root MSE 2.56193 R-Square 0.9530  
 Dependent Mean 34.50000 Adj R-Sq 0.9474  
 Coeff Var 7.42590

### Parameter Estimates

Variable	DF	Parameter Estimate	Standard Error	t Value	Pr >  t	95% Confidence Limits
Intercept	1	10.04634	2.09960	4.78	0.0002	5.61656 14.47611
X1	1	12.83445	0.69919	18.36	<.0001	11.35929 14.30962
X2	1	-0.05652	0.06164	-0.92	0.3719	-0.18656 0.07352

من الجدول الأول

نحلل النتائج كما يلي : Analysis of Variance من جدول تحليل التباين

• مجموع المربعات ( SS ) :

( 2261.42 ) هو Model نموذج انحدار )

( 111.579 ) هو Error البواقي أو الخطأ )

( 2373 ) هو Corrected Total المجموع )

• درجات الحرية ( df ) :



( 2 ) يساوي Model نموذج انحدار )

( 17 ) يساوي Error البواقي أو الخطأ )

( 19 ) يساوي Corrected Total المجموع )

• متوسط المربعات ( MS ) :

( 1130.71 ) هو Model نموذج انحدار )

( 6.5635 ) هو Error البواقي أو الخطأ )

• أدوات الاختبار :

و تساوي P-value أو Significance F 172.27 المحسوبة هي F قيمة  
0.0001

من الجدول الثاني

سنأخذ منه بعض الاحصاءات المهمة مثل :

• الخطأ المعياري ( Root MSE ) = 2.56193

• معامل التحديد R-Square أو  $R^2 = 0.9530$  ويمكن إيجاد معامل الارتباط المتعدد R وذلك بأخذ جذر معامل التحديد كما يلي :

$$R = \sqrt{R^2} = \sqrt{0.9530} = 0.9762$$

• معامل التحديد المعدل ( Adj R-Sq ) أو  $R^2$  ويساوي 0.9474

من المقاييس السابقة نجد أن النموذج ملائم للبيانات المعطاة حيث أن معامل التحديد  $R^2 = 0.9530$  .  
X2 وعلى عمره X1 يعتمد على مساحة Y وهذا يدل أن سعر البيت

من الجدول الثالث

( نجد أن معاملات معادلة الانحدار هي : **Parameter Estimates** أما من الجدول )

$$\alpha_0 = 12.83 , \alpha_1 = -0.056 , \alpha_2 = 10.04634 \text{ الحد الثابت}$$

لذا يمكن كتابة معادلة الانحدار بالشكل التالي :

$$\hat{Y} = 10.05 + 12.83X_1 - 0.056X_2$$

X . عند أي قيم للمتغير Y وهذه المعادلة تمكننا من التنبؤ لأي قيمة للمتغير

هو **Error Standard** \* وأيضاً نجد أن الخطأ المعياري لتقدير المعاملات

$$S.E(\hat{\alpha}_0) = 2.0996$$

$$S.E(\hat{\alpha}_1) = 0.69919$$

$$S.E(\hat{\alpha}_2) = 0.06164$$

التي تستخدم في اختبار الفرضيات حول معاملات الانحدار هي t ونجد أن قيم الاحصاءات كالتالي :

$$\text{الحد الثابت} \quad t = 4.78 \Rightarrow P\text{-value} = 0.0002$$

$$\text{معامل } X_1 \quad t = 18.36 \Rightarrow P\text{-value} = 0.0001$$

$$\text{معامل } X_2 \quad t = -0.92 \Rightarrow P\text{-value} = 0.3719$$

وعدم معنوية الحد الثابت في النموذج المقدر . وهذا يدل على أن معنوية معامل

فترات ثقة لمعاملات نموذج الانحدار هي : 95% أما

- للحد الثابت ( 5.61656 , 14.47611 )
- لمعامل X1 ( 11.35929 , 14.30962 )
- لمعامل X2 ( -0.18656 , 0.07352 )

### \*\* تحليل الارتباط Correlation

للارتباط مقاييس مختلفة تحدد مقدار الارتباط بين ظاهرتين منها معامل الارتباط لبيرسون ، معامل الارتباط لسبيرمان ، معامل الاقتران ، ومعامل التوفيق .

وتكون قيمة معامل الارتباط ما بين  $-1 < r < 1$

مثال

أوجد معامل الارتباط بين إنتاج شهري Y والتكلفة الشهرية X والربح الشهري X2

لمجموعة مكونة من سبع الآت لمصنع معين حسب البيانات التالية:

Y	8	10	12	12	13	15	20
X	8	9	12	10	10	13	19
X2	9	12	10	10	11	13	17

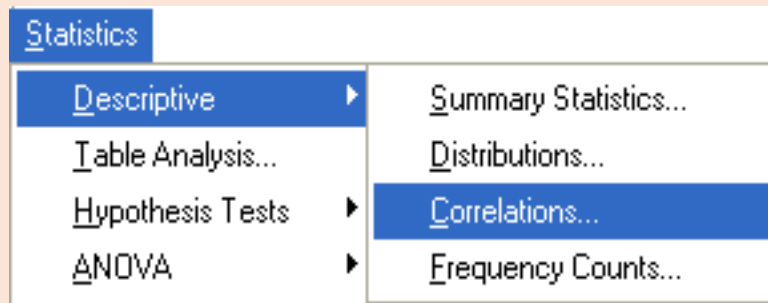
الحل باستخدام ساس :

(1) ادخل البيانات في حزمة ساس كما في الشكل التالي

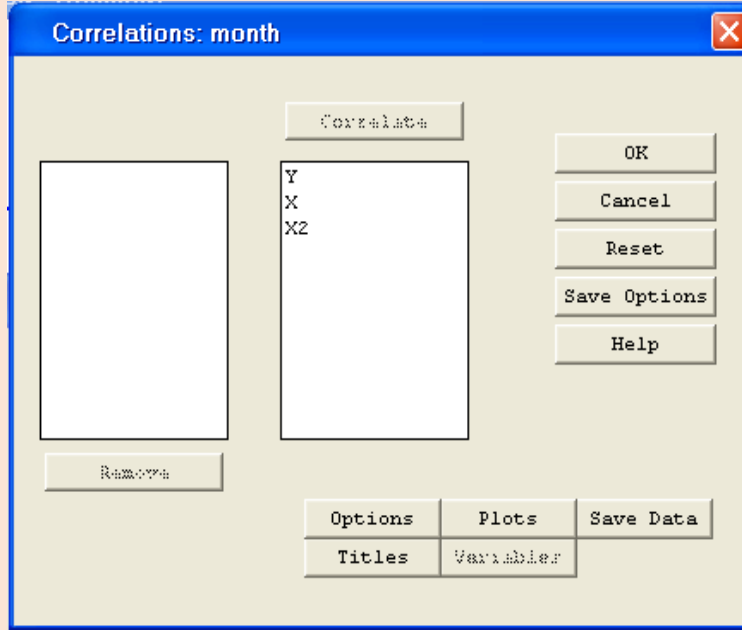
	Y	X	X2
1	8	8	9
2	10	9	12
3	12	12	10
4	12	10	10
5	13	10	11
6	15	13	13
7	20	19	17

(2) اختار من القائمة **Statistics** الامر **Descriptive** ومنه اختار **Correlations**

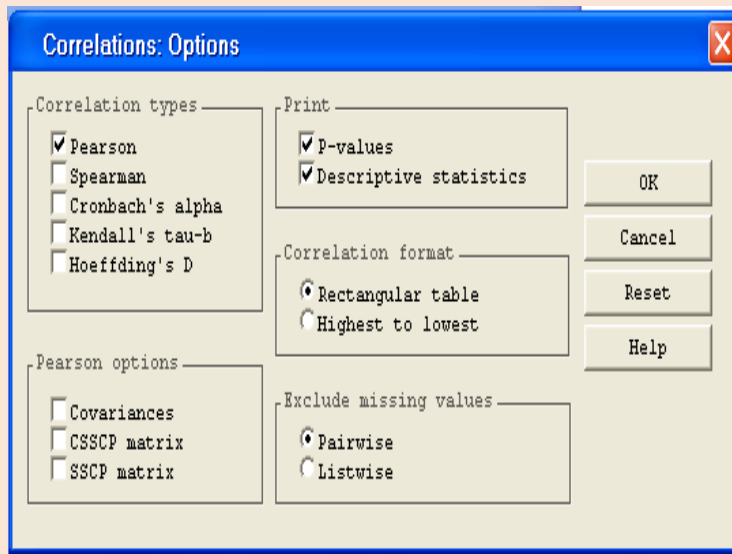
كما في الشكل التالي:



(2) يظهر مربع حوارى، قم بنقل الأعمدة  $X_2, X, Y$  الى المستطيل **Correlate** كما يلى:



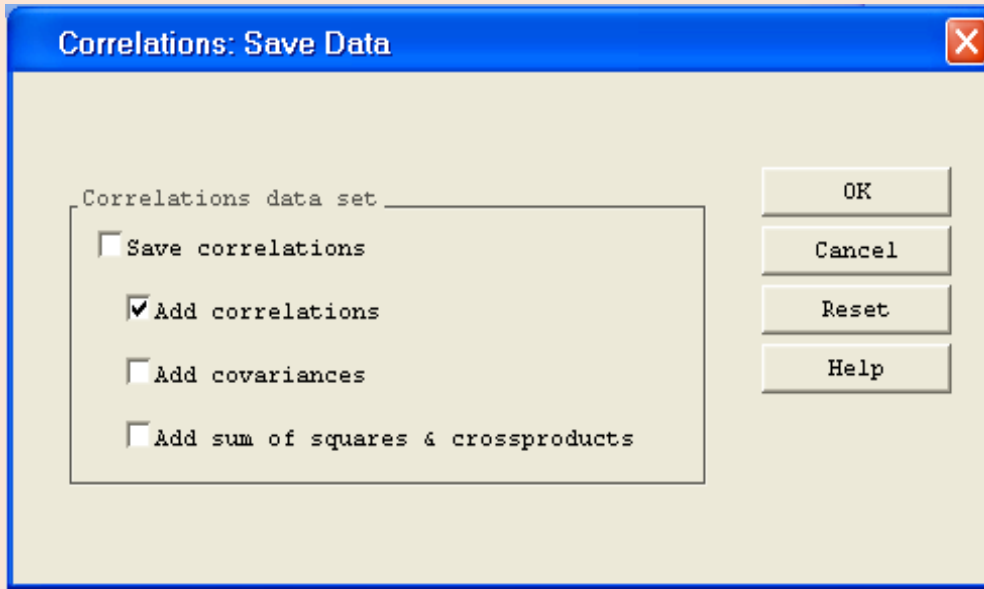
3) من مربع الحوار السابق اضغط على **Options** ليظهر لك مربع آخر نختار فيه نوع الارتباط ( بيرسون ، سبيرمان وغيرها ) ، وسنختار في مثالنا هذا معامل بيرسون فقط وبعض الخيارات كما في الشكل التالي :



ثم نضغط على OK لنعود الى مربع الحوار السابق

(4) من مربع الحوار السابق اضغط على زر Save Data ويظهر لك مربع آخر

وفيه اختر الخيار Add orrelations كما يلي:



ثم اضغط OK لنعود الى مربع الحوار السابق وفيه نضغط OK لتظهر النتائج:

**Pearson Correlation Coefficients, N = 7**

**Prob > |r| under H0: Rho=0**

	Y	X	X2
Y	1.00000	0.95729	0.89688
X		0.0007	0.0062
X2	0.95729	1.00000	0.89209
		0.0007	0.0069
Y	0.89688	0.89209	1.00000
X		0.0062	0.0069

تحليل النتائج :

وجد ان الارتباط بين الإنتاج Y والتكلفة X ارتباط قوى ويساوى 96%

وان الارتباط بين التكلفة X والربح X2 ارتباط قوى ويساوى 89%

والارتباط بين الربح X2 والإنتاج Y ارتباط قوى ويساوى 90% وجميعها طردى (موجب)

قوى.

