# الباب الأول

# مقدمة البرمجة



## 1.1 مقدمة

شارك التقدم العلمي في جميع المجالات في تبلور الحضارة التقنية الحديثة حتى بلغت الحد الذي نعيشه اليوم وما أدى إليه ذلك التطور من تسهيل كثير من شؤون الناس اليومية على الصعيد الفردي والمجتمعي المتمثلة في الاتصالات والمواصلات والبحث العلمي والمجالات الطبية والهندسية المختلفة وغيرها مما يصعب حصره.

ظهور الحاسوب سبب تحولا هائلا في مسار ذلك التطور ونقلة ملفتة للتقدم العلمي والتقني، وإذا سبرنا أهم الخدمات التي يوفرها الحاسوب والتي لا يكاد يخلو منها أيّ من المجالات التي أشرت إلى بعضها سنجد أنها البرمجة؛ لذلك اهتمت الشركات بتوفير بيئات مختلفة لتمكين المبرمجين من بناء برامجهم التي تخدم شتى التطبيقات.

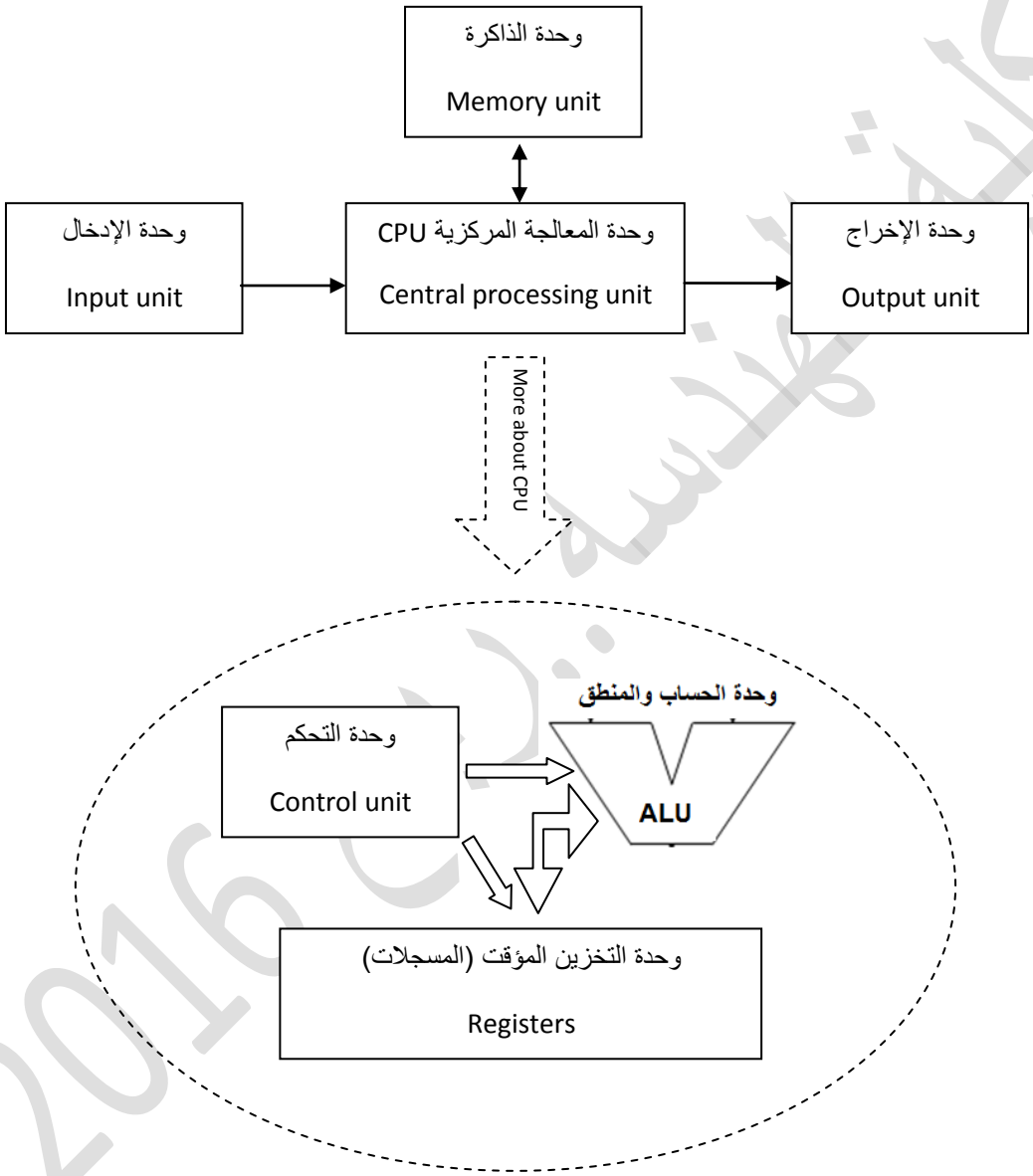
تتفاوت لغات البرمجة في الإمكانات التي تقدمها للمبرمج من حيث عدد الدوال والأوامر وغيرها، كما تختلف في الجانب الذي تدعمه فمنها ما يدعم الجانب الرياضي ومنها الإحصائي أو الهندسي إلى غير ذلك. ومن أشهر اللغات التي تخدم الجانب الهندسي والتي اهتم بها المهندسون على اختلاف تخصصاتهم لغة C؛ لذا فقد اعتُمد في هذا المقرر أحد صيغ مترجمات هذه اللغة وهو C++.

يهدف هذا المقرر إلى إكساب الطالب المهارة الكافية لإنشاء البرامج المختلفة بلغة البرمجة C++ نظريا وعمليا مع مراعاة التسلسل العلمي للمنهج بما يضمن إلى حد كبير حصول الفائدة المبتغاة منه.

# 2.1 مفاهيم عامة

## 1.2.1 الأجزاء الرئيسية للحاسوب

تجدر الإشارة في مطلع هذا المقرر إلى المكونات الأساسية للحاسوب، ويمكن إجمال الأهم في ذلك من خلال المخطط الكتلي التالي:




**شكل (1): مخطط عام لأهم مكونات الحاسوب**

## 2.2.1 البرنامج program:

هو مجموعة متسلسلة من الأوامر والجمل الحسابية والمنطقية لتحقيق غرض معين.

### 3.2.1 لغات البرمج programming languages:

هي عبارة عن بيئات برمجية تسمح للمبرمج بكتابة برنامجه على الحاسوب وتنفيذه والتنبيه على الأخطاء البرمجية. تقوم لغات البرمجة كذلك بتحويل البرنامج إلى شفرات تناسب فيزيائية المعالج عن طريق برنامج المترجم.

عند الحاجة إلى برمجة بعض المتحكمات الدقيقة (Microcontrollers) وخاصة القديمة منها فإن البرنامج سوف يحتوي على أوامر مباشرة للمعالج (أي أن كل أمر يقابل شفرة تعني عمل واحد للمعالج ولأجزاء التنفيذ الفيزيائية) تسمى لغة البرمجة والحال هذه بلغة التجميع (Assembly languages) وهذا النوع من لغات البرمجة اصطلح على تسميته بلغات منخفضة المستوى (Low level language). في حين أن اللغات عالية المستوى (High level languages) تحتوي على مصطلحات أقرب إلى اللغة البشرية، ومعظم الأوامر المستخدمة فيها لا تقابل شفرة تنفيذية مباشرة من لغة الآلة بل برنامجا تابعا يعرف بـ (Microprogram)، وهذا النوع من اللغات أسهل في البرمجة وأعلى بكثير في الإمكانيات، ومن أمثلتها لغة سي ولغة بيسك ولغة دلفي ولغة فورتران وغيرها كثير.

### 4.2.1 المترجم compiler:

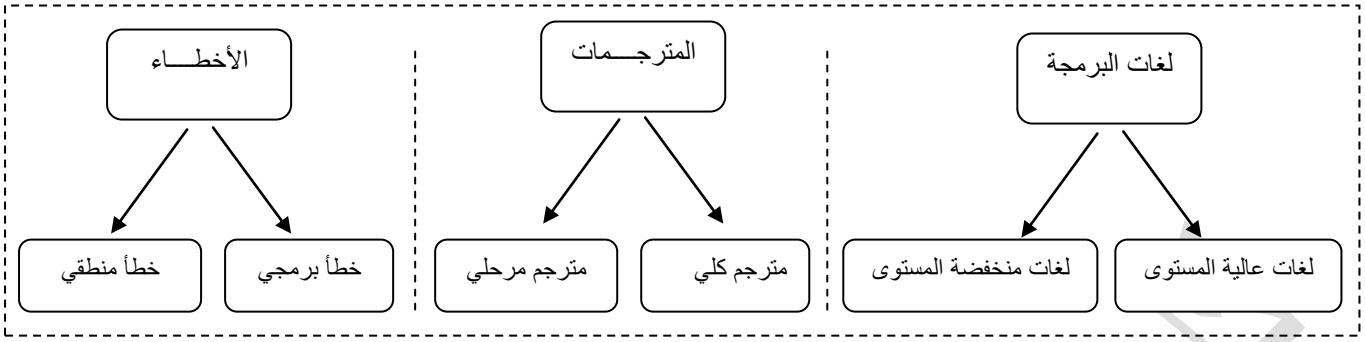
هو عبارة عن برنامج تعده شركات البرمجة لتحويل البرنامج المعد بلغة البرمجة إلى شفرات ثنائية يمكن للمعالج الدقيق أن يتعامل معها. بعض هذه المترجمات يقوم بترجمة البرنامج دفعة واحدة بعد الانتهاء من برمجته، والبعض الآخر يقوم بترجمة الأوامر والخطوات عند كتابة البرنامج واحدة تلو الأخرى ويعرف بالمفسر (Interpreter).

في الواقع فإن المعالج بوحداته الرئيسية المشار إليها في الشكل (1) والمتمثلة في وحدة التحكم ووحدة الحساب والمنطق ووحدة التخزين المؤقت وباقي الدوائر البينية ودوائر العنونة هو الذي يسند إليه التنفيذ المباشر للأوامر والتعليمات التي يتألف منها البرنامج، وسوف يتسنى للطالب الفهم الدقيق لآلية التنفيذ الحقيقية والاقتراب أكثر من تعامل معالج الحاسوب مع البرنامج عند دراسته لمقرر المعالجات الدقيقة.

**5.2.1 الأخطاء Errors:** عند كتابة البرنامج يكون من المتوقع جدا حدوث أخطاء في الكتابة أو في ترتيب الأوامر أو في الإدخال، معظم هذه الأخطاء يقوم المترجم باكتشافها وتحديد أماكنها، ولا يبقى على المبرمج إلا أن يصلحها، وهذا النوع من الأخطاء يعرف بالخطأ البرمجي أو النصي (Syntax error).