## \*\* اختبارات الفرضيات Testing Hypothesis

يحاول الباحث في كثير من الأحيان اتخاذ قرار بشأن خواص توزيع مجتمع ما ، وذلك بناء على بيانات عينة عشوائية اختيرت من المجتمع نفسه ، نضع عادة فروض عن خواص المجتمع ونختبر هذا الفروض بناء على عينة عشوائية نختارها من المجتمع تحت الدراسة .

\*\*\* اختبار متوسطي مجتمعين طبيعيين مجهولي التباين:

\*\*\*\* بفرض عدم تساوي تبايني المجتمعين :

### T-Test: Two-sample Assuming Unequal Variances

عندما تكون التباينات غير معلومة ولا نعرف تساويهما في هذه الحالة يتم حساب تقدير تباين مشترك للمجتمعين كما يلي :

$$S^2 = \frac{S_1^2}{n_1} + \frac{S_2^2}{n_2}$$

ثم يجرى اختبار تساوي متوسطي المجتمعين باستخدام فرض عدم كتالي :

$$H_0 : \mu_1 = \mu_2$$

ضد أحد الفروض البديلة :

$$H_1 : \mu_1 > \mu_2 , \quad H_1 : \mu_1 < \mu_2 , \quad H_1 : \mu_1 \neq \mu_2$$

وفي حالة العينات الصغيرة يتم حساب  $t_0$

مع قيمة  $t$  الجدولية حيث نرفض فرض عدم اذا كانت

$$|t_0| > t$$



كذلك يمكنك اتخاذ قرار بمقارنة مستوى المعنوية مع قيمة P-value.

مثال

في دراسة للمقارنة بين السرعات الحرارية الناتجة لنوعين من الفحم المنتج من منجمين مختلفين كانت النتائج التالية بملايين السرعات الحرارية هي :

المنجم الأول:

7930, 7860, 8380, 8230, 8400

المنجم الثاني :

7660, 8070, 7720, 7690, 7510

اختبر الفرض القائل أن المنجمان لهما السرعات الحرارية نفسها عند مستوى معنوية

0.05

الحل باستخدام ساس :

(1) فرض العدم والفرض البديل كتالي

$$H_0 : \mu_1 = \mu_2$$

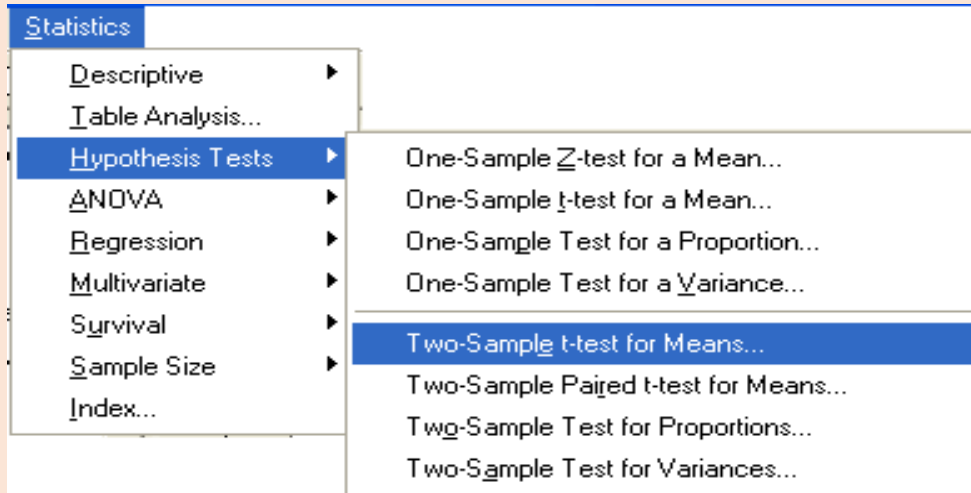
$$H_1 : \mu_1 \neq \mu_2$$

(2) ندخل البيانات في حزمة ساس كما في الشكل التالي

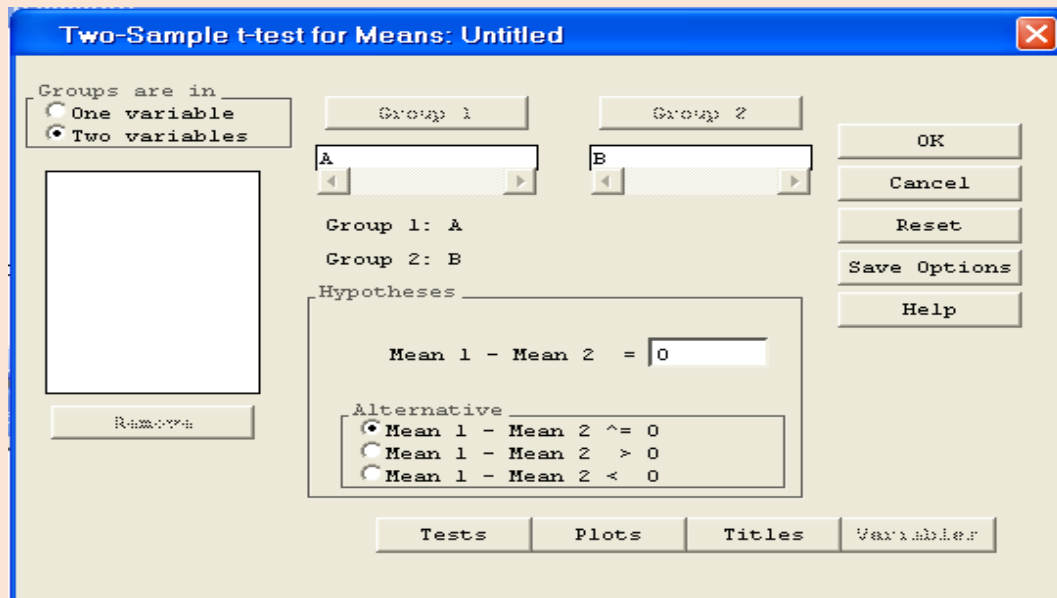
	A	B
1	7930	7660
2	7860	8070
3	8380	7720
4	8230	7690
5	8400	7510

3) اختر من القائمة **Statistics** الامر **Hypothesis Tests** ومنه اختر الامر **Two-sample t-test for means**

كما يلي:



4) يظهر المربع الحوارى نختار الخيار **Two variables** وهو يعنى ان لدينا مجموعتين **A,B** نختار **A** ونضغط على المستطيل **Group1** ونختار **B** ونضغط على المستطيل **Group2** ونحدد الفرض البديل كما في الشكل التالى:



ثم نضغط على OK لتظهر النتائج التالية:

### Two Sample t-test for the Means of A and B

#### Sample Statistics

Group	N	Mean	Std. Dev.	Std. Error
A	5	8160	251.89	12.65
B	5	7730	206.52	92.358

#### Hypothesis Test

Null hypothesis: Mean 1 - Mean 2 = 0

Alternative: Mean 1 - Mean 2  $\neq$  0

If Variances Are	t statistic	Df	Pr > t
Equal	2.952	8	0.0184
Not Equal	2.952	7.70	0.0192

#### (4) تحليل النتائج

- ( N يعطي الجدول الأول بعض الإحصاءات الوصفية مثل عدد البيانات في كل مجموعة )  
( وغيرها (Variance) والتباين ( mean والمتوسط )  
( فيتضح أن Two-tail أما الجدول الثاني حيث أن الاختبار في المثال من طرفين )

وبما أن :  $t_0 = 0.0184 = P\text{-value}$  . و 2.952

$$0.0184 = P -value < \alpha = 0.05$$

نقول : لا نستطيع قبول فرض العدم ( أي أن المنجمين لهما سرعت حرارية مختلفة )

\*\*\*\* بفرض تساوي تبايني المجتمعين

### T-Test: Two-sample Assuming Equal Variances

عندما تكون التباينات غير معلومة ولكن نعرف تساويهما في هذه الحالة يتم حساب تقدير التباين المشترك للمجتمعين على النحو التالي:

$$S_P^2 = \frac{(n_1 - 1)S_1^2 + (n_2 - 1)S_2^2}{n_1 + n_2 - 2}$$

حيث  $S_1, S_2$  تقدير تباين المجتمعين محسوباً من العينة .

ثم يجرى اختبار تساوي متوسطي المجتمعين باستخدام فرض العدم التالي :

$$H_0 : \mu_1 = \mu_2$$

ضد أحد الفروض البديلة :

$$H_1 : \mu_1 > \mu_2 \quad , \quad H_1 : \mu_1 < \mu_2 \quad , \quad H_1 : \mu_1 \neq \mu_2$$

ويتم ذلك بحساب قيمة  $t_0$  في حالة العينات الصغيرة كتالي :

$$t_0 = \frac{\bar{X}_1 - \bar{X}_2}{S_P \sqrt{\frac{1}{n_1} + \frac{1}{n_2}}}$$

الجدولية حيث نرفض فرض العدم إذا كانت : ثم مقارنة قيمة  $t_0$  مع قيمة

$$|t_0| > t$$

حيث أننا نرفض P-value كذلك يمكنك اتخاذ القرار بمقارنة مستوى المعنوية مع قيمة فرض العدم إذا كانت :

$$P -value < \alpha$$

\*\*\* اختبار حول متوسطي البيانات المزدوجة :

### T-Test : Paired Two-sample for Means

في كثير من الأحيان يتعذر الحصول على عينتين عشوائيتين متماثلتين بحيث يمكن استخدامهما في دراسة الفرق بين المتوسطين فنستخدم عينة واحدة مرتين في هذه الحالة يجري اختبار الفرق بين متوسطي البيانات  $\mu_d = \mu_1 - \mu_2 = 0$  باستخدام فرض العدم التالي :

$$H_0 : \mu_d = 0$$

ضد أحد الفروض البديلة التالية :

$$H_1 = \mu_d \neq 0 \quad , \quad H_1 = \mu_d < 0 \quad , \quad H_1 : \mu_d > 0$$

ويتم حساب القيمة  $t_0$  على النحو التالي :

$$t_0 = \frac{\bar{d}}{S_d / \sqrt{n}}$$

حيث أن :

$$\bar{d} = \frac{\sum_{j=1}^n d_j}{n} \quad , \quad d_j = x_{1j} - x_{2j}$$

و  $S_d$  هو الانحراف المعياري لـ  $d_j$

$$S_d = \sqrt{\left( \sum d_j^2 - \frac{(\sum d_j)^2}{n} \right) / (n-1)}$$

الجدولية حيث نرفض فرض العدم إذا كانت  $t$  مقارنة قيمة  $t_0$  مع قيمة

$$|t_0| > t$$

حيث أننا نرفض  $P$ -value كذلك يمكنك اتخاذ القرار بمقارنة مستوى المعنوية مع قيمة

فرض العدم إذا كانت :

$$P\text{-value} < \alpha$$

مثال

في تجربة لمعرفة أسلوب جديد في الوسائل التوضيحية التعليمية ، اجري اختبار على عينة بدون استخدام أية وسائل توضيحية . ثم بعد فترة مناسبة أجي اختبار 9 عشوائية حجمها آخر بعد استخدام الوسائل التوضيحية ، كانت النتيجة كالتالي :

الاختبار الأول : 18, 39, 41, 21, 35, 18, 45, 27, 32

الاختبار الثاني : 35, 36, 51, 23, 29, 45, 43, 42, 35

اختبر هل هناك فرق بين الاختبارين عند مستوى معنوية  $\alpha = 0.05$  ؟

الحل باستخدام ساس :

(1) فرض العدم والفرض البديل كالتالي

$$H_0 : \mu_d = 0$$

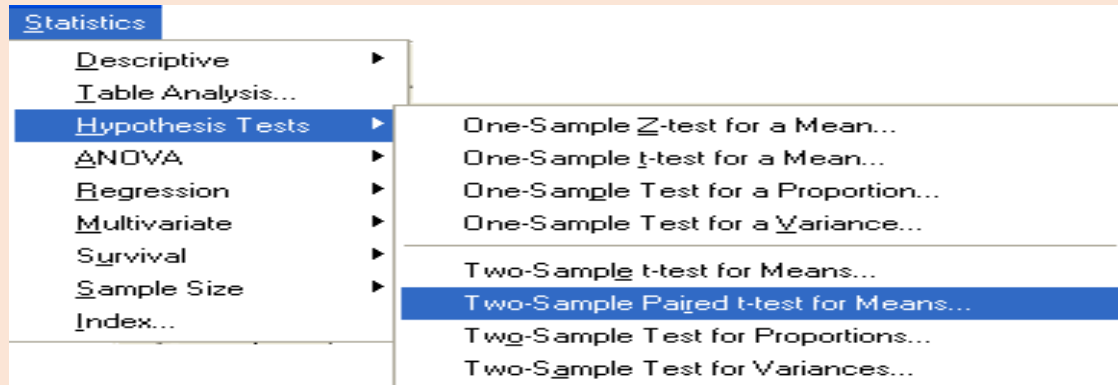
$$H_1 = \mu_d \neq 0$$

(2) ادخل البيانات في حزمة ساس كما في الشكل التالي :

	A	B
1	32	35
2	27	36
3	45	51
4	18	23
5	35	29
6	21	45
7	41	43
8	39	42
9	18	35

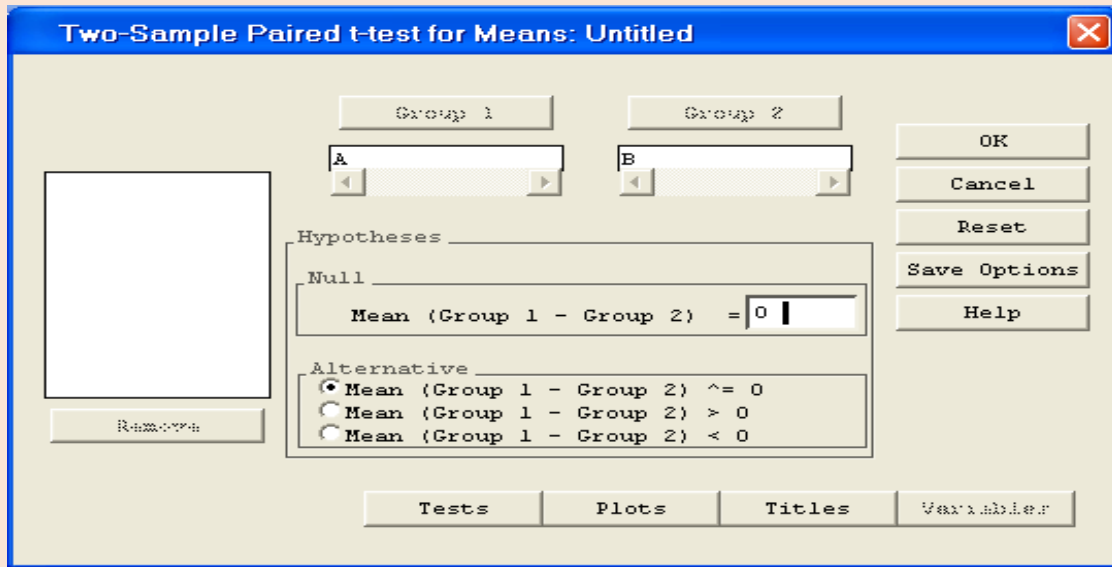
ومنه اختر الأمر **Hypothesis Tests** الأمر **Statistics** ( اختر من القائمة 3)

كما في الشكل التالي : **Two-sample Paired for means**



واضغظ **Two-Sample Paired t-test for Means...**

وحدد الفرض البديل كما في الشكل التالي : **Group 2** على المستطيل



لتظهر النتائج كما يلي : OK ثم اضغط

## Two Sample Paired t-test for the Means of A and B

### Sample Statistics

Group	N	Mean	Std. Dev.	Std. Error
A	9	30.66667	10.186	3.3953
B	9	37.66667	8.5586	2.8529

### Hypothesis

#### Test

Null hypothesis: Mean of (A - B) = 0

Alternative: Mean of (A - B)  $\leq$  0



t Statistic      Df      Prob > t

-2.378      8      0.0447

( تحليل النتائج . 5 )

( N يعطي الجدول الأول بعض الإحصاءات الوصفية مثل عدد البيانات في كل مجموعة )  
( وغيرها ( Variance. والتباين ( mean والمتوسط ) )

( فيتضح أن Two-tail أما الجدول الثاني حيث أن الاختبار في المثال من طرفين )

وبما أن :  $t_0 = 0.0447 = P\text{-value}$  و  $-2.378$

$$0.0447 = P\text{-value} < \alpha = 0.05$$

نقول : لا نستطيع قبول فرض العدم ( أي يوجد فروق بين الاختبارين ) .

\*\*\* اختبار تساوي تباين مجتمعين :

### F-Test : Two-sample for Variances

يجري اختبار تساوي تباين مجتمعين طبيعيين  $\sigma_1, \sigma_2$  باستخدام فرض العدم

$$H_0 : \sigma_1^2 = \sigma_2^2$$

ضد أحد الفروض البديلة التالية

$$H_1 : \sigma_1^2 > \sigma_2^2 \quad , \quad H_1 : \sigma_1^2 < \sigma_2^2 \quad , \quad H_1 : \sigma_1^2 \neq \sigma_2^2$$

ويتم ذلك بحساب  $F_0$  كتالي :

$$F_0 = \frac{S_1^2}{S_2^2}$$

الجدولية ، حيث نرفض فرض العدم إذا كانت  $F_0$  مقارنة قيمة  $F_0$  مع قيمة

$$F_0 > F$$

حيث أننا نرفض  $P$ -value كذلك يمكنك اتخاذ القرار بمقارنة مستوى المعنوية مع قيمة فرض العدم إذا كانت :

$$P \text{ -value} < \alpha$$

مثال على الطالب حل هذا المثال:

## \*\* المعانية العشوائية Sampling

يمكنك Random Sample اذا كان لدينا بيانات وأردنا أن نأخذ عينة عشوائية فإن الأمر من ذلك بسهولة .

مثال

نريد أخذ عينة عشوائية حجمها 3 من البيانات التالية

32, 27, 45, 18, 35, 21, 41, 39, 18, 35, 36, 51, 23, 29, 45, 43,  
42, 35

الحل باستخدام ساس :

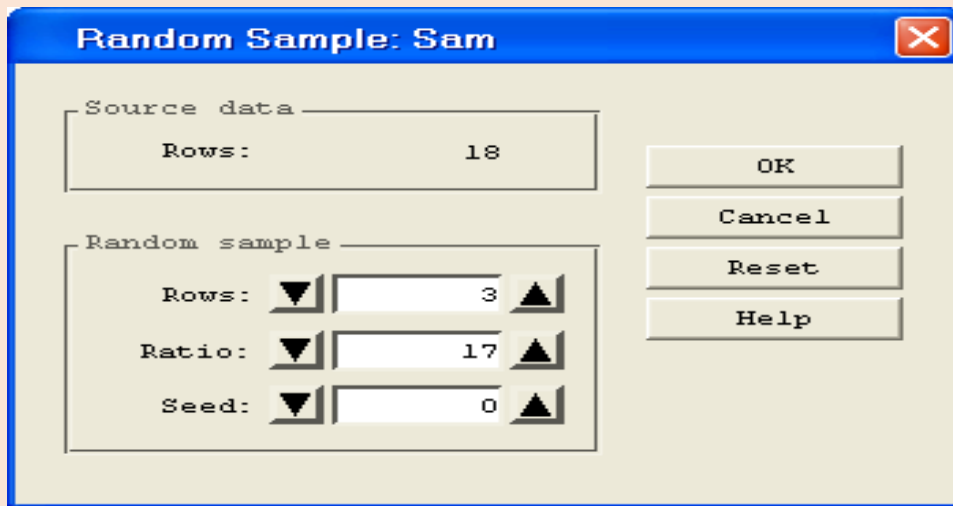
1) ادخل البيانات كما في الشكل التالي

	Δ
1	32
2	27
3	45
4	18
5	35
6	21
7	41
8	39
9	18
10	35
11	36
12	51
13	23
14	29
15	45
16	43
17	42
18	35

(2) من القائمة Data اختار الامر Random Sample كما يلي:



. كما Rows ( يظهر لك مربع حوار حدد فيه حجم العينة فقط وذلك بداخل مستطيل 3) في الشكل التالي :



لتظهر النتيجة كما في الشكل التالي : OK ثم اضغط

	A
1	39
2	18
3	43

، ويعني Rows يتغير تلقائياً بتغير قيمة المستطيل Ratio ملاحظة : قيمة المستطيل  
أخذ نسبة مئوية من حجم البيانات . في مثالنا السابق أخذنا عينة بحجم Ratio المستطيل  
من البيانات 17%

**\*\* مقارنة بين اختبارات إحصائية واختبارات الفروض :**

الاختبارات الإحصائية سبق شرحها في الفصل الثاني ، واختبارات الفروض الإحصائية سبق شرحها في الفصل الثالث . وكلا الموضوعين يعالجان اتخاذ القرار .

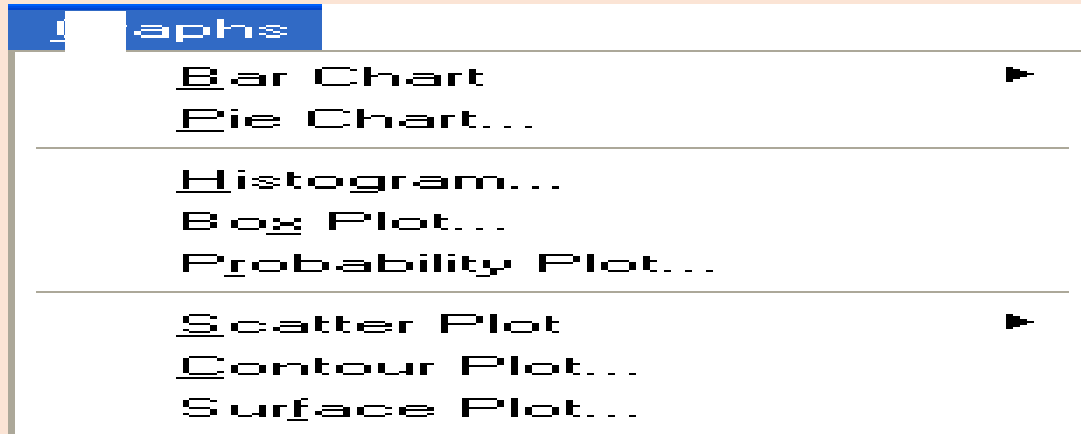
اختبارات الفروض	اختبارات إحصائية	نوع الاختبار
يحسب القيمة الجدولية P-value والمحسوبة وكذلك	فقط P – value يحسب	
اختبار تساوي مجتمعين	اختبار تساوي مجتمع بقيمة معينة	Z – Test
يقوم بجميع خيارات هذا الاختبار بنفس ما يتم في الاختبارات الإحصائية ولكن كل خيار في نافذة منفصلة	يقوم بجميع خيارات هذا الاختبار في نافذة واحدة	T – Test
اختبار تساوي التباينات من طرف واحد	اختبار تساوي التباينات من طرفين	F – Test
اختبار الاستقلال وهو غير موجود في هذا الفصل	اختبار الاستقلال وهو موجود في هذا الفصل	Chi – Test

## الفصل الخامس

### التحليل الإحصائي باستخدام الأشكال البيانية

هناك بعض الجداول الإحصائية يلزم عرضها في شكل رسومات هندسية لتبسيطها وجعل رؤية العلاقة بين المتغيرات أكثر سهولة من الجدول من حيث الزيادة والنقصان لبعض الظواهر خلال فترة زمنية محددة .