ولكن قد يكون الخطأ غير مخالف لقواعد البرمجة ولكن البرنامج لا يعطي النتائج المتوقعة، وهذا مثل الخطأ في كتابة معادلة رياضية معينة مثلا، في هذه الحال لا يقوم المترجم بالتنبيه على الخطأ ويعود اكتشافه إلى مهارة

المبرمج، وهذا النوع يعرف بالخطأ المنطقي (Logical error). ولو قلنا إن هذا النوع من الأخطاء أكثر اعتمادا في الحكم على مدى كفاءة المبرمج لكان لهاذا القول وجه.




**شكل (2): ملخص لبعض المفاهيم**

## 6.2.1 مراحل إعداد البرنامج:

يعد الوقت من أهم العوامل التي ينبغي للمهندس أن يعتبرها في تصميمه، بالإضافة إلى الجودة وقلة الموارد، إلا أن الاستعجال في كتابة أي برنامج مهما كان بسيطا قد يؤدي إلى الوقوع في أخطاء مما يؤدي بدوره إلى التأخير الذي لم يكن ليحصل لو أن المبرمج تتبع الخطوات النموذجية لإنشاء البرنامج والتي تتلخص إجمالا في النقاط التالية:

1. تحديد المسألة: والمتمثل في معرفة المعطيات والمطلوب والإمكانات المتوفرة للبرمجة، كما يشمل في بعض التطبيقات المتقدمة زمن التنفيذ وحيز التخزين.
2. تحليل المسألة: هذه المرحلة تتوقف على نوع وتعقيد المسألة المراد برمجتها، حيث تحتاج بعض المسائل إلى تحليل رياضي أو إحصائي معين حسب المطلوب.
3. تصميم البرنامج: والمراد به ترتيب خطوات البرنامج على هيئة خوارزمية أو مخطط انسيابي.
4. كتابة البرنامج بلغة البرمجة المناسبة والتي تتناسب امكاناتها مع المطلوب من البرنامج.

سبقت الإشارة إلى أن لغة البرمجة المقررة في هذا الفصل هي لغة (C++)، إلا أن البرمجة فعليا بهذه اللغة سيكون في أبواب لاحقة، أما الباب التالي فسيكون في المراحل الثلاث الأولى من إعداد البرنامج للتمرين على تحليل المسائل وإعداد الخوارزميات والمخططات لها؛ ليتسنى للطالب بعد ذلك تحويل هذه الخطوات إلى برنامج فعلي من خلال نقل كل خطوة في خوارزمية الحل إلى ما يقابلها من لغة البرمجة مع مراعاة قواعد تلك اللغة.

# الباب الثاني



## الخوارزميات والمخططات الانسيابية

## Algorithms & Flowcharts

بعد أن يتم تحديد مدخلات ومخرجات المسألة والطريقة الأفضل للوصول إلى الحل تأتي مرحلة ترتيب خطوات الحل بأفضل الطرق، وهو ما يعرف بالخوارزمية. هذه التسمية هي نسبة للعالم محمد الخوارزمي الذي سبق إلى وضع تسلسل لحل المسائل الجبرية في خطوات مرتبة بما يضمن الوصول لحل تلك المسائل بشكل آمن وسليم.

وظيفة الخوارزمية هي توضيح ترتيب الخطوات بغض النظر عن قواعد البرمجة؛ لذا لا يجب تحديد ألفاظ معينة فيها ولا توجد شروط معينة في الرموز أو المتغيرات المستخدمة في هذه المرحلة، إلا أنه جرى العمل بالتزام أن تبدأ الخوارزمية بخطوة تفيد بداية البرنامج وتنتهي بالتوقف.

### 1.2 أمثلة على الخوارزميات والمخططات الانسيابية

**مثال 1.2 :**

خوارزمية الحل لإدخال قيمتين وإخراج حاصل جمعهما.

1.. ابدأ

2.. أدخل العددين a و b

3.. احسب c=a+b

4.. أخرج c

5.. توقف

| |
|---|
| الدخل: |
| العددان aو b |
| الخرج: |
| المجموع c |

**تمرين 1.2 :**

ماذا لو كان المطلوب في المثال السابق المجموع وحاصل الضرب والمتوسط؟

**مثال 2.2 :** خوارزمية الحل لحساب وطباعة مساحة ومحيط دائرة نصف قطرها (r).

الدخل:
نصف القطر r
الخرج:
المساحة A و المحيط c

1.. ابدأ

2.. أدخل نصف القطر (r)

3.. احسب المساحة A = 3.14*r^2

4.. احسب المحيط c = 2*3.14*r

5.. أخرج c و A

6.. توقف

**ملاحظة**:

- المدخلات المتعددة يمكن جمعها في خطوة واحدة وكذلك المخرجات المتعددة، أما المعالجة فيفضل جعل كل مرحلة في خطوة مستقلة. وهذا ليس ملزما في الخوارزميات ولكن يفضل أن تكون الخوارزمية محاكية للبرنامج الفعلي قدر الإمكان.
- الترتيب المنطقي لأي برنامج هو أن تكون الخطوات مقسمة إلى ثلاث مراحل متعاقبة، وهي الإدخال ثم المعالجة ثم الإخراج.

**مثال 3.2 :**

خوارزمية الحل لحساب وطباعة القيمة الأكبر من بين قيمتين مدخلتين.

الدخل:
القيمتان x و y
الخرج:
الأكبر x أو y

1.. ابدأ

2.. أدخل القيمتين x و y

3.. إذا كان x>y

اطبع x

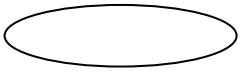
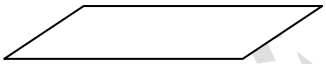
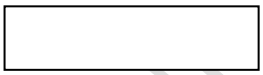
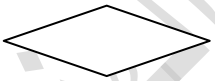
وإلا اطبع y

4.. توقف

**تمرين 2.2 :** ماذا لو كان المطلوب في المثال السابق الأكبر بين ثلاث قيم؟

## 2.2 المخططات الانسيابية Flowcharts

تستخدم المخططات الانسيابية كوسيلة لتوضيح خطوات البرنامج بحيث تكون أيسر في التتبع، بحيث تستخدم الأشكال الهندسية كدلالة على وظيفة الخطوة على النحو المبين في الجدول التالي:

| الشكل | ما يشير إليه |
|---|---|
| الشكل البيضاوي  | بداية ونهاية البرنامج |
| متوازي الأضلاع  | لعمليتي الإدخال والإخراج |
| المستطيل  | لعمليات المعالجة الحسابية |
| المعين  | لعمليات المعالجة المنطقية |
| الشكل  | للإشارة إلى أن المخطط يتبع |
| دائرة بها رقم أو علامة  | للربط بمرحلة سابقة أو تالية |
| السهم  | لربط المراحل المتسلسلة للمخطط |

جدول (1) : وظائف الأشكال المختلفة في لمخطط الانسيابي

**مثال 4.2 :** خوارزمية الحل والمخطط الانسيابي لطباعة الأعداد من أي عدد مدخل إلى ثلاثة أضعافه.

1.. ابدأ

2.. أدخل العدد a

3.. دع b=a*3

4.. اطبع a

5.. دع a = a+1

6.. إذا كان a<=b اذهب إلى الخطوة 4

7.. توقف

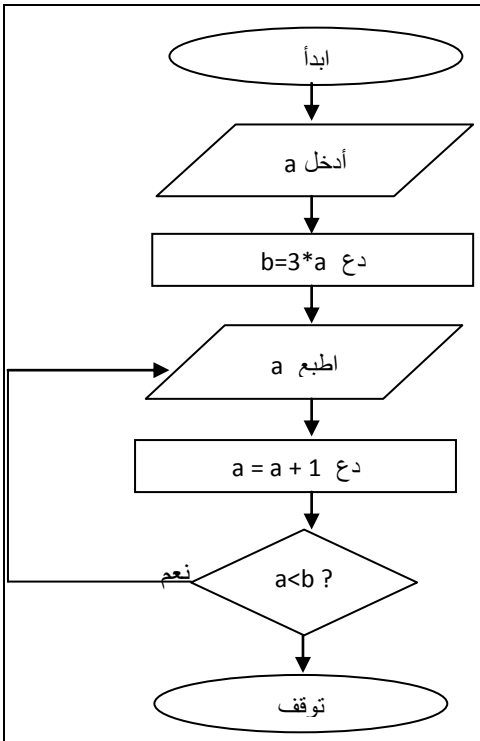
الدخل:

العدد a

الخرج:

الأعداد من a إلى 3a

نلاحظ في هذا المثال استخدام ما يعرف بالعداد (التكرار) والذي يستخدم عند الحاجة إلى تكرار خطوة أو أكثر لعدد من المرات بحيث يحتوي أي عداد في البرنامج على الخطوات الاساسية التالية:




- ✓ القيمة الابتدائية ...
- ✓ الإجراء المراد تكراره ...
- ✓ الزيادة ...
- ✓ شرط انتهاء تكرار العداد...

قد يكون استخدام العداد اختياريا وقد يكون إجباريا كما في المثال السابق.

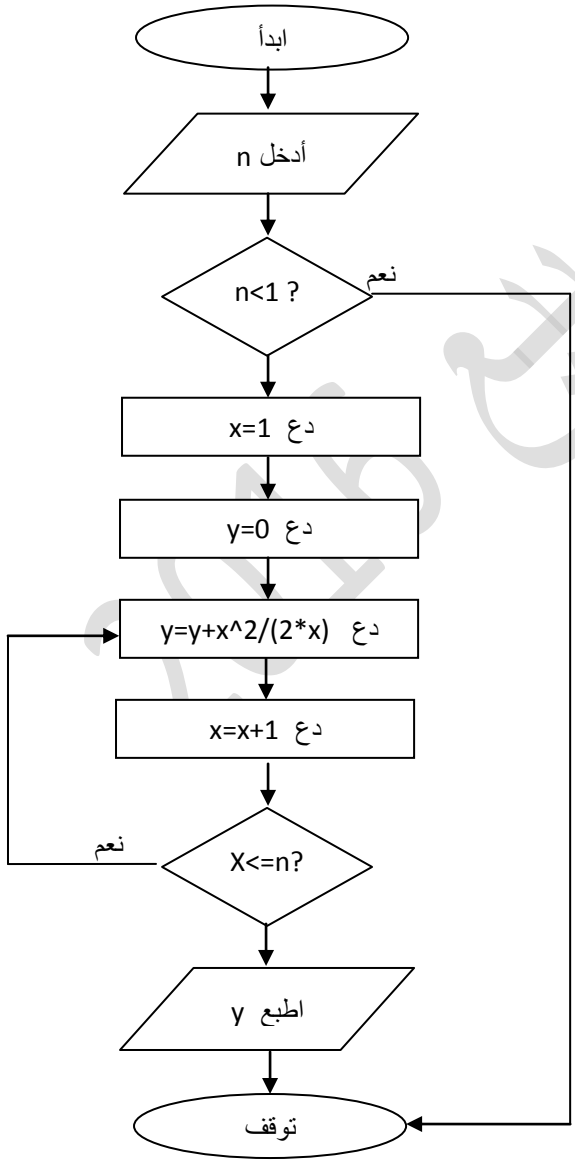
**تمرين 3.2 :** ماذا لو كان المطلوب في المثال السابق طباعة القيم الزوجية فقط؟

**مثال 5.2 :**المخطط الانسيابي لحساب وطباعة ناتج المتسلسلة التالية:

$$y = \sum_{x=1}^{n} \frac{x^2}{2x} \quad , \; n > x$$

الدخل: العدد n

الخرج: المجموع التراكمي y حسب المعادلة المعطاة




1.. ابدأ

2.. أدخل العدد n

3.. إذا كان n<1 اذهب إلى الخطوة 10

4.. دع x=1

5.. دع y=0

6.. دع y = y+x^2/(2*x)

7.. دع x = x+1

8.. إذا كان x <= n اذهب إلى الخطوة 6

9.. اطبع قيمة y

10.. توقف

**مثال 6.2 :**

طباعة نتيجة (ناجح أو راسب) لعدد من الطلبة بناء على متوسط 10 درجات لكل منهم.