في حزمة ساس توجد قائمة تختص بعمل الرسومات وهي قائمة Graphs وهي تضم أنواع من الرسومات والتي سنستخدم بعضها ، وشكل القائمة كما يلي :



\* الرسم باستخدام الأمر ( Bar Chart ) :

مثال:

في دراسة لاختبار اللياقة للعدائين وذلك بالاعتماد على معدل استهلاك العدائين للأوكسجين ، اختيرت عينة مكونة من 31 عداء وطلب منهم قطع ميل ونصف الميل يتخللها

فترة ركض وراحة وذلك لإجراء قياس نبضات القلب لكل عداء ، وتم جمع معلومات وهي كالتالي :  
 العمر ، الوزن ، وقت قطع المسافة بالدقائق ، قياس نبض القلب في فترة الراحة ، قياس نبض القلب أثناء الركض ، قياس أعلى نبض للقلب ، معدل استهلاك الأوكسجين . حيث أن الهدف من الدراسة هو اكتشاف مستوى لياقة العدائين .

الحل باستخدام ساس :

البيانات محفوظة في حزمة ساس ومقسمة إلى ثلاث مجموعات ( 0 ، 1 ، 2 )  
 ولإظهارها نتبع الخطوات التالية :

1 ( من قائمة Tools اختر الأمر Sample Data .

2 ( يظهر مربع يحوي مجموعة من الخيارات اختر الخيار Fitness واختر المكتبة التي تريد وضع الخيار فيها . ولتكن مكتبة Sasuser . ثم اضغط على OK .

3 ( من قائمة File اختر الأمر Open By SAS Name يظهر مربع يحوي المكتبات اضغط على المكتبة Sasuser واختر الملف Fitness الذي يحوي البيانات ثم اضغط OK وستظهر البيانات كما في الشكل التالي :

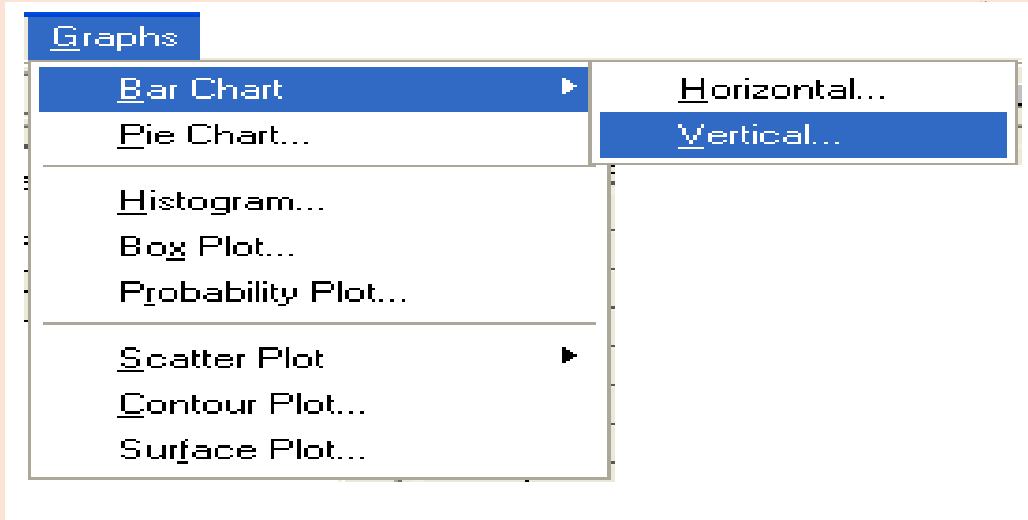
	age	weight	runtime	rstpulse	ru
1	57	73.37	12.63	58	
2	54	79.38	11.17	62	
3	52	76.32	9.63	48	
4	50	70.87	8.92	48	
5	51	67.25	11.08	48	
6	54	91.63	12.88	44	
7	51	73.71	10.47	59	
8	57	59.08	9.93	49	
9	49	76.32	9.4	56	
10	48	61.24	11.5	52	
11	52	82.78	10.5	53	
12	44	73.03	10.13	45	
13	45	87.66	14.03	56	
14	45	66.45	11.12	51	
15	47	79.15	10.6	47	
16	54	83.12	10.33	50	
17	49	81.42	8.95	44	
18	51	69.63	10.95	57	
19	51	77.91	10	48	
20	48	91.63	10.25	48	
21	49	72.37	10.08	76	



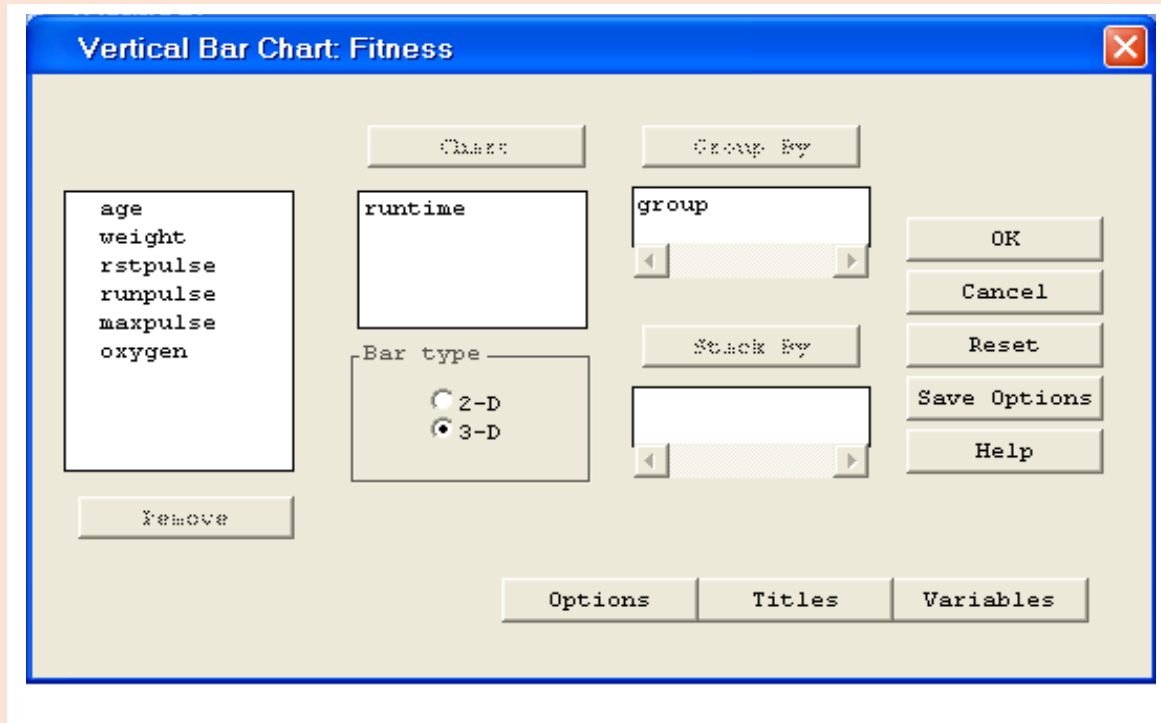
والآن سنبدأ الحل :

1 ( اختر من القائمة Graphs الأمر Bar Chart منه أحد الأمرين Vertical أو Horizontal للرسم العمودي أو الأفقي ، وسنختار الرسم

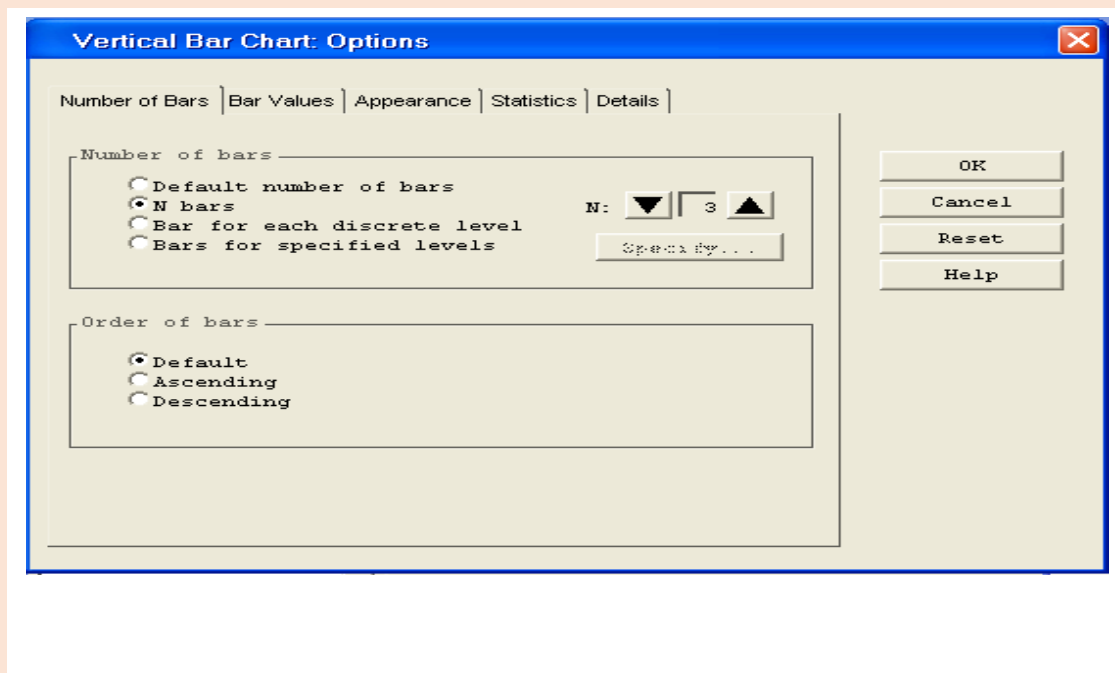
في الشكل التالي :



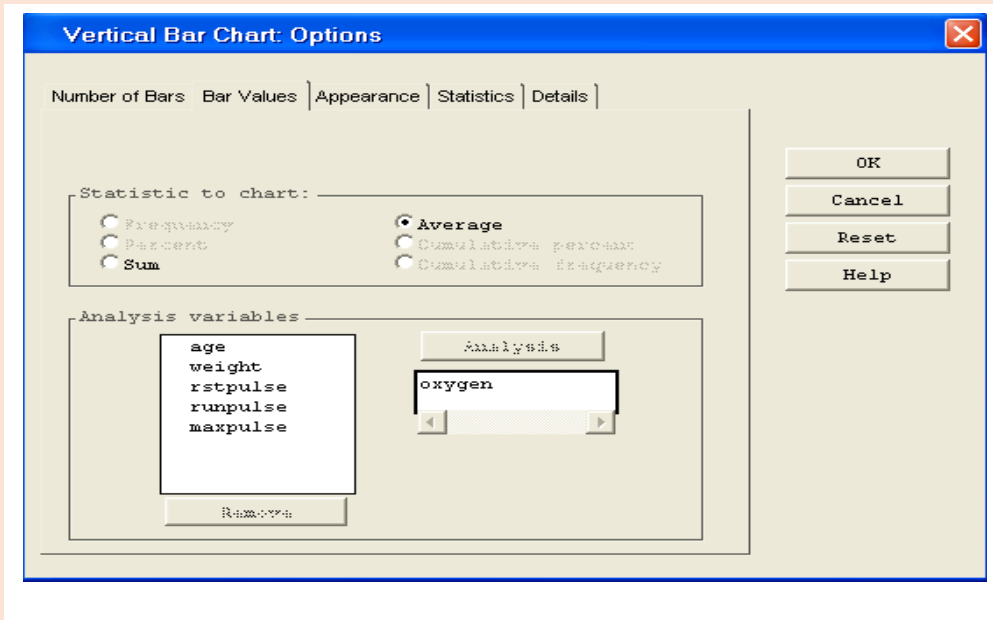
2 ( يظهر مربع حوار، ولتكوين رسم بياني معتمد على الوقت المستغرق لقطع ميل ونصف الميل حدد العمود runtime واضغط على المستطيل Chart لينتقل تحته حيث سيكون العمود runtime في الرسم البياني هو المحور السيني ، ثم حدد عمود المجموعات Group واضغط على المستطيل Group By لينتقل تحته ، وحدد شكل الرسم في مربع Bar type ( ثنائي الأبعاد أو ثلاثي الأبعاد ) كما في الشكل التالي :



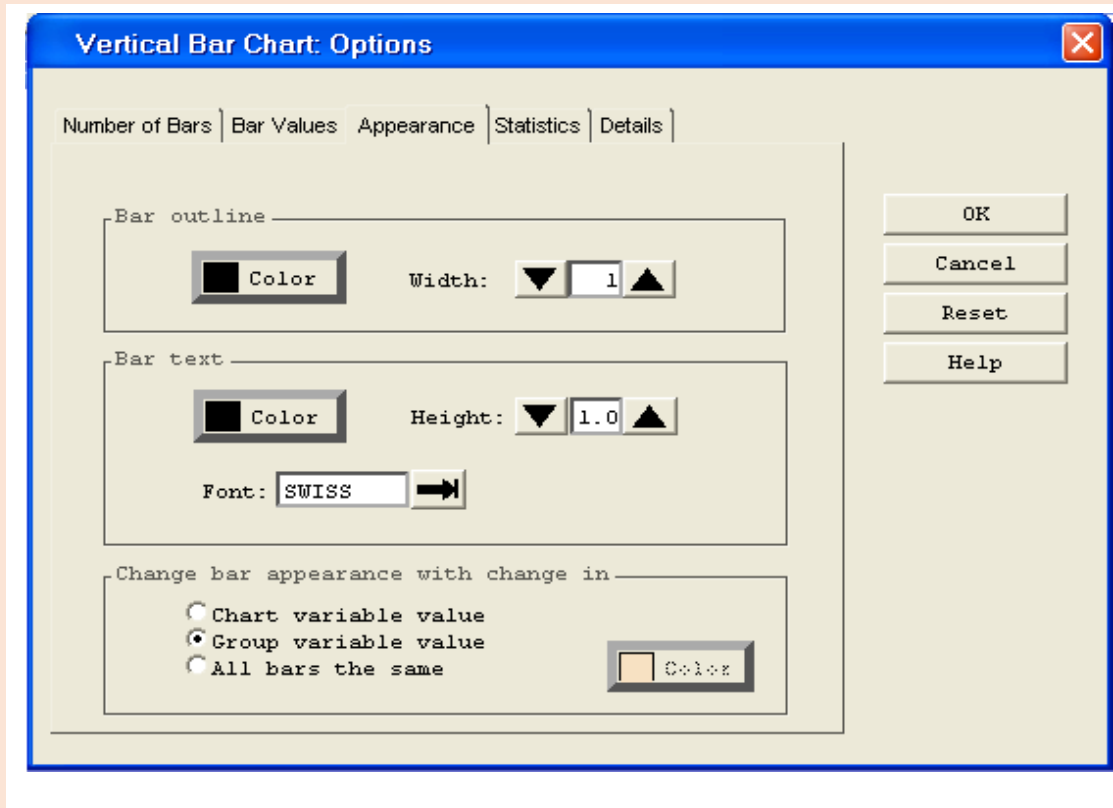
3 ) من المربع السابق اضغط على زر **Options** لخيارات الرسم البياني ، حدد عدد أعمدة كل مجموعة وذلك باختيار الخيار **N bars** ، وليكن عدد الأعمدة  $N = 3$  ثم اختر الخيار **Default** .



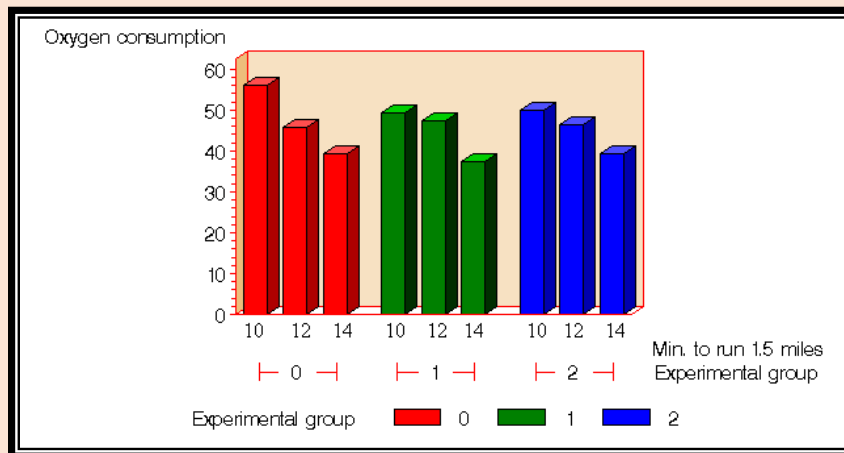
4 ) في نفس مربع Options اضغط على Bar Values لتحليل المتغيرات ، وفيه حدد المتغير المراد تحليله مثل الأوكسجين ( oxygen ) ، ولإيجاد معدل استهلاك العدائين للأوكسجين والذي سيكون في الرسم البياني المحور الصادي اختر الخيار Average كما في الشكل التالي :



5 ) في نفس مربع Options اضغط على Appearance لخيارات الألوان ولتحديد سمك الأعمدة وطولها ، وفيه اختر الألوان المناسبة ، ويفضل اختيار الخيار Group variable value داخل مستطيل عنوانه Change bar appearance with change in كما في الشكل التالي :



ثم اضغط على OK للعودة إلى المربع الذي في الخطوة ( 1 ) وفيه اضغط على OK لتنفيذ الرسم ( العلاقة ما بين قطع المسافة واستهلاك الأوكسجين وسيظهر على الشكل التالي :



## 6 ( تحليل الرسم البياني :

نلاحظ من الرسم أن المجموعة الأولى ويرمز لها بالرمز ( 0 ) هي مجموعة العدائين الأسرع في قطع مسافة ميل ونصف الميل فهم الأكثر لياقة والأكثر استهلاكاً للأوكسجين ويأتي بعدهم المجموعة الثانية ورمزها ( 1 ) ، والمجموعة الثالثة ورمزها ( 2 ) وهي المجموعة الأبطأ في قطع المسافة وبالتالي فهم الأقل لياقة والأقل استهلاكاً للأوكسجين .

\* الرسم باستخدام الأمر ( Pie Chart ) :

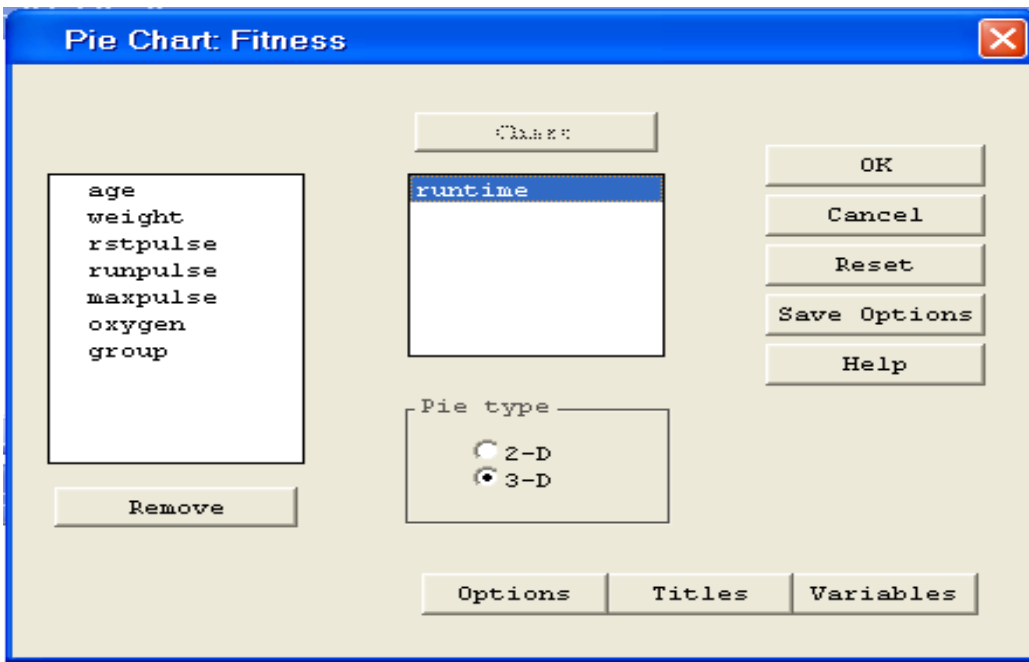
مثال:

في المثال السابق ، لو أردنا حساب نسبة أغلب الأوقات التي قطع فيها العدائين مسافة الميل ونصف الميل ، ويتم ذلك بالخطوات التالية :

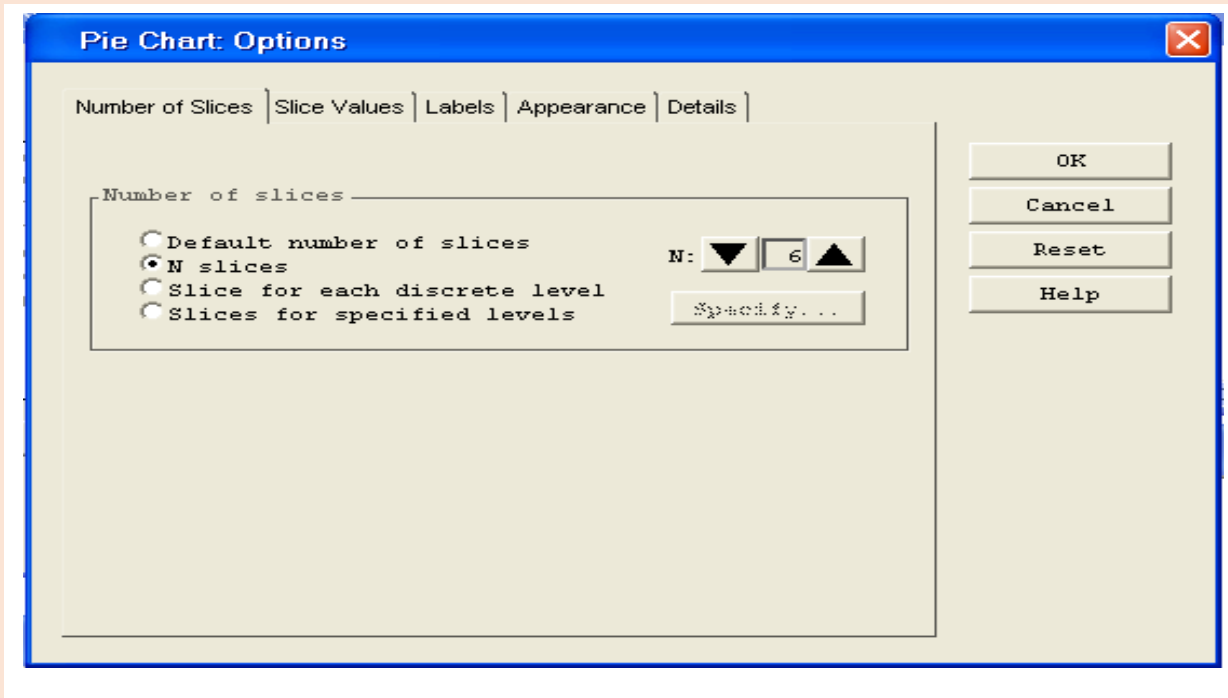
1 ( اختر من القائمة Graphs الأمر Pie Chart كما في الشكل التالي :



2 ( يظهر مربع حوار، ولتكوين رسم بياني . حدد العمود runtime واضغط على المستطيل Chart لينتقل تحته وحدد نوع الأبعاد للرسم البياني وليكن ثلاثي الأبعاد 3-D . كما في الشكل التالي :