الدخل: درجات المواد m

الخرج: كلمة ناجح أو راسب لكل طالب

1.. ابدأ

2.. دع s=0

3.. دع L=1

4.. أدخل درجة الطالب m

5.. دع s=s+m

6.. دع L=L+1

7.. إذا كان L<= 10 اذهب إلى الخطوة 4

8.. دع av=s/10

9.. إذا كان av>=50 اطبع (ناجح)

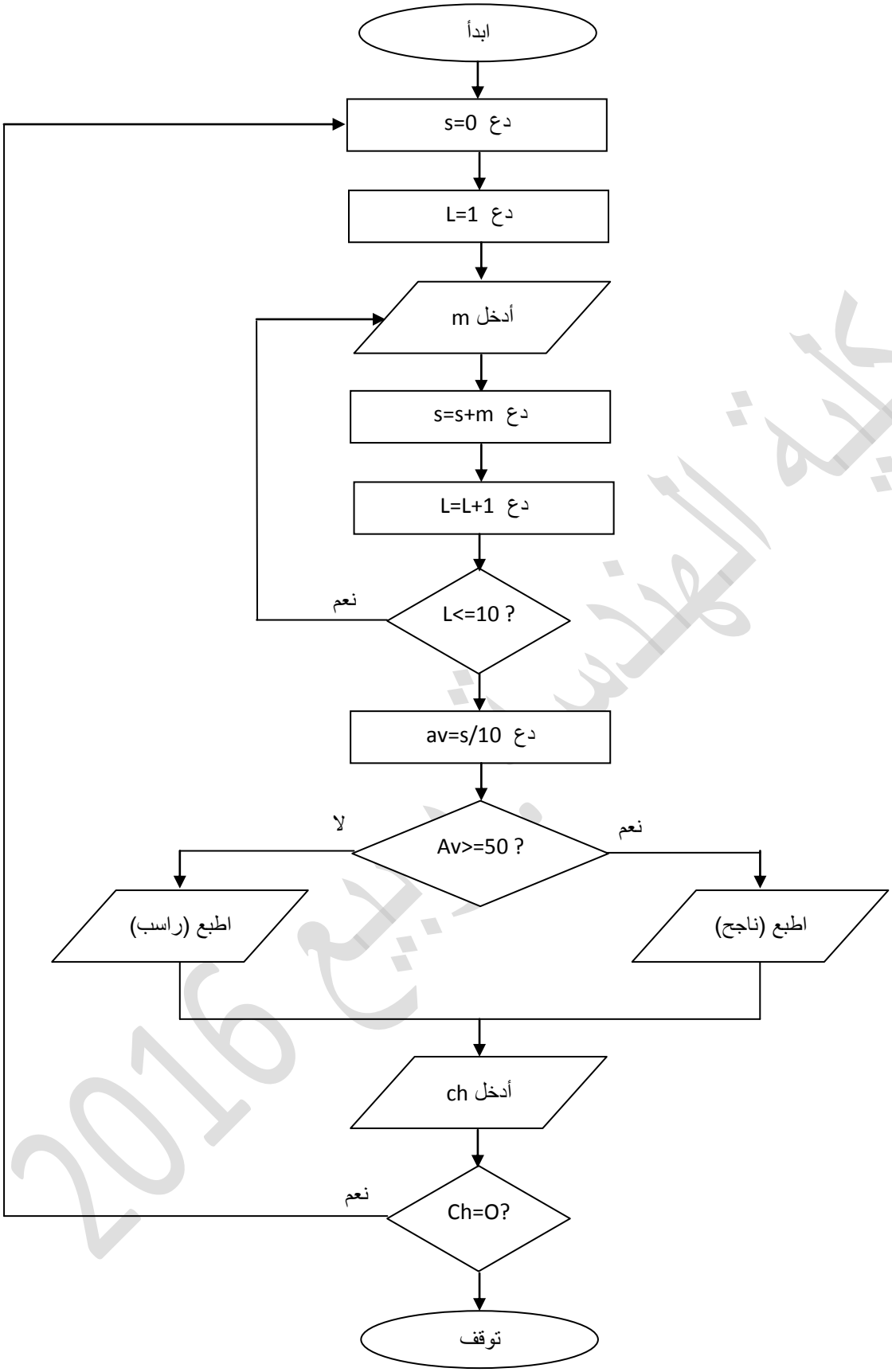
وإلا اطبع (راسب)

10.. أدخل ch ( O أو N)

11.. إذا كان ch=O اذهب إلى الخطوة 2

12.. توقف

والمخطط الانسيابي هو كما يلي:




**تمرين 4.2 :** ماذا لو كان المطلوب في المثال طباعة نسبة النجاح العامة؟

**ملاحظة:**

1 -نلاحظ في المثال السابق استخدام المدخل ch مع أنه غير مطلوب في نص المسألة. احتجنا لهذا المتغير لتحديد نهاية الإدخال؛ لأن عدد الطلبة غير محدد، بحيث يدخل مستخدم البرنامج الحرف O في حال الاستمرار في الإدخال والحرف N أو أي حرف آخر في حال انتهاء الإدخال. وهكذا ينبغي للمبرمج أن يتصرف بما يراه الأنسب لمصلحة البرنامج في حال عدم كفاية المعطيات، وحتى مع كفايتها يفضل للمبرمج الناجح أن يتصرف في البرنامج بحيث يجعله شاملا لحالات غير مطلوبة عند احتمال وقوعها أو الحاجة إليها.

2 -عادة ما يحتوي البرنامج وخاصة في البرامج الكبيرة (المنظومات) على واجهة توضح هذه الاستخدامات وكيفية التعاطي مع المنظومة، والأفضل أن تكون هذه المعلومات مختصرة ومضمنة في واجهة استخدام البرنامج (User interface).

**مثال 7.2 :**

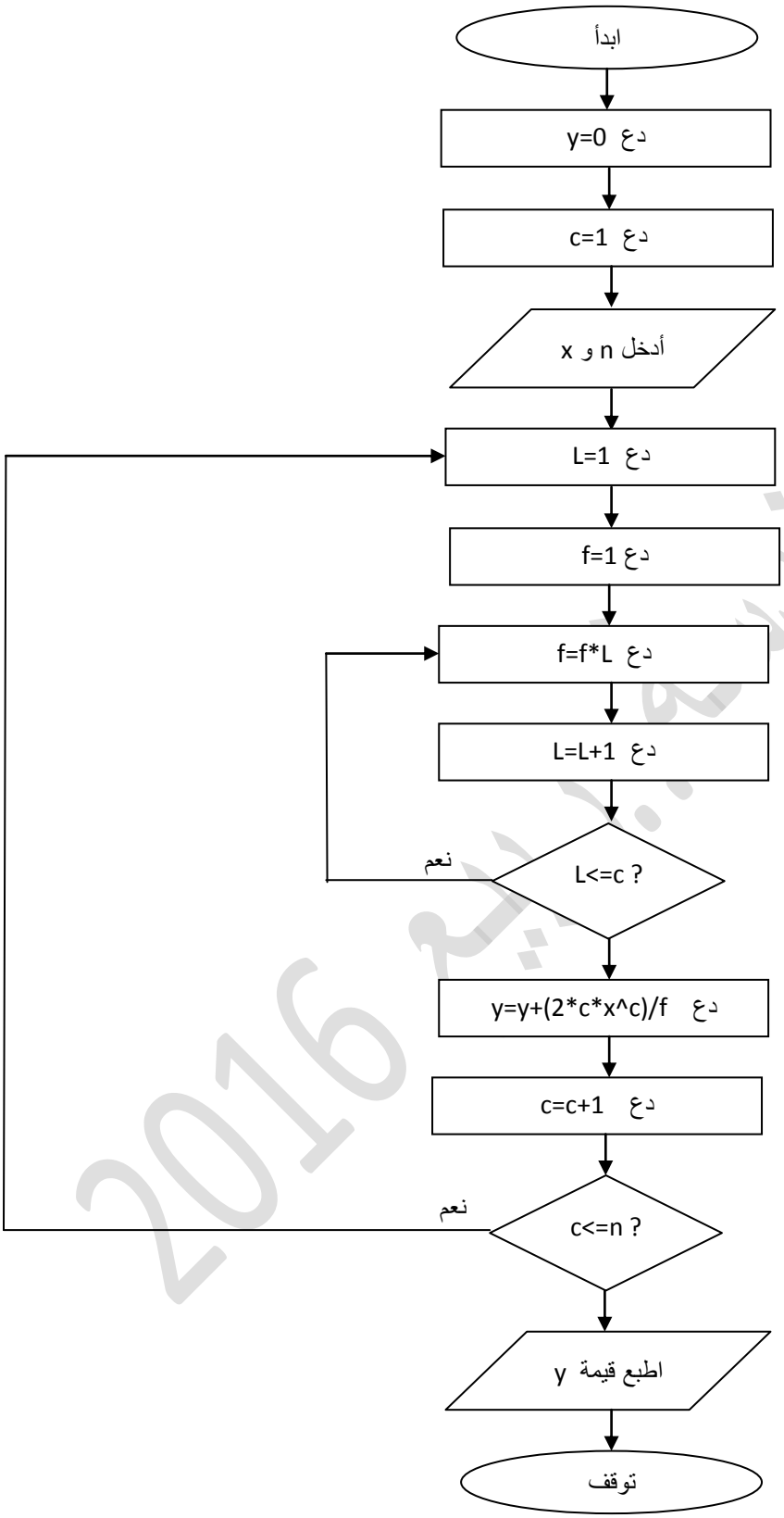
المخطط الانسيابي لحساب وطباعة ناتج المتسلسلة التالية:

$$y = \frac{2x}{1!} + \frac{4x^2}{2!} + \cdots \quad , n \geq 1$$

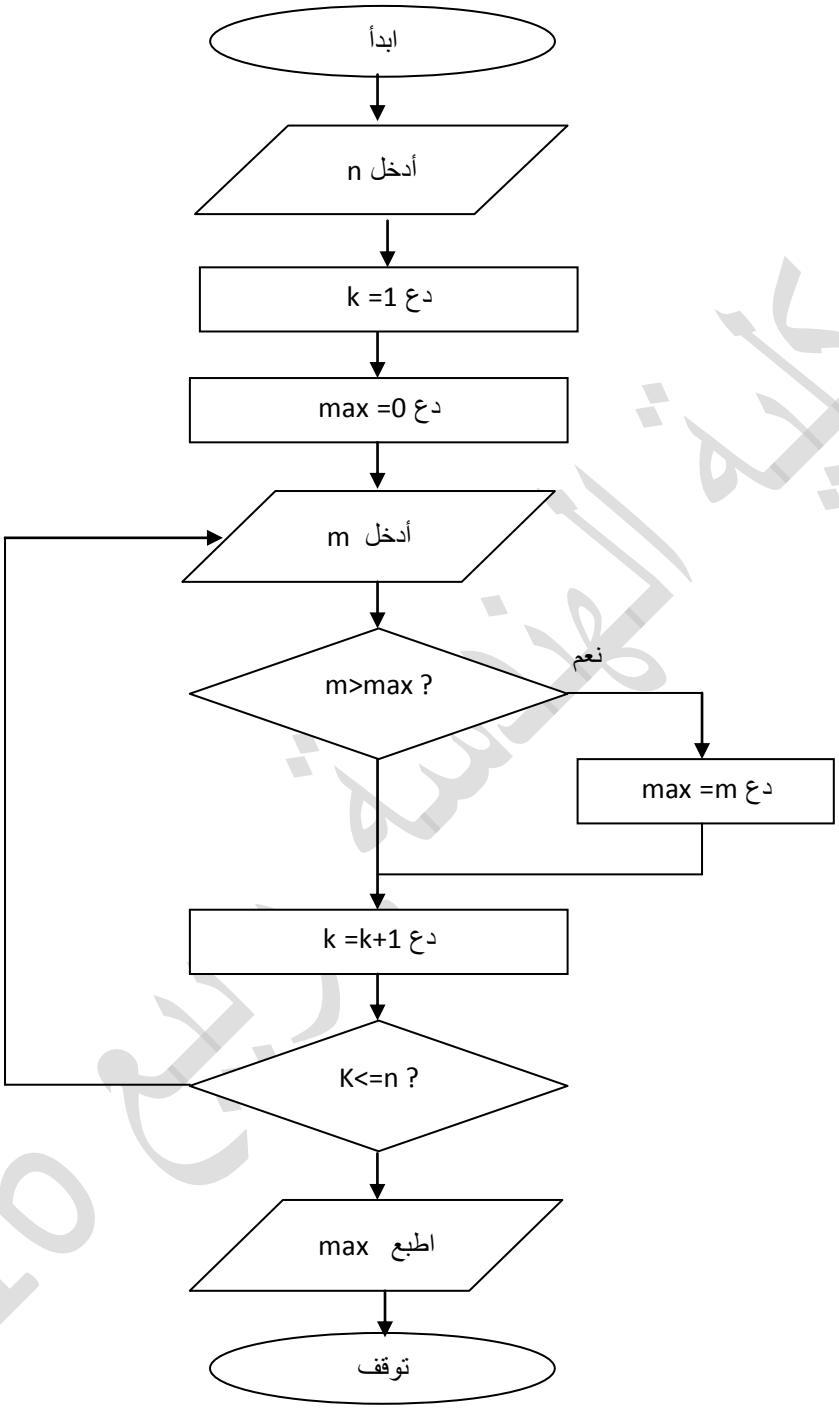
الدخل: قيمة n و x

الخرج: المجموع التراكمي y حسب المعادلة المعطاة

المخطط الانسيابي:



**مثال 8.2 :** المخطط الانسيابي لطباعة الأكبر بين n قيمة موجبة مدخلة:




**تمرين 5.2 :** ماذا لو لم يشترط في المثال السابق أن تكون القيم موجبة؟

**تمرين 6.2 :** ارسم المخطط الانسيابي لإدخال أسماء وأعمار عدد من الأشخاص وطباعة اسم وكلمة مقبول لكل من كان عمره بين 30 و 40.

# الباب الثالث



## أساسيات البرمجة بلغة C++

## Programming in C++

تشترك كل لغات البرمجة في بعض الأساسيات التي سنشير إلى ما يهم منها في هذه الوحدة من المنهج، والتي تتلخص في التعرف على الثوابت والمتغيرات وصياغة التعابير الحسابية والمنطقية برمجيا ونحو ذلك.

ومع اتفاق كل اللغات في ما أشرنا إليه إلا أنها تختلف في بعض الأمور من أهمها:

- ✓ الصيغة الكتابية الأوامر (التعليمات instruction) والجمل (Statements) وبعض الرموز والتعابيرالحسابية (mathematical excerptions)
- ✓ بعض شروط تسمية المتغيرات المختلفة، واستخدام الأحرف الكبيرة والصغيرة وغير ذلك.
- ✓ التفاوت في توفير الدوال الجاهزة لإنجاز المهام الرياضية والإجرائية المختلفة.
- ✓ اختلافها في توفير بيئات مناسبة للربط مع البرامج التطبيقية الأخرى مثل الإكسل والآكسس وغيرها.

### 1.3 الثوابت

المقصود بالثابت في لغات البرمجة عموما هو قيمة معينة ممثلة في رقم صحيح أو كسري أو نص. نقسم كل الثوابت ترجع إلى نوعين :عددية Numerical أو نصية Text. وتوفر لغة الـ C++ أنواعا من الثوابت منها:

- الثوابت العددية الصحيصة Integer

وهي الأرقام الصحيحة الموجبة والسالبة. ولها مدى معين حسب طول المسجل في نظام التشغيل المستخدم، فمثلا إذا كان طوله 32bit فإن مدى الأعداد الصحيحة يكون من 2147483648 - إلى 2147483648 كما يمكن تمثيل الأعداد في لغة C++ بأنظمة العد الأخرى مثل الثماني والست عشري.

- الثوابت العددية الحقيقية Float

وهي كل الأرقام التي تحتوي على فاصلة عشرية في ضمن المدى الذي يسمح به نطاق الذكرة، ومكن تمثيله بالصورة الاعتيادية مثل 22.34 أو بصورة المثال : 323e-4

- الثوابت الحرفية character

وهي أي حرف أو رمز يوضع بين علامتي التنصيص المفردة مثل 'x'. وفي الحقيقة يمثل هذا الثابت قيمة هذا الحرف في جدول آسكي ASCII code

- ثوابت السلاسل النصية String

وهي أي حرف أو رمز أو أكثر يوضع بين علامتي التنصيص مزدوجة مثل "xfg". وفي حالة طول السلسلة الحرفية وكتابتها في أكثر من سطر يمكن ذلك من خلال وضع العلامة \0 في آخر السطر السابق. كما أن المترجم يقوم بدمج السلسلتين الحرفيتين المتجاورتين دون فاصل ببعضهما:

"123" "abc" = "123abc"

ومما ينبغي ملاحظته أن المعالج يعين حرف إضافي في نهاية كل سلسلة حرفية كعلامة على انتهائه مما يجعل الثابت 'a' أقل في حيز الذاكرة من "a".

## 2.3 المتغيرات : Variables

يمكن تعريف المتغير على أنه اسم يختاره المبرمج لموقع في الذاكرة يخزن فيه قيمة ثابت.

ويفضل عند تسمية المتغيرات استخدام الكلمات بدلا من الحروف المفردة، وأن تكون هذه الأسماء معبرة عن ما تمثله قيمة المتغير المخزن فيها. ويشترط عند تسمية المتغيرات الشروط التالية:

- ✓ أن لا يبدأ برقم، ويجوز أن يكون الرقم في أثناء اسم المتغير أو في آخره.
- ✓ أن لا يحتوي على حرف عربي أو أي من رموز آسكي إلا علامة الربط (_).
- ✓ أن لا تستخدم في التسمية الكلمات المحجوزة لمترجم اللغة مثل: cout – cin – for – if …
- ✓ يفرق مترجم C++ بين الأحرف الكبيرة والصغيرة فمثلا: المتغير azs ليس هو نفسه Azs

فمن أمثلة أسماء المتغيرات المقبولة :

Abc , x_az , array , _age , qw23 , D …….

فمن أمثلة أسماء المتغيرات غير المقبولة :

2bc , x-az , array^ , ب , qw2.3 , D 5 …….

توفر لغة C++ عددا من أنواع المتغيرات حسب نوع الثابت المخزن فيها وأهمها:

- **المتغير العددي الصحيح Integer**

يخزن فيه ثابت عددي صحيح مثل age = -34;

- **المتغير العددي الكسري Float**

يخزن فيه ثابت عددي كسري مثل: mark = 79.75;

- **المتغير العددي الكسري المضاعف Double**

يخزن فيه ثابت عددي كسري دقيق (يمين الفاصلة أرقام أكثر) مثل: K = 9.74767427855745;

- **المتغير الحرفي Character**

يخزن فيه ثابت حرفي مثل: name = 'salim';

**ملاحظة**:

1 -في لغة C++ وفي كل أنظمة البرمجة فإن العلامة (=) لا تعد علامة مساواة كما هو الشأن في الرياضيات، بل تعتبر علامة تعيين Assignment أي أنها تعني الأمر بتخزين الثابت أو ناتج التعبير الحسابي أو قيمة المتغير الموجود على يمين الرمز (=) في المتغير الموجود على يسارها.

2 -كل جملة في لغة C++ يجب أن تنتهي بالفاصلة المنقوطة (;).

توجد أنواع أخرى للمتغيرات متفرعة من الأنواع السابقة ويمكن تطبيقها على المتغيرات العددية. هذه الأنواع الفرعية تسمى الواصفات، وهي:

- الطويل Long : يستخدم لتخزين ثابت صحيح أو جزء كسري له خانات كثيرة.
- القصير Short : يستخدم لتخزين ثابت صحيح أو جزء كسري له خانات قليلة.
- اعتبار الإشارة Signed : وضع هذا الواصف قبل المتغير العددي يدل على أنه يجب اعتبار الإشارة السالبة أو الموجبة لقيمة هذا المتغير.
- عدم اعتبار الإشارة Unsigned : وضع هذا الواصف قبل المتغير العددي يدل على إهمال الإشارة السالبة لقيمة هذا المتغير على كل حال.على عكس الوصف Signed

وفي حال عدم تحديد أحد الواصفين السابقين فإنه يتم تحديده تلقائيا (Signed).

- تعيين الثابت Const : يستخدم لغرض تعيين قيمة ثابتة لمتغير من أي نوع في مرحلة الإعلان عن نوعه، بحيث قيمة هذا المتغير ثابتة طوال فترة تنفيذ البرنامج.

## 3.3 الإعلان عن المتغيرات : Variables Declaration

من أساسيات البرمجة أن يتم الإعلان على المتغيرات التي ستستخدم في البرنامج وتحديد أنواعها في أول البرنامج. تختلف لغات البرمجة في طريقة الإعلان عن المتغيرات وتحديد الأنواع، وفيما يلي أمثلة شاملة لكل أنواع المتغيرات التي تم تناولها:

**مثال 1.3**

```
int count, num, rot=4;
long int pop;
short int abc;
float val1;
double length=0.00123432145 43;
char nam, word;
unsigned int val2;
```

كما يمكن تحديد طول المتغير الصحيح كما يتضح من خلال الأمثلة التالية:

**مثال 2.3**

```
Int8 a;
Int16 b;
Int32 c;
```

## 4.3 نظام المكتبات في لغة c++

تتميز لغة البرمجة هذه بميزة أخرى عن باقي اللغات وهي نظام المكتبات. من المعلوم أن من أهم ما يميز أي لغة برمجة هو احتواؤها على أكبر عدد من الدوال الجاهزة لاستدعائها من قبل المبرمج، وهذا مما يسهل على المبرمج ويقلل من تعقيد برنامجه ويجعله أكثر كفاءة. ولما كانت لغة C++ من اللغات التي تحتوي كما كبيرا من الدوال الجاهزة فقد تم ترتيب هذه الدوال في مجموعات حسب تصنيفها العلمي، وهذه المجموعات تسمى المكتبات. ومن أمثلتها :

- stdio هي المكتبة الأكثر استعمالا ولا يكاد يخلو منها برنامج؛ وذلك لأنها تحتوي على الدوال الأكثر شيوعا في البرامج مثل دوال الإدخال والإخراج والعشرات من الدوال الأساسية الأخرى.
- math تحتوي على الدوال الرياضية المختلفة.
- time تحتوي على دوال التعامل مع الوقت.
- string تحتوي على دوال التعامل مع السلاسل الحرفية.

يجب على المبرمج استدعاء المكتبة التي يعلم أنه سوف يستخدم أحدى الدوال الخاصة بها في برنامجه، وذلك من خلال كتابة اسم المكتبة في أول البرنامج بالصيغة التالية:

#Include <liprary name.h>

ومن أهم الدوال التي يجب استدعاؤها في أول كل برنامج هي الدالة main وهي الدالة الرئيسية التي تكتب فيها جميع البرامج، وتنتهي هذه الدالة بالأمر return (0) أي أن هذه الدالة لم تستخدم لإرجاع قيمة معينة. ونحتاج إلى هذا الأمر إذا تم عريف الدالة الرئيسية main من النوع الصحيح int أما إذا تم تعريفها من النوع المهمل void فلا نحتاج إليه. وسيأتي بيان كل هذا عند الكلام على الدوال الفرعية.

## 5.3 التعابير الحسابية في البرمجة:

تستخدم في لغة C++ العديد من الرموز الحسابية والمنطقية لصياغة التعابير المختلفة، فمن ذلك رموز العمليات المعتادة مثل (+ - * / ) ، كذلك تختص هذه اللغة بالكثير من الرموز والمعاملات التي تميزها عن باقي لغات البرمجة ومنها (التزايد ++ والتناقص -- ).

### 1.5.3 التحويل بين المتغيرات

مما يجب الانتباه إليه عند صياغة التعابير الحسابية أن المتغيرات تتحول إجباريا إلى ما يناسب بعض أنواع الحدود الأخرى فمثلا بالنسبة للأنواع العددية يتم التحويل إلى النوع الأكبر مطلقا كما يلي:

- إذا كان أحد الحدين double يتم تحويل الآخر إلى double
- إذا كان أحد الحدين long double يتم تحويل الآخر إلى long double
- إذا كان أحد الحدين long int والآخر unsigned int يتم تحويل الآخر إلى long int إذا كانت خانات الـ long int تستوعب القيمة التي ستتولد من التحويل وإلا يتحولان إلى unsigned int.
- ناتج العملية الحسابية يكون حسب النوع الأكبر إلا في حالة الإجبار على التحويل.

يمكن إجبار إي متغير على التحول من نوع إلى آخر من خلال كتابة اسم النوع قبله أثناء العملية ويسمى معامل التحويل Type cast operator كما يلي:

(Type) data

(Type) Variable

(Type) Expression

### 2.5.3 عملية التعيين Assignment operation

يقصد بعملية التعيين تخزين ثابت أو قيمة متعير أو ناتج تعبير رياضي في متغير آخر، وتتم هذه العملية بالرمز (=). هذا الرمز يعني رياضيا أن الطرف الأيمن يساوي الطرف الأيسر، أما برمجيا فهو أمر للتخزين المؤقت للطرف الأيمن في الطرف الأيسر أثناء البرنامج.

**مثال 3.3** أمثلة على عمليات التعيين

```
Name=”Khalid”;

Cost=net*0.56777 +1023;

A=b=c=d=4;

S=(d=4.5,d-2);
```

العمليتان الأوليان مباشرتان. في العملية الثالثة نلاحظ أن C++ توفر من خلال هذا التسلسل عدة أسطر في البرنامج، حيث يتم تخزين القيمة 4 لكل من d,c,b,a. أما العملية الرابعة فهي أيضا مما تتميز بها هذه اللغة حيث يتم في هذه الحالة إجراء العمليات داخل القوس ثم إسناد الناتج إلى المتغير في الطرف الأيسر بالتالي s=2.5

### 3.5.3 التزايد والتناقص increment & decrement

من خصائص لغة C++ أيضا عملية التزايد والتناقص عن طريق المعاملان (++ & --). أي زيادة أو نقص 1 للمتغير.

يوجد نوعان من هذه العملية هما الزيادة أو النقصان القبلي pre increment & decrement operator

مثل v-- , a++ وفي هذه الحالة تتم الزيادة أو النقصان للمتغير قبل تنفيذ التعبير الحسابي الذي يحتوي هذا المتغير. كما توجد عملية الزيادة أو النقصان البعدي post increment & decrement operator مثلا :

**مثال 4.3**

```
d=f=3;

a=3*++f+--d;
```

سوف تكون قيمة a النهائية هي 14

**مثال 5.3**

```
a=2;
b=3-(--a);
c=++a-(++b)+(--a);
```

ستكون القيمة النهائية لـ c,b,a على التوالي هي 1،3،0

**مثال 6.3**

```
a=2;
b=5%a++;
c=++b-1+a;
```

ستكون القيمة النهائية لـ c,b,a على التوالي هي 3،2،4

**ملاحظة:** الرمز % هو أحد الرموز المستخدمة في لغة C++ والذي يعطي باقي قسمة الطرف الأيسر على الطرف الأيمن كما في المثال السابق.

**مثال 7.3** لاحظ الفروق بين مقاطع البرامج التالية:

```
int a,b,c;
a=2;
b=5%a++;
c=++b-1+a/2.0;
```

a , b , c = 3 , 2 , 2

```
Int a,b;float c;
a=2;
b=5%a++;
c=++b-1+a/2;
```

a , b , c = 3 , 2 , 2

```
int a,b;float c;
a=2;
b=5%a++;
c=++b-1+a/2.0;
```

a , b , c = 3 , 2 , 2.5

```
int a,b;float c;
a=2;
b=5%a++;
c=++b-1+int(a/2.0);
```

a , b , c = 3 , 2 , 2

**ملاحظة :** عند التحويل من كسري إلى صحيح يتم قص القيمة الكسرية أما عند العكس فسيتم تحويله إلى كسري. أما عند التحويل من double إلى float فسيتم القص مع التقريب.

**مثال 8.3**

```
int k=4;
float l;
l=int (k+0.8);
k=l-1.7;
```

ستكون القيمة النهائية لـ k,l على التوالي هي 4 و 2

**مثال 9.3**

لاحظ الفرق بين مقطعي البرنامجين التاليين:

```
int i;
float j;
j=i=9.8;
```

i , j = 9 , 9

```
int i;
float j;
i=j=9.8;
```

i , j = 9 , 9.8

### 4.5.3 التعيين التراكمي

توفر لغة C++ إضافة إلى ما سبق ميزة تسرع من عملية كتابة البرامج كما يلي:

```
a=a+2    ------ in c++ ------    a+=2
a=a-2    ------ in c++ ------    a-=2
a=a*2    ------ in c++ ------    a*=2
a=a/2    ------ in c++ ------    a/=2
```

### 5.5.3 أولويات التنفيذ في العمليات الحسابية

عندما يقوم المعالج بتنفيذ التعابير الحسابية فإنه يقوم بذلك وفق أولوية معتمدة وذلك حسب الترتيب الآتي:

| الرمز | الوصف |
|---|---|
| ( ) | الأقواس - من اليسار إلى اليمين |
| ++ و -- | التزايد والتناقص البعدي - من اليسار إلى اليمين (التعويض في أثناء العملية بالقيمة السابقة، ويتم اعتماد القيمة الجديدة بعد الانتهاء من الخطوة قيد التنفيذ) |
| ++ و -- | التزايد والتناقص القبلي - من اليسار إلى اليمين |
| + و - | قلب الإشارة - من اليسار إلى اليمين |
| (type) | التحويل الإجباري لأنواع المتغيرات - من اليسار إلى اليمين |
| %و / و* | الضرب والقسمة وباقي القسمة - من اليسار إلى اليمين |
| + و - | الجمع والطرح - من اليسار إلى اليمين |
| = | التعيين - من اليسار إلى اليمين |
| =- و =+ .... | العمليات التراكمية - من اليسار إلى اليمين |

### 6.3 التعابير المنطقية

التعابير المنطقية هي التي يكون ناتجها إحدى قيمتين فقط هما صح (true) أو خطأ (false)، وفي لغة C++يعطى الناتج true قيمة (1) ، والناتج false قيمة (0). أهم رموز التعليمات المنطقية في C++ هي:

| الرمز | الوصف |
|---|---|
| && | يعني المنطق And – يعطي ناتج 1 عندما تكون كل التعابير المنطقية 1 |
| \|\| | يعني المنطق Or - يعطي ناتج 1 عندما تكون إي من التعابير المنطقية 1 |
| ! | يعني المنطق Not – يعكس القيمة منطقيا |

أما العلاقات المنطقية المستخدمة في لغة C++ فهي العلاقات المعتادة رياضيا وهي:

| الرمز | الوصف |
|---|---|
| == | يساوي |
| > | أكبر من |
| < | أصغر من |
| >= | أكبر أو يساوي |
| <= | أصغر أو يساوي |
| =! | لا يساوي |

**ملاحظة** : سنتناول الكثير من الأمثلة على التعليمات والعلاقات المنطقية لاحقا في موضوع الجمل الشرطية.

### 7.3 تعليمات التعامل على مستوى البت

تختص لغة C++ كذلك عن باقي لغات البرمجة بأنها لغة تتعامل على مستوى البت؛ أي يمكن للمبرمج إجراء العمليات المنطقية والبرمجية المختلفة على الخانات الثنائية للبيانات، وهذا يفتح أفقا كبيرا جدا على برمجة المعالجات الدقيقة والمتحكمات المنطقية؛ فلا غرو إذا من أن معظم البيئات البرمجية للمعالجات والمتحكمات المنطقية مبرمجة بلغة C أو برامج متفرعة منها. وهذه التعليمات مبينة في الجدول التالي:

| الرمز | الوصف |
|---|---|
| ~ | النفي – تستخدم لقلب الخانة من 0 إلى 1 والعكس. |
| & | تستخدم كبوابة منطقية AND بين كل بت من العامل الأول وما يقابلها من الرقم الثاني. |
| \| | تستخدم كبوابة منطقية OR بين كل بت من العامل الأول وما يقابلها من الرقم الثناي. |
| ^ | تستخدم كبوابة منطقية XOR بين كل بت من العامل الأول وما يقابلها من الرقم الثناي. |
| >> | لإزاحة اخانات الثنائية للرقم إلى اليمين لعدد من الخانات حسب المحدد في التعليمة |
| << | لإزاحة اخانات الثنائية للرقم إلى اليسار لعدد من الخانات حسب المحدد في التعليمة |