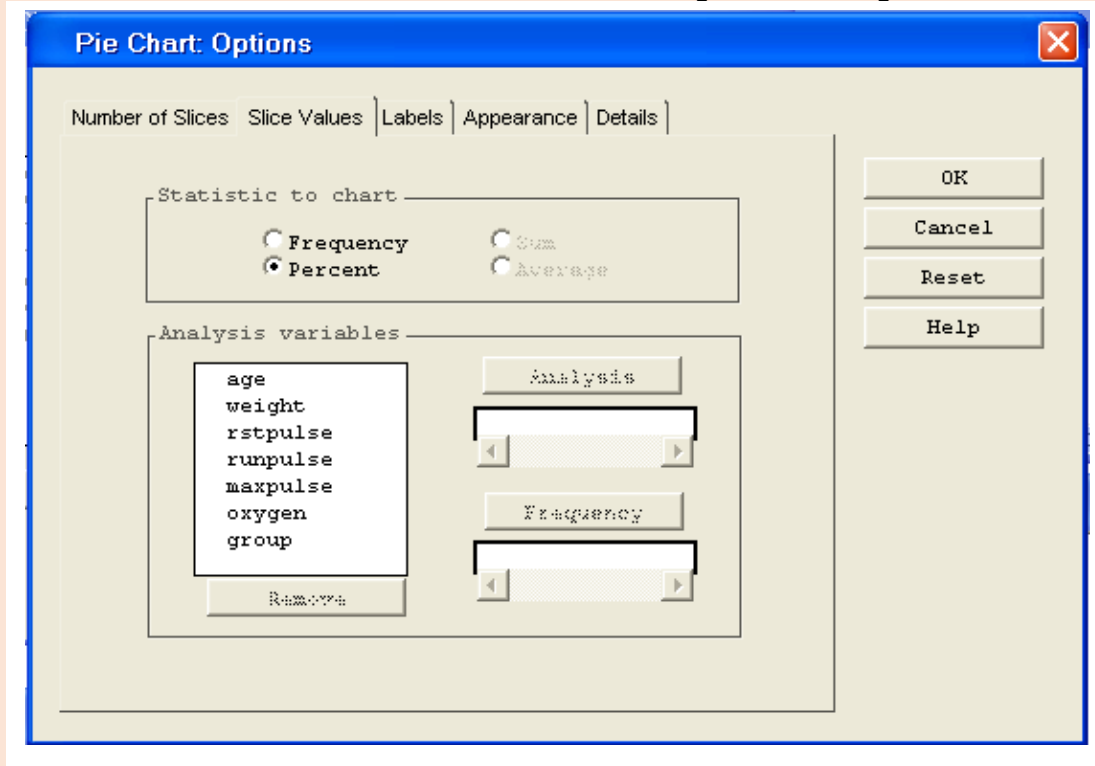


1 ) من المربع السابق اضغط على زر **Options** لخيارات الرسم البياني ، حدد عدد الشرائح المقسمة بالنسبة للوقت المستغرق وذلك باختيار الخيار **N slices** وتحديد العدد المراد تحديده ، وليكن 6 شرائح ، كما في الشكل التالي :



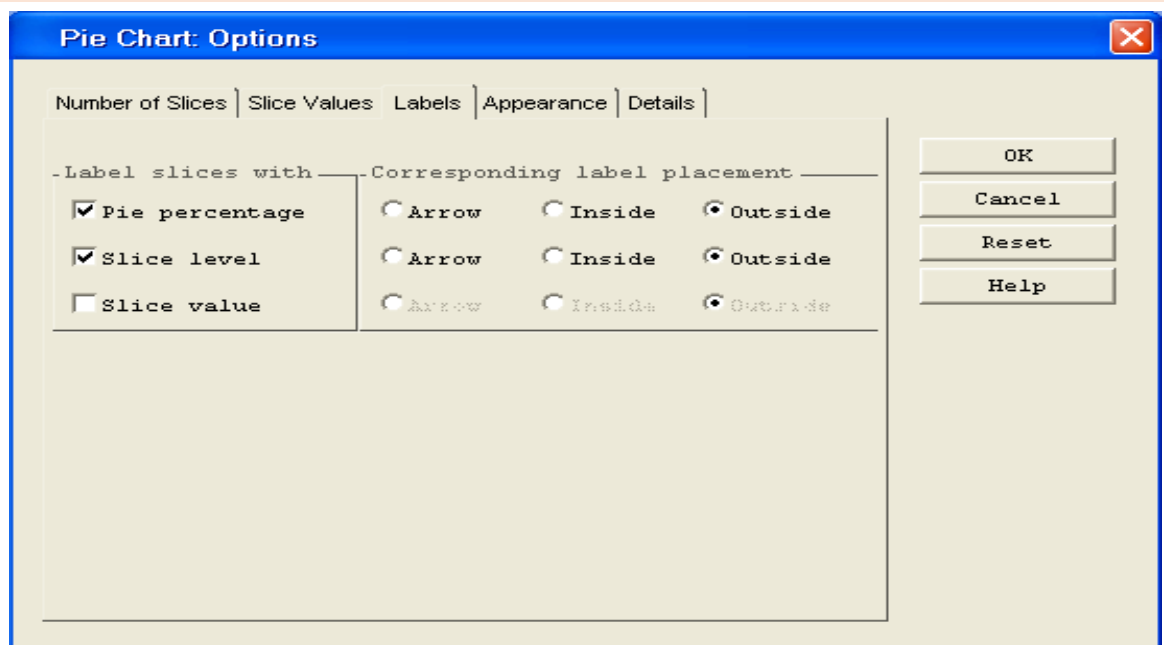
4 ) في نفس مربع Options ( مربع الخطوة ( 3 ) ) اضغط على Slice value وفيه

اختر الخيار Percent كما في الشكل التالي :

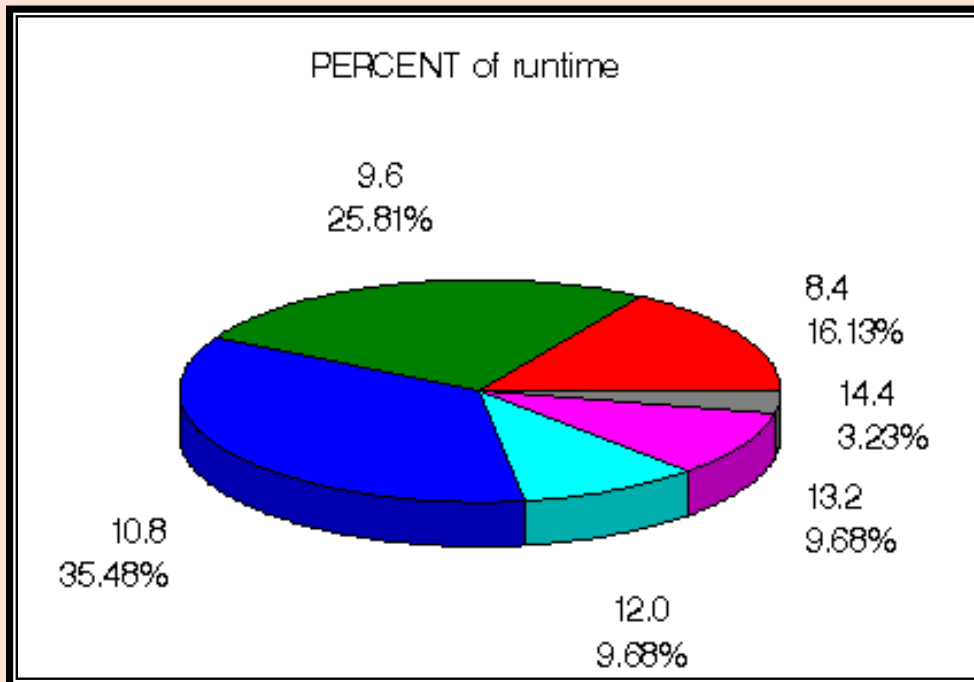


5 ) في نفس مربع Options اضغط على Labels لتنظيم وضع الأرقام التابعة للرسم

البياني ، وفيه اختر بعض الخيارات كما في الشكل التالي :



ثم اضغط OK للعودة إلى مربع الحوار السابق في الخطوة الأولى وفيه اضغط على OK لتنفيذ الرسم الذي سيظهر كما في الشكل التالي :





## 6 ( تحليل الرسم :

نسبة أكثر الأوقات التي قطع فيها أكثر العدائين مسافة الميل ونصف الميل بمعدل 10.8 دقيقة هي 35.48% بينما الأوقات قطع فيها القليل من العدائين بمعدل 14.4 بنسبة 3.23%

### \* رسم المدرج التكراري ( Histogram ) :

هو عبارة عن تنفيذ تكرار كل فئة من فئات التوزيع التكراري بمستطيل حدود قاعدته الحدود الفعلية لتلك الفئة ، وارتفاعه يتناسب مع تكرارها . أي أننا نأخذ محورين متعامدين ، نرصد على المحور الأفقي الحدود لكل فئة من فئات التوزيع التكراري ونقيم على كل فئة مستطيل يتناسب ارتفاعه مع تكرار تلك الفئة كما يتضح من المثال التالي :

مثال:

البيانات التالية تمثل أطوال حياة خمسين بطارية سيارة ( لأقرب شهر )

29, 28, 25, 26, 26, 27, 25, 30, 33, 34, 32, 31, 30, 34, 32, 31, 33,  
30, 34, 31, 31, 39, 35, 31, 33, 31, 32, 30, 38, 37, 35, 39, 36, 37,  
38, 38, 42, 37, 44, 36, 40, 35, 44, 49, 40, 45, 41, 39, 43, 48

ارسم المدرج التكراري لأعمار هذه البطاريات .

الحل باستخدام ساس :

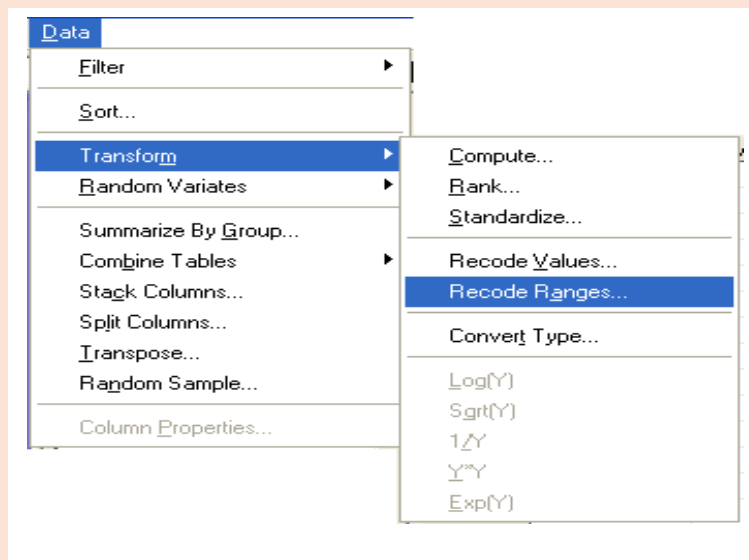
1 ( ندخل البيانات في عمود A من 1 إلى 50 كما في الشكل التالي :

	A
1	29
2	28
3	25
4	26
5	26
6	27
7	25
8	30
9	33
10	34
11	32
12	31
13	30
14	34
15	32
16	31
17	32

2 ( من القائمة Edit اختر الأمر Mode ومنه اختر الأمر Edit .

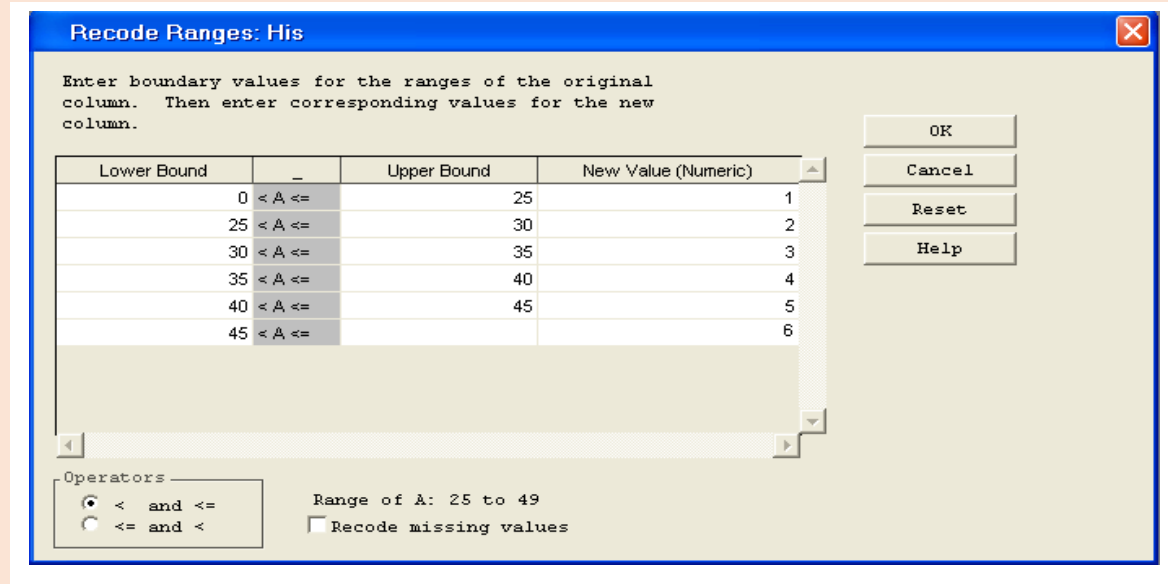
3 ( بعد ذلك ظلل أو حدد العمود A ، واختر القائمة Data ومنها اختر الأمر

Transform ومنه اختر الأمر Recode Ranges . كما في الشكل التالي :



4 ) يظهر لك مربع حدد فيه اسم العمود A داخل المستطيل ( Column to recod ) أما المستطيل ( New Column Name ) يحدد تلقائياً الاسم للعمود الجديد الذي يقسم البيانات إلى الفئات و تستطيع تحديد اسمه ، كما تستطيع تحديد محتوى العمود الجديد إما عددي Numeric أو حرفي Charater ، وليكن عددي . حدد عدد الفئات ولتكن 6 فئات . ولاحظ أن المربع حدد لك فترة المدى في السطر Rang of A : 24 إلى 49 . كما في الشكل التالي :

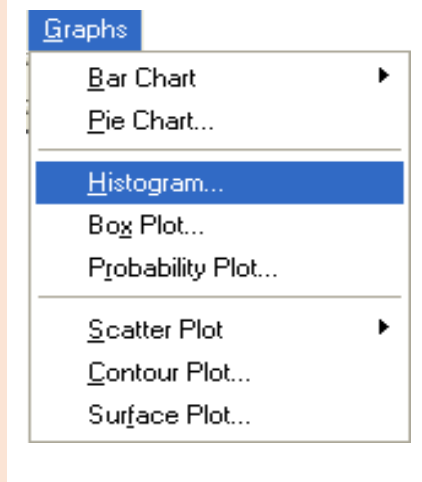
5 ) ثم اضغط على OK ليظهر لك مربع آخر حدد فيه فترات ورقم تسلسل كل فئة وذلك باختيار  $\leq$  and  $<$  داخل المستطيل Operators وقم بتحديد كل فترة حيث وذلك بتحديد الحد الأدنى والحد الأعلى وهي كما يلي  $(0 < A \leq 25)$  و  $(25 < A \leq 30)$  و  $(30 < A \leq 35)$  و  $(35 < A \leq 40)$  و  $(40 < A \leq 45)$  و  $(45 < A \leq \infty)$  حيث يترك المستطيل الأخير من العمود Upper Bound فارغاً ويدل ذلك على  $\infty$  ، كما في الشكل التالي :



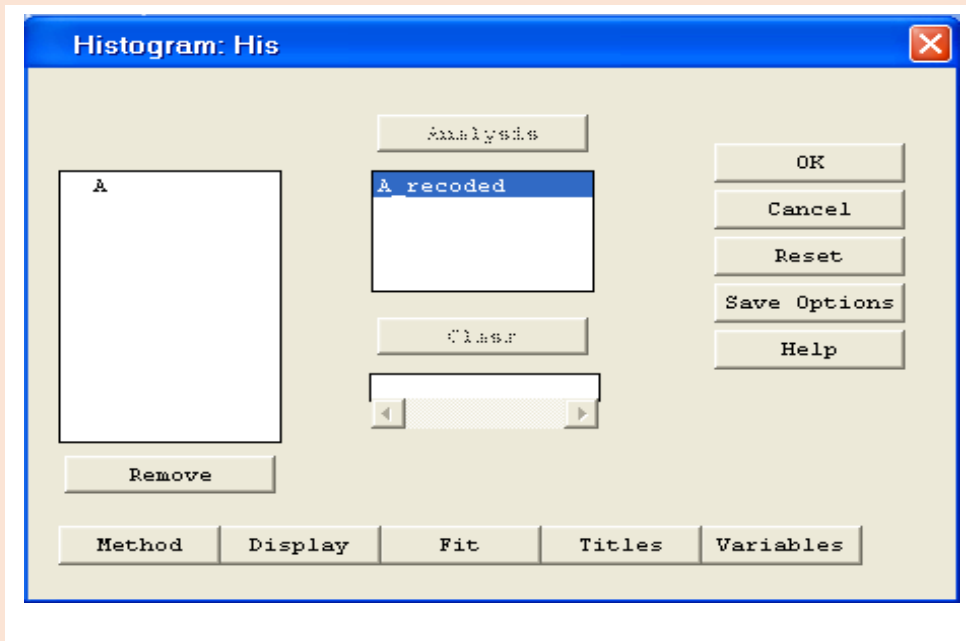
ثم اضغط OK ليظهر العمود الجديد ( A\_recoded ) بجانب العمود A كما في الشكل التالي :

	A	A_recoded
1	29	2
2	28	2
3	25	1
4	26	2
5	26	2
6	27	2
7	25	1
8	30	2
9	33	3
10	34	3
11	32	3
12	31	3
13	30	2
14	34	3
15	32	3
16	31	3
17	33	3
18	33	3
19	33	3
20	33	3
21	33	3
22	33	3
23	33	3
24	33	3
25	33	3
26	33	3
27	33	3
28	33	3
29	33	3
30	33	3
31	33	3
32	33	3
33	33	3
34	33	3
35	33	3
36	33	3
37	33	3
38	33	3
39	33	3
40	33	3
41	33	3
42	33	3
43	33	3
44	33	3
45	33	3
46	33	3
47	33	3
48	33	3
49	33	3
50	33	3
51	33	3
52	33	3
53	33	3
54	33	3
55	33	3
56	33	3
57	33	3
58	33	3
59	33	3
60	33	3
61	33	3
62	33	3
63	33	3
64	33	3
65	33	3
66	33	3
67	33	3
68	33	3
69	33	3
70	33	3
71	33	3
72	33	3
73	33	3
74	33	3
75	33	3
76	33	3
77	33	3
78	33	3
79	33	3
80	33	3
81	33	3
82	33	3
83	33	3
84	33	3
85	33	3
86	33	3
87	33	3
88	33	3
89	33	3
90	33	3
91	33	3
92	33	3
93	33	3
94	33	3
95	33	3
96	33	3
97	33	3
98	33	3
99	33	3
100	33	3

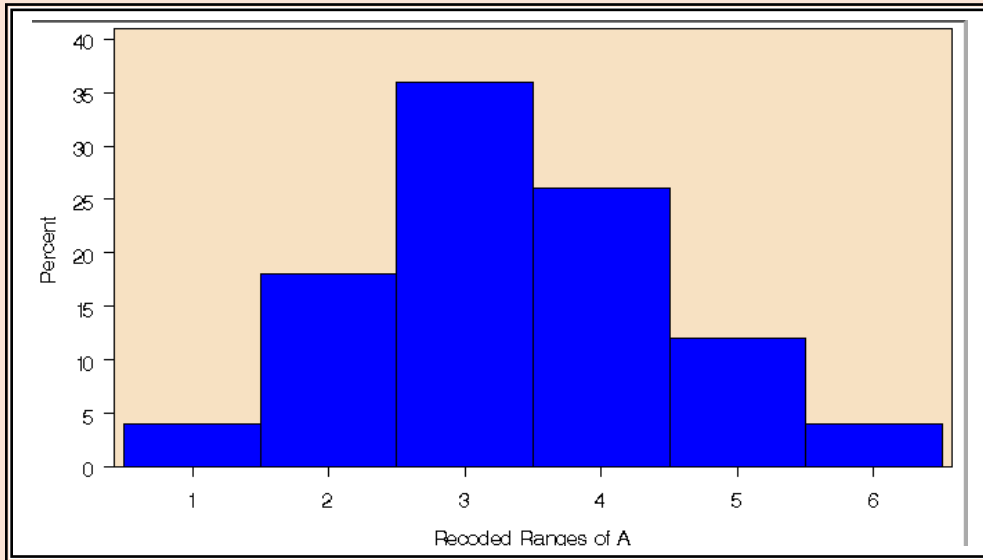
6 ( و لرسم المدرج التكراري اختر من القائمة Graphs الأمر Histogram كما في الشكل التالي :



7 ( يظهر لك مربع حوار ، وفيه حدد اسم العمود الجديد ( A\_recoded ) واضغط على زر Analysis لينتقل تحته كما في الشكل التالي :

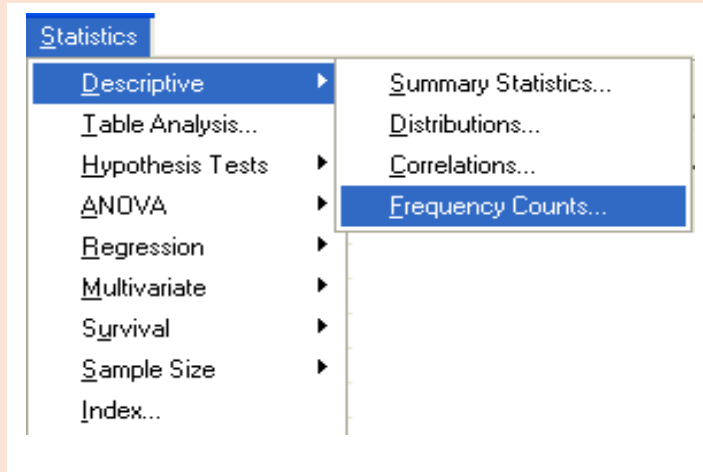


ثم اضغط على OK ليظهر رسم المدرج التكراري كما في الشكل التالي :



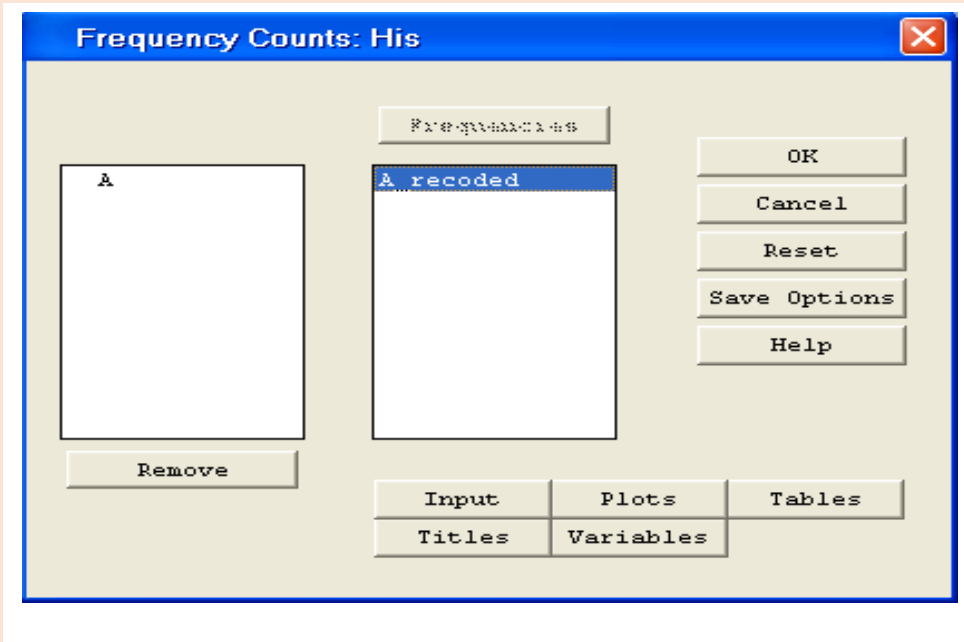
8 ( ويمكن إيجاد تكرار كل فترة , فمن قائمة **Statistics** اختر الأمر **Descriptive** ومنه

اختر الأمر **Frequency Counts** كما في الشكل التالي :



9 ( يظهر لك مربع حوار ، وفيه حدد اسم العمود الجديد ( **A\_recoded** ) واضغط على زر

**Frequencies** لينتقل تحته كما في الشكل التالي :



ثم اضغط على OK لتظهر لك النتيجة كما يلي :

### The FREQ Procedure

#### Recoded Ranges of A

A_recoded	Frequency	Cumulative Percent	Cumulative Frequency	Cumulative Percent
-----------	-----------	-----------------------	-------------------------	-----------------------

////////////////////

1	2	4.00	2	4.00
2	9	18.00	11	22.00

3	18	36.00	29	58.00
4	13	26.00	42	84.00
5	6	12.00	48	96.00
6	2	4.00	50	100.00

العمود Frequency يبين لنا تكرار الفترات فمثلا القيم التي تنتمي إلى الفترة  $30 < A \leq 35$  يساوي 18 .