**مثال 10.3**

```
int x=5,y;
y=x<<1;
cout<<x<<'\t'<<y;
```

```
5       10
```

**مثال 11.3**

تتبع البرنامج التالي ولاحظ الخرج

```
int a=5,b=3,c,d,e;
c=a&b;
d=a|b;
e=a^b;
cout<< c<<'\n'<<'\n'<<d<<'\n'<<'\n'<<e<<'\n';
```

```
1

7

6
```

يمكن تحسين البرنامج لعرض المخرجات كالتالي:

```
int a=5,b=3,c,d,e;
c=a&b;
d=a|b;
e=a^b;
cout<<a<<'\t'<<b<<'\n'<<'\n';
cout<<"a&b="<< c<<'\n'<<'\n';
cout<<"a|b="<<d<<'\n'<<'\n';
cout<<"a^b="<<e<<'\n';
```

```
5       3
a&b=1
a|b =7
a^b =6
```

**مثال 11.3 :** لاحظ الفرق بين مقطعي البرنامجين التاليين واكتشف السبب في خرجيهما

| | |
|---|---|
| **int x=5,s=5,y;**<br>**y=x++&++x;**<br>**cout<<'\t'<<"x="<<x<<endl;**<br>**cout<<'\t'<<"y="<<y<<endl;**<br><br>X=7<br>Y=6 | **int x=5,s=5,y;**<br>**y=s++&++x;**<br>**cout<<'\t'<<"x="<<x<<endl;**<br>**cout<<'\t'<<"y="<<y<<endl;**<br><br>X=6<br>Y=4 |
| | |

البرنامج الأول واضح، أما الثاني فلأن x زادت 1 ثم عوض بها في المتغيرين بنفس القيمة لإيجاد y ثم زادت قيمة x. لأن أولوية = تسبق أولوية ++ التالية.

# الباب الرابع

## الإدخال والإخراج

## Input & output



توفر لغة C++ أكثر من طريقة لإدخال وإخراج البيانات.

### 1.4 الإدخال:

لإدخال البيانات يمكن استخدام الدالة scanf التي توفرها المكتبة stdio والدالة cin التيتوفرها المكتبة iostream. ولنعتمد هنا الدالة cin.

الصيغة العامة لها هي:

```
cin>>variable1>>variable2, …. ;
```

### 2.4 الإخراج:

لإخراج البيانات يمكن كذلك استخدام الأمر printf الذي توفره المكتبة stdio والأمر cout االذي توفره المكتبة iostream (وقد سبقت لنا بعض الأمثلة عليه). ولنستخدم هنا الأمر cout الذي له الصيغة العامة التالية:

```
cout << variable1<< variable1;
```

مع ملاحظة أنه يمكن إخراج القيم المباشرة أو قيم المتغيرات أوالنصوص على حد سواء.

### 3.4 مؤثرات الإخراج:

يمكن التحكم في شكل المخرجات على الشاشة من خلال أمر الإخراج مع كتابة بعض المؤثرات التي يبين الجدول التالي أهمها:

| الرمز | الوظيفة |
|---|---|
| \n | تنقل المؤشر إلى سطر جديد. |
| \a | إعطاء جرس تنبيه |
| \b | تنقل المؤشر حرف و احد إلى الخلف |
| \f | هذه العلامة تقوم بنقل المؤشر من الصفحة الحالية إلى بد اية صفحة جديدة وهذه غالباً ما تستخدم في التحكم في طباعة الملفات |

| | |
|---|---|
| \t | يقوم بنقل المؤشر بمقد ار ضغطة على مفتاح ال(tab) في لوحة المفاتيح |
| \“ | تقوم بطباعة(“) . كما يمكن طباعة أي علامة بهذه الطريقة. |
| \ddd | تقوم هذه العلامة بطباعة قيمة بالنظام الثماني. |
| \xdd | تقوم هذه العلامة بطباعة قيمة بالنظام الست عشري. |

كما يمكن التحكم في شكل المخرجات من خلال المؤثرات المبينة في الجدول التالي:

| | |
|---|---|
| Endl | Start new line |
| ends | Go to end of line |
| setw(int n) | Make n spaces |
| setfill(char c) | Fill spaces with character c |
| setprecision(int n) | Set precision of float numbers |
| width(int n) | Set the width of the output stream |

**مثال 1.4** لاحظ المخرجات التالية:

```
#include <iostream>

#include<iomanip>

using namespace std;

int main()

{     int n_i = 123;

      float n_f = 5.6;

cout<<"My name is Ali"<<ends<<"Her name is Fatima"<<ends<<"What is yours please?"<<endl;

return 0;}
```

**المخرجات:** **My name is Ali  Her name is Fatima  What is yours please?**

**مثال 2.4** لاحظ المخرجات التالية:

```
#include <iostream>
#include<iomanip>
using namespace std;
int main()
{     int n_i = 123;
      float n_f = 5.6;
      cout<<endl<<"The numbers you have are:";
      cout<<endl<<setw(10)<<setfill('*')<<n_i;
      cout<<endl<<setw(10)<<setfill('*')<<n_f;
      cout<<endl;
return 0;
}
```

**المخرجات:**

```
The numbers you have are:
*******123
*******5.6
```

**ملاحظة:**

1 -ستأتي الكثير من الأمثلة على الإدخال والإخراج في ما يلي من مواضيع.

2 -يوجد غير هذه الطريقة للإدخال والإخراج لكن باستخدام مكتبات أخرى كما سبقت الإشارة إليه.

# الباب الخامس

# الجمل الشرطية

## Conditional statement

كثيرا ما يحتاج المبرمج أثناء برنامجه اتخاذ قرار بين اختيارين بناء على شرط معين. توفر كل لغات البرمجة جملا برمجية مختلفة لهذا الغرض.

توفر لغة C++ ثلاثة أنواع من الجمل الشرطية هي:

- جملة إذا if
- جملة الاختيار المتعدد switch
- أداة الشرط ?

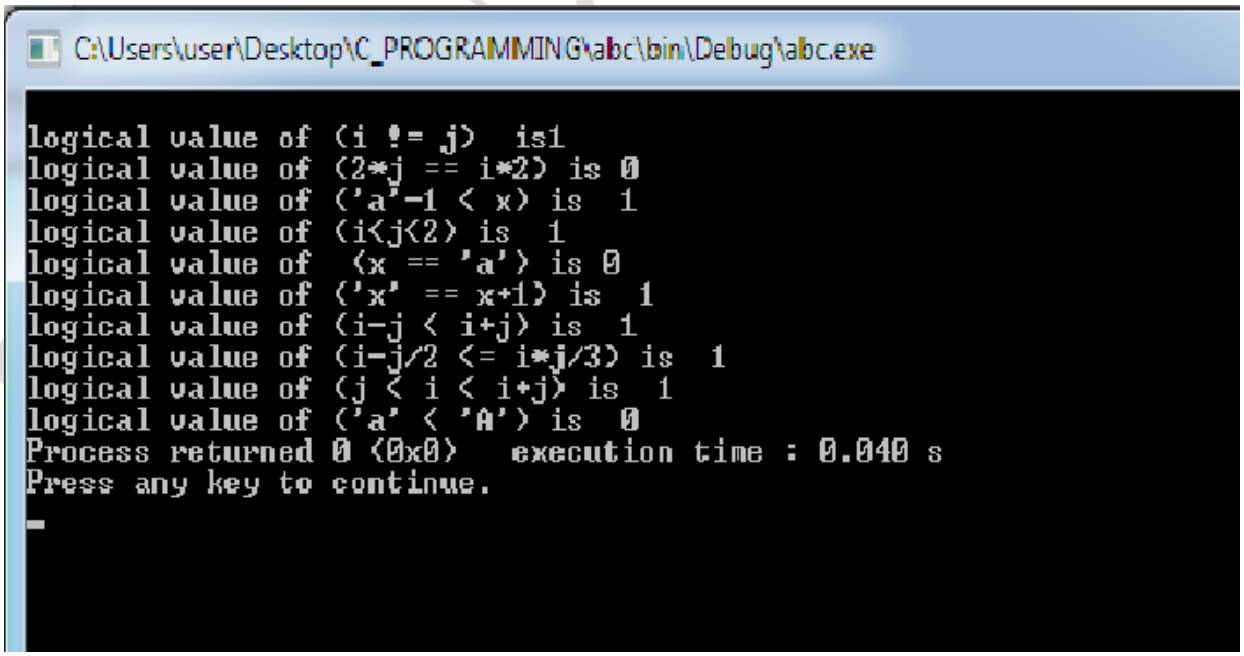
**ملاحظة**: التعابير الشرطية يتم التعويض عنها بـ 0 إذا كانت النتيجة خاطئة وبـ1 إذا كانت صحيحة.