## تدريبات عامة

\* أختار الأجابة الصحيحة من بين (أ) و (ب) و (ج) و (د)

\*\* الرمز التالي يمثل عملية التكرار أو الدوران:

(أ)  (ب)  (ج)  (د) لا شيء مما سبق

\*\* يقوم بترجمة البرنامج المصدر الى برنامج بلغة الآلة ويترجم التعليمات وينفذها تعليمة تلو الأخرى:

(أ) Compiler (ب) Interpreter (ج) Assembler (د) كل ما سبق

\*\* يقوم الحاسب بمعالجة ..... لتصبح معلومات ذات معنى:

(أ) المعلومات (ب) المعرفة (ج) البيانات (د) غير ذلك

\*\* الآتى وحدات نظام System Units فيما عدا:

(أ) وحدات الإدخال (ب) وحدة المعالجة المركزية (ج) الذاكرة الرئيسية (د) الذاكرة الثانوية

\*\* التالي وحدات إدخال فيما عدا:

(أ) لوحة المفاتيح (ب) السماعة الخارجية (ج) الكاميرا الرقمية (د) الماكروفون

\*\* تعد الطابعة من:

(أ) وحدات أذخال (ب) وحدات نظام (ج) وحدات أذراج (د) كل ما سبق

\*\* تعد لغة الآلة من:

(أ) لغات عالية المستوى (ب) أذى لغات التجميع (ج) لغات منخفضة المستوى (د) كل ما سبق

2 - استكمل العبارات التالية:

\* يعرف النظام بأنه

.....  
.....

\* تصنف مكونات الحاسب المادية Hardware الى جزئين هما:

..... \*\*  
..... \*\*

\*\* تستخدم ..... لتحويل الأوامر المكتوبة بلغة برمجة معينة الى لغة الآلة

\*\* البرنامج المصدر Source Program

.....  
.....

\*\* البرنامج الهدف Object Program هو

.....  
.....

\* أرسم خريطة تدفق لما يلي:

\*\* حساب الوسط الحسابي لخمسة أرقام؟

\*\* طباعة الأرقام من 1 الى 50 على الشاشة؟

\*\* إيجاد ناتج المعادلة التالية:

$$F(X) = \begin{cases} X & X \geq 0 \\ -X & X < 0 \end{cases}$$

\* صمم برنامج بلغة C++ لحل المشاكل التالية:

\*\* يستقبل عدد من لوحة المفاتيح ثم يحدد ما إذا كان العدد زوجي أم فردي

\*\* قراءة أربعة قيم يدخلها المستخدم، ثم يطبعها كل واحدة على سطر بين علامتي اقتباس بسيطة

\*\* حساب مكعب العدد المدخل من قبل المستخدم؟

\*\* إيجاد أصغر عنصر في مصفوفة ثنائية الأبعاد؟

\*\* أدخل عدد الثواني وتحولها الى صيغة الوقت فمثلاً تحويل 4586 ثانية الى 1: 16: 26؟

\*\* يستقبل عدد من المستخدم ثم يطبع إذا كان العدد زوجي أم لا؟

\* ما هو الخطأ في الآتي:

- `cin << value;`

- `while (z>=0)`

`sum +=z;`

- `for ( x = 100, x>=1,x++)`

- `char str [5];`

`cin>>str; // user types hello`

- `int a[3] ;`

`cout<<a[1] << “ “ << a[2],<<” “<<a[3] <<endl;`

- `float f[3] = { 1.1 ,10.01 , 100.001 , 1000.0001 } ;`

- `double d[2][10];`

`d[1,9] = 2.345;`

\* ما هي المخرجات

\*\* بفرض أن `x=9 , y=11`

```
if ( x < 10)
if ( y > 10)
    cout << "*" * * * * " << endl;
    else
    cout << "# # # # #" << endl;
    cout << "$ $ $ $ $" << endl;
```

\*\*

```
#include <iostream.h>
main ( )
{
int y, x = 1, total =0;
while (x<= 10) {
y = x+x;
cout <<y << endl;
total +=y;
++x;
}
cout << " total is: " << total << endl;
return 0;
}
```

## تدريبات أخرى:

### أولاً:

١. حدد ما إذا كانت العبارات الآتية صحيحة أم خطأ:
  - التعليقات تجبر الحاسوب على طباعة النص الذي يلي // على الشاشة عند تنفيذ البرنامج.
  - تتابع الهروب \n يجبر المؤشر على الانتقال إلى سطر جديد.
  - برنامج C++ والذي يقوم بطباعة ثلاث أسطر على الشاشة يجب أن يحتوى على ثلاث عبارات تستعمل cout.
  - يجب الإعلان عن المتغيرات قبل استعمالها في البرنامج.
  - يجب تحديد نوع المتغيرات عند الإعلان عنها.
  - لا تفرق C++ بين المتغيرات Number و number .

- \* تمتلك العوامل الحسابية + ، - ، % نفس درجة الأولوية
- \* برنامج C++ والذي يقوم بطباعة ثلاثة أسطر على الشاشة يجب أن يحتوى على ثلاث عبارات تستعمل cout

## ثانياً:

### 1/ أكتب عبارة C++ تؤدي التالي:

- إدخال قيمة متغير صحيح x باستخدام cin و >> .
- إدخال قيمة متغير صحيح y باستخدام cin و >>.
- تمهيد قيمة متغير صحيح i عند 1.
- تمهيد قيمة متغير صحيح power عند 1.
- ضرب قيمة المتغير x في المتغير power وتعيد النتيجة للمتغير power.
- زيادة قيمة المتغير y ب 1.
- اختبار ما إذا كانت قيمة المتغير y أقل من أو تساوي x.
- طباعة قيمة المتغير power.

## ثالثاً:

أي من العبارات الآتية تعطي المخرجات التالية:

1

2

2

4

- 1- cout << " 1\t2\t\n3\t4\n";
- 2- cout <<'1' << '\t' << '2' << '\n' <<'3' <<'\t' <<'4' <<'\n';
- 3- cout << "1 \n 2\t 3\n 4\t";
- 4- cout <<1 << '\t' << 2 << '\n' <<3 <<'\t' <<4 <<'\n';

رابعاً:

أكتب عبارة **C++** صحيحة تقوم بالآتي:

- تعريف المتغيرات **x**، **y**، **z** و **result** لتكون من النوع **int**.
- الطلب من المستخدم إدخال ثلاثة أرقام صحيحة.
- توضيح أن برنامجاً ما سيقوم بحساب حاصل ضرب ثلاثة أرقام صحيحة.
- الطلب من المستخدم إدخال ثلاثة أرقام صحيحة.
- إدخال ثلاثة أرقام عن طريق لوحة المفاتيح وتخزين قيمها في المتغيرات **x**، **y** و **z**.
- حساب حاصل ضرب الأرقام المخزنة في المتغيرات **x**، **y** و **z** وتعيين النتيجة للمتغير **result**.
- طباعة العبارة " The product is: " متبوعة بقيمة المتغير **result**.
- إرجاع قيمة من الدالة **main** لتوضيح أن البرنامج انتهى بنجاح.



خامسا: ما هو الغرض من البرنامج التالي:

\*

```
#include <iostream.h>
main ( )
{
int x ,y ;
cout << "Enter two integers in the range 1-20";
cin >> x>> y;
for (int I = 1; I <= y ; I++) {
    for ( int j = 1; j <= x; j++)
        cout << " ";
    cout << endl;
}
return 0;
}
```

```
#include <iostream.h>
int WhatIsThis (int[ ] ,int);
main
{
const int arraysize = 10;
int a[arraysize] = { 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 };
int result = WhatIsThis (q, arraysize);
cout << " Result is: " << result << endl;
return 0;
}
int WhatIsThis (int b[ ], int size)
{
if (size == 1)
return b[0];
else
return b[size -1] +WhatIsThis[b, size -1];
}
```

## المراجع

### مراجع باللغة العربية

- 1 - أحمد عبادة سرحان، طرق التحليل الإحصائي، دار الكتاب الجامعي، القاهرة، 1970.
- 2 - أحمد عبدالعظيم الدغيدى، برمجيات الحاسب الآلى التطبيقات التجارية والإحصائية، كلية التجارة، جامعة جنوب الوادى، 2010.
- 3 - احمد عبد العظيم الدغيدى، صدام حسين، برمجيات الحاسب الآلى وتطبيقاتها التجارية، كلية التجارة، جامعة جنوب الوادى، 2017.
- 4 - مصطفى جلال مصطفى، محمد محمود خطاب، مقدمة فى التحليل الإحصائي، بدون ناشر، 2004.

### مراجع باللغة الإنجليزية

- 1- James T. and Benson P., Statistics for Business and Economics, Macmillan Collage Publishing Co., USA, 1993.
- 2 - Deitel, Paul, and Harvey Deitel, C++ How to Program, 7th Edition, Prentice Hall, 2010.
- 3 - Musaed Alhussein, Learning C++ Basics, King Saud University, 2010.
- 4- Robert D. and Douglas A., Statistical Techniques in Business and Economics, S. Ninth ed., Irwn C., Chicago, 1996.
- 5 - SAS Institute, Inc, SAS/STAT User's Guide, Volume 2, GLM-VARCOMP, Version 6, Fourth Edition, Cary, NC: SAS Institute, Inc., 1990.



# المحتويات

رقم الصفحة	الموضوع
3	مقدمة
5	الباب الأول اساسيات البرمجة
7	الفصل الأول لغات الحاسب الآلى
17	الفصل الثانى خرائط التدفق
45	الباب الثانى التطبيقات التجارية للغة ++C
47	الفصل الأول أسس البرمجة بلغة ++C
73	الفصل الثانى الجمل الشرطية
85	الفصل الثالث أوامر التكرار
105	الفصل الرابع المصفوفات والمؤشرات
139	الفصل الخامس الدوال
157	الباب الثالث التطبيقات التجارية لبرنامج SAS
159	الفصل الأول مكونات ونوافذ حزمة SAS
177	الفصل الثانى مقاييس وتوزيعات واختبارات إحصائية
225	الفصل الثالث اجراء الاختبارات الإحصائية باستخدام SAS
245	الفصل الرابع التحليل الإحصائي باستخدام قوائم ساس والبرمجة
283	الفصل الخامس التحليل الاحصائى باستخدام الاشكال البيانية
301	تدريبات عامة
311	المراجع