**مثال 1.5** لاحظ نواتج البرنامج التالي:

```
#include <iostream>
#include<iomanip>
using namespace std;
int main()
{    char  x = 'w';
     int  i = -3;
     int  j =  5;
      cout<<endl<<"logical value of (i != j)  is"<<(i != j);
```

```
cout<<endl<<"logical value of (2*j == i*2) is "<<(2*j == i*2);

cout<<endl<<"logical value of ('a'-1 < x) is  "<<('a'-1 < x);

cout<<endl<<"logical value of (i<j<2) is  "<<(i<j<2);

cout<<endl<<"logical value of  (x == 'a') is "<<(x == 'a');

cout<<endl<<"logical value of ('x' == x+1) is  "<<('x' == x+1);

cout<<endl<<"logical value of (i-j < i+j) is  "<<(i-j < i+j);

cout<<endl<<"logical value of (i-j/2 <= i*j/3) is  "<<(i-j/2 <= i*j/3);

cout<<endl<<"logical value of (j < i < i+j) is  "<<(j < i < i+j);

   cout<<endl<<"logical value of ('a' < 'A') is  "<<('a' < 'A');

return 0;

}
```

المخرجات :




## 1.5 الجملة الشرطية IF

الصيغة المستخدمة فيما إذا كان المراد تطبيق قرار ما في حال تحقق الشرط، ثم الاستمرار في البرنامج على كل حال هي كالتالي:

```
If (condition)
{statement1;
Statement2;
 ….}
```

وتستخدم الصيغة التالية إذا كان المطلوب تنفيذ جملة واحدة:

```
If (condition)    statement;
```

أما إذا كان المطلوب تنفيذ أحد القرارين في حال تحقق الشرط وتنفيذ الآخر في حال عدم تحققه فإن الصيغة المستخدمة هي:

```
If (condition)
{statement1;
 Statement2;
 ….}
else
{statement1;
 Statement2;
 ….}
```

أو:

```
If (condition)
 Statement1;
else  statement2;
```

**مثال 1.5:** برنامج لطباعة الأكبر والأصغر من عددين مدخلين.

```
#include <iostream>

using namespace std;

//program for determining the maximum and minimum value.

int main()

{int x,y;

cin>>x>>y;

if (x>y)  cout<<x<<'\t'<<"is bigger than"<<'\t'<<y<<endl;

else cout<<y<<'\t'<<"is bigger than"<<'\t'<<x<<endl;

return 0;}
```

كما يمكن إنجاز هذا العمل من خلال البرنامج التالي:

```
#include <iostream>
using namespace std;
int main()
//program for determining the maximum and minimum value.
{int x,y;
cin>>x>>y;
if (x>y)  y=x;
cout<<y<<'\t'<<"is bigger <<endl;
return 0;}
```

**ملاحظة على البرنامجين السابقين**

استخدام التعليقات في البرنامج من الأهمية بمكان حيث ترشد المستخدم والذي يريد تطوير البرنامج إلى وظيفة البرنامج وغير ذلك. ويستخدم لهذا الغرض العلامة // في بداية السطر. أما إذا كان التعليق يأخذ أكثر من سطر فتوضع العلامة */ في أول الملاحظة والعلامة /* في آخرها.

**مثال 2.5**

برنامج لحساب الأصغر من بين ثلاثة أعداد مدخلة

```
#include <iostream>

using namespace std;

 //program for determining the minimum of three values.

int main()

{int x,y,z;

cin>>x>>y>>z;

if (x>y)  x=y;

if (x>z)  x=z;

cout<<"the least is"<<'\t'<<x;

return 0;

}
```

**مثال 3.5**

برنامج لحساب وطباعة نسبة وتقدير الطالب بناء على متوسط درجاته في ثلاث مواد.

```
#include <iostream>

using namespace std;

int main()

{int mark1,mark2,mark3;

float avr;

string got;

cout<<"input the three marks";
```

```
cin>>mark1>>mark2>>mark3;
avr=(mark1+mark2+mark3)/3.0;
if (avr>=85)
got="Excellent";
else if (avr>=75)
got="very good";
else if (avr>=65)
got="good";
else if (avr>=50)
got="satesfied";
else got="fail";
cout<<"he / she is"<<'\t'<<avr<<'\t'<<got;
return 0;
}
```

**تمرين 1.5 أعد حل المثال السابق بدون استخدام else.**

**مثال 4.5** برنامج لتمييز نوع المادة من حيث التوصيل وعدمه حيث:

-يتميز الموصل بكبر حجم الذرة وقلة إلكترونات التكافؤ (أقل من 4) وقوة الترابط بين الذرات.
-يتميز شبه الموصل بحجم متوسط للذرة وأن إلكترونات التكافؤ = 4 وقوة ترابط متوسطة.
-يتميز العازل بصغر حجم الذرة وكثرة إلكترونات التكافؤ (أكثر من 4) وضعف الترابط.

اكتب برنامجا لتحديد نوع المادة وفقا للمتغيرات الثلاثة السابقة.

```
#include <iostream>
```

```
using namespace std;

 /* program for determining the type of the material according to the
 number of electrons, atoms interconnection and atom volume */

int main()

{int no_of_elec;

string atom_volume,connection,type;

cout<<"input the number of electrons";

cin>>no_of_elec;

cout<<"input the atom volume as"<<'\t'<<"big or small or Medium ";

cin>>atom_volume;

cout<<"input the interconnection as"<<'\t'<<"strong or week or Medium ";

cin>>connection;

if ((no_of_elec<4)&&(atom_volume=="big")&&(connection=="strong"))

type="conductor";

elseif
((no_of_elec==4)&&(atom_volume=="Medium")&&(connection=="Medium"))

type="semiconductor";

elseif ((no_of_elec>4)&&(atom_volume=="small")&&(connection=="week"))

type="insolator";

else

{cout<<'\n'<<'\n'<<"this material has not the straight behaviour of the three types"<<'\n';
```

```
goto abc;}
cout<<'\n'<<'\n'<<"this material is "<<'\t'<<type<<'\n';
abc:
return 0;
}
```

**ملاحظة على المثال السابق**

-يستحسن دائما تضمين البرنامج بما يرشد المستخدم له بحيث يعرف المستخدم عمل البرنامج والمدخلات التي يقوم بإدخالها.

-نلاحظ استخدام أمر القفز غير المشروط بالصيغة الموضحة في البرنامج.

**مثال 5.5** برنامج لتمييز الحرف أو الرمز أو الرقم المدخل:

```
#include <iostream>
#include<iomanip>
using namespace std;
int main()
{   char ch;
cout<<"Enter a character:";
cin>>ch;
if((ch>='a')&&(ch<='z'))
    cout<<"The character ["<<ch<<"] is small character"<<endl;
else
    if((ch>='A')&&(ch<='Z'))
        cout<<"The character you type ["<<ch<<"] is capital character"<<endl;
```

```
    else
        if((ch>='0')&&(ch<='9'))
            cout<<"The character you type ["<<ch<<"] is digit character"<<endl;
        else
         cout<<"The character you type ["<<ch<<"] is special character"<<endl;
cout<<"All done"<<endl;
return 0;
}
```

### 2.5 جملة الاختيار المتعدد

تستخدم هذه الجملة فيما إذا ترتب على الشرط عدة اختيارات (أكثر من تفرعين)، وتأخذ الصيغة العامة التالية:

```
Switch (expression)
{case  constant1: statement1; break;
case  constant2: statement2; break;
….
case  constantn: statementn; break;
default:  otherwise;
}
```

**مثال 6.5**

برنامج لتحديد اسم المدينة بناء على إدخال الحرف الأول منها.

```
#include <iostream>
using namespace std;
// programe for specifying  the name of city
```

```
main()
{char ch;
cout<<"Enter the first letter"<<'\t';
cin>>ch;
switch (ch)
{
case 'm' :cout<<"misrata";
break;
case 't': cout<<"tripoli";
break;
case 'b': cout<<"bengazi";
break;
case 's': cout<<"sabha";
break;
default:cout<<”error”;
}}
```

**مثال 7.5**

برنامج لمحاكاة آلة حاسبة بسيطة لها إمكانية العمليات الأساسية البسيطة الأربعة.

```
#include <iostream>
using namespace std;
// programe for a simple calculator
main()
{int a,b;
float c;
char d;
cout<<"enter the two numbers"<<'\t';
cin>>a>>b;
cout<<"enter the symbol of the operation as + _ * / :"<<'\t';
cin>>d;
```

```
switch (d)
{
case '+': c=a+b;
break;
case '-': c=a-b;
break;
case '*': c=a*b;
break;
case '/': c=a/b;
break;
}
cout<<"the result is"<<'\t'<<c;}
```

**ملاحظة على المثال السابق**

نلاحظ أنه عندما لا نعرف الدالة main كـ int فإن البرنامج لا يشترط كتابة return0.

**تمرين 2.5**

أعد حل المثال السابق مع إضافة كل من باقي القسمة والمتوسط وكذلك طباعة جملة (الرمز خطأ) في حال الخطأ في إدخال الرمز الحسابي..

**تمرين 3.5**

ما وظيفة البرنامج التالي؟

```
#include <iostream>

using namespace std;

int main ()

{

int x;
```

```
cin>>x;
if((x%2)==0)
cout<<"even";
else
cout<<"odd";
return 0;
}
```

### 3.5 الشرط عن طريق المعامل (?)

يستخدم المعامل (?) كطريقة مختصرة للجملة IF ، ويأخذ الصيغة العامة التالية:

```
Variable=(condition)? Statement1:statement2;
```

### مثال 8.5

برنامج لحساب وطباعة ناتج المعادلة التالية باستخدام معامل الاختيار.

$$y=\begin{cases} 2x^4-3 & ,x>=7.8 \\ 3/x-1 & ,x<7.8 \end{cases}$$

```
#include <iostream>
#include <math.h>
using namespace std;
int main()
{     float x,y;
```

```
cin>>x;
y=(x>=7.8)?2*pow(x,4)-3:3/x-1;
cout<<y;
return 0;
}
```

**ملاحظة على المثال السابق**

-استخدام الدالة pow والتي تستخدم لرفع متغير ما إلى أس معين وتأخذ الصيغة:

```
Variable1 = pow(Variable2,Exceprition);
```

-استدعاء المكتبة math التي توفر الدالة pow.

-معامل الاختيار يختصر جملة if.

**مثال 9.5** برنامج لطباعة مجموع القيم الموجبة من 10 قيم مدخلة.

```
#include <iostream>
using namespace std;
int main()
{double m,sum;
int l;
sum=0;
l=1;
start: cout<<"Enter the values ";
cin>>m;
sum=(m>=0)?sum+m:sum+0;
l++;
```

```
if (l<=10)
goto start;
cout<<"summation ="<< sum;
return 0;
}
```

**ملاحظة**

البرنامج السابق احتوى على تكرار بعض الأوامر عدة مرات. تم تمثيل هذا عن طريق الأمر goto. في حين توفر لغة c++ - كما هو الحال في سائر لغات البرمجة – آلية برمجية أسهل تعرف بالحلقات.

# الباب السادس

# حلقات التكرار

# Loops

تستخدم حلقات التكرار لإعادة تنفيذ جزء من البرنامج عدة مرات حسب عدد محدد مسبقا أو وفقا لتحقق أو عدم تحقق شرط معين. توجد عدة صيغ للبناء البرمجي لحلقات التكرار في لغة C++ وهي:

## 1.6 حلقة التكرار For

إن المراحل الأساسية المكونة لأي تكرار هي:

-القيمة الابتدائية للحلقة.

-عدد مرات التكرار أو الشرط للتوقف.

-مقدار الزيادة للحلقة.

وهذه الأمور الثلاثة مضمنة في الصيغة العامة لهذه لحلقة التكرار for كما يلي:

```
for (initial value; condition; increment)
{
one or more statements;
}
```

وفيما يلي العديد من الأمثلة لتوضيح أوجه الاستفادة من الحلقات التكرارية:

**مثال 1.6** برنامج لطباعة الأعداد الصحيحة من 1 إلى 10.

```
#include <iostream>
using namespace std;
int main()
{int i;
for (i=1;i<=10;i++)cout<<i<<'\t';
return 0;}
```

المخرجات:

```
1    2    3    4    5    6    7    8    9    10
```

**مثال 2.6** برنامج لطباعة الأعداد الفردية عموديا من 15 إلى قيمة يتم إدخالها.

```
#include <iostream>
using namespace std;
int main()
{int i,n;
cout<<"Enter the ending value";
cin>>n;
if (n>15)
  for (i=15;i<=n;i+=2)cout<<i<<'\n';
else
cout<<"the value has to be >15";
return 0;
}
```

عند إدخال 30 مثلا يكون الخرج كالتالي:

```
15
17
19
21
23
25
27
29
```

**تمرين 1.6** ماذا لوكان المطلوب طباعتها على صفين؟

**مثال 3.6** برنامج لطباعة مجموع مضاعفات العدد 6 من 12 إلى 120.

```
#include <iostream>
using namespace std;
int main()
```

```
{int i,s=0;
  for (i=12;i<=120;i+=6)s+=i;
cout<<"the summation is  "<<s;
return 0;
}
```

**مثال 4.6** برنامج لطباعة متوسط درجات الحرارة خلال شهر ما عدا الأيام من 13 إلى 17.

```
#include <iostream>
using namespace std;
int main()
{int i,tmp,n,m,s=0;
float avr;
cout<<"Enter the number of month :"<<'\n';
cin>>n;
(n==2)?m=28:((n/2.0)==int(n/2.0))?m=30: m=31;
cout<<"Enter the tempretuer in sequence:"<<'\n';
for (i=1;i<=m;i++)
  {cin>>tmp;
   if (i<13||i>17)s+=tmp;}
   avr=s/(m-5);
cout<<"the average is  "<<avr;
return 0;}
```

**ملاحظة:** يستخدم الأمر *continue* للاستمرار في الحلقة والأمر *break* في الخروج منها.

**مثال 5.6** حل المثال السابق باستخدام *continue*.

```
#include <iostream>
using namespace std;
int main()
{int i,tmp,n,m,s=0;
float avr;
cout<<"Enter the number of month :"<<'\n';
cin>>n;
(n==2)?m=29:((n/2.0)==int(n/2.0))?m=30:m=31;
cout<<"Enter the tempretuer in sequence:"<<'\n';
for (i=1;i<=m;i++)
  {cin>>tmp;
   if (i>=13&&i<=17) continue;
   s+=tmp;}
   avr=s/25;
cout<<"the average is  "<<avr;
return 0;
}
```

**مثال 6.6** برنامج لحساب وطباعة ناتج العلاقة التالية:

$$y = \mathrm{n}! + \sum_{i=1}^{n} \frac{i^2}{3} \quad , \quad n > 1$$

```
#include <iostream>

using namespace std;

int main()

{int i,f,n;

float y,s;

s=0;

f=1;

cout<<"Enter n:"<<'\n';

cin>>n;

for (i=1;i<=n;i++)

  {   s+=i*i/3.0;

      f*=i;}

   y=s+f;

cout<<"the result is    "<<y;

return 0;

}
```

**مثال 7.6** برنامج لطباعة أكبر قيمة من بين n قيمة موجبة مدخلة.

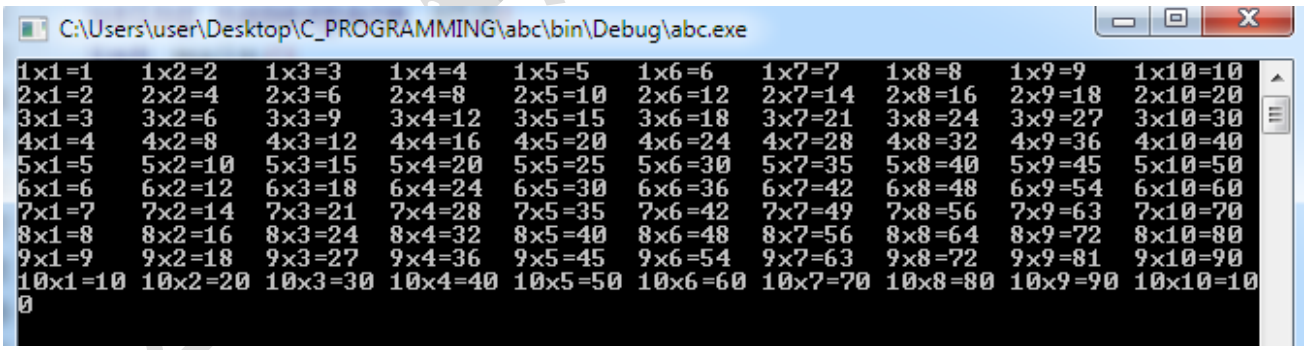
```
#include <iostream>

using namespace std;

int main()

{int i,max,n,m;

max=0;

cout<<"how many values:"<<'\n';

cin>>n;

for (i=1;i<=n;i++)

  {   cout<<"Enter the "<<i<<"th value  :";

      cin>>m;

if (m>max) max=m;}

cout<<"\nthe maximum falue is  "<<max<<'\n';

return 0;

}
```

**تمرين 2.6 :** ماذا لو كانت القيم المدخلة في المثال السابق موجبة وسالبة والمطلوب أكبر وأصغر قيمة؟

**أمثلة على الحلقات المتداخلة وطباعة الأشكال:**

**مثال 8.6** برنامج لطباعة جدول الضرب من 1 إلى 10.

```
#include <iostream>

//aprogram for printing out the multiplication table

using namespace std;

int main()

{int i,j;

for (i=1;i<=10;i++)

 { for (j=1;j<=10;j++)

    cout<<i<<"x"<<j<<"="<<i*j<<'\t';

  cout<<endl<<endl; }

return 0;

}
```




**مثال 9.6** برنامج لحساب وطباعة متوسط كل طالب من 5 طلبة في 4 مواد.

```
#include <iostream>

using namespace std;

int main()
```

```
{int i,j;
float mark,s;
for (i=1;i<=5;i++)
{s=0;
  for (j=1;j<=4;j++)
    {cout<<"mark "<<j<<" for student "<<i<<'\t';
     cin>>mark;
     s+=mark;}
   cout<<"average of student "<<i<<" is "<<s/4<<'\n'<<'\n';}
return 0;}
```

**مثال 10.6** برنامج لتمييز الأعداد الأولية وغير الأولية من 20 قيمة مدخلة.

```
#include <iostream>
using namespace std;
main()
{int x,i ,h,t;
int a;
h=t=0;
for(x=0;x<20;x++)
{
cout<<"Enter the nomber:";
cin>>a;
```

```
for(i=2;i<a;i++)
if(a%i==0) t=1;
if (t==1)
cout<<a<<"   is not prime\n";
else {
h=h+1;
cout<<a<<"   is prime\n";
}t=0;}
cout<<"number of prime is "<<h;
return 0;}
```

**مثال 11.6** برنامج لطباعة الشكل التالي:

```
#include <iostream>
using namespace std;
int main()
{int i,j;
for (i=1;i<=4;i++)
 { for (j=1;j<=10;j++)
    cout<<"* ";
    cout<<'\n'<<'\n';}
return 0;}
```

**مثال 12.6** برنامج لطباعة الشكل التالي:

```
#include <iostream>
using namespace std;
int main()
{int i,j;
for (i=1;i<=6;i++)
 { for (j=1;j<=10;j++)
    cout<<i<<'\t';
    cout<<'\n'<<'\n';}
return 0;}
```

```
1 1 1 1 1 1 1 1 1 1
 2 2 2 2 2 2 2 2 2 2
3 3 3 3 3 3 3 3 3 3
4 4 4 4 4 4 4 4 4 4
5 5 5 5 5 5 5 5 5 5
6 6 6 6 6 6 6 6 6 6
```

**مثال 13.6** برنامج لطباعة الشكل التالي:

```
#include <iostream>
using namespace std;
int main()
{int i,j;
for (i=1;i<=6;i++)
{ for (j=6;j>=i;j--)
    cout<<"*";
    cout<<'\n';}
return 0;
}
```

```
******
*****
****
***
**
*
```

**تمرين 3.6** برنامج لطباعة كل من الأشكال التالية:

a)
```
1 1 1 1 1
2 2 2 2
3 3 3
4 4
5
```

b)
```
1 2 3 4 5
1 2 3 4
1 2 3
1 2
1
```

c)
```
5 4 3 2 1
4 3 2 1
3 2 1
2 1
1
```

**مثال 14.6** برنامج لطباعة الشكل التالي:

```
    1
  2 1
3 2 1
4 3 2 1
```

```
#include <iostream>

using namespace std;

int main()

{int i,j;

for (i=1;i<=4;i++)

  {for (j=1;j<=4;j++)

     if (i+j<5)

      cout<<" ";

     else

      cout<<5-j;

      cout<<'\n';

  }

return 0;

}
```

**مثال 15.6** برنامج لطباعة الشكل التالي:

```
#include <iostream>

using namespace std;

int main()

{int i,j;

for (i=1;i<=7;i++)

  {for (j=1;j<=7;j++)

     if (i==1||i==7||j==1||j==7)

       cout<<"# ";

     else

     cout<<"& ";

     cout<<'\n';

  }

return 0;

}
```

```
# # # # # # #
# & & & & & #
# & & & & & #
# & & & & & #
# & & & & & #
# & & & & & #
# # # # # # #
```

**تمرين 4.6** برنامج لطباعة كل من الأشكال التالية:

```
8 8 9 8 8 8 9 8 8
8 8 9 8 8 8 9 8 8
8 8 9 8 8 8 9 8 8
8 8 9 8 8 8 9 8 8
8 8 9 8 8 8 9 8 8
8 8 9 8 8 8 9 8 8
```

```
A A A A A A
A A A A A A
A A A B A A
A A A A A A
```

**مثال 16.6** برنامج لطباعة الشكل التالي:

```
#include <iostream>

using namespace std;

int main()

{int i,j;


for (i=1;i<=8;i++)

  {for (j=1;j<=8;j++)

     if (i+j==9||i+j==13)

      cout<<"y ";

     else

      cout<<"x ";

    cout<<'\n';

  }

return 0;

}
```

```
x x x x x x x y
x x x x x x y x
x x x x x y x x
x x x x y x x x
x x x y x x x y
x x y x x x y x
x y x x x y x x
y x x x y x x x
```

**تمرين 4.6** برنامج لطباعة كل من:

a)

```
1 1 1 1 1 1 1
  2 2 2 2 2
    3 3 3
      4
```

b)

```
      5
    4   4
  3   3   3
2   2   2   2
1   1   1   1   1
```

c)

```
      4
    4 4 4
  4 4 4 4 4
4 4 4 4 4 4 4
  4 4 4 4 4
    4 4 4
      4
```

## 2.6 حلقة التكرار While & Do

هذا الأسلوب يستخدم عادة للتكرار فيما إذا كان عدد مرات التكرار غير معلوم حيث ينبني استمرار الحلقة على تحقق الشرط، فلا يمكن في هذه الحال استخدام حلقة For. كما يمكن استخدامه في حال معرفة عدد مرات التكرار.

الصيغة العامة لهذه الحلقة هي:

```
While (condition) statement;
```

كما يمكن تكرار عدة جمل باستخدام الأقواس { }

**مثال 17.6** برنامج لحساب وطباعة مجموع قائمة من الأرقام تنتهي بقيمة أكبر من 1200.

```
#include <iostream>

using namespace std;

main()

{float x=0,s=0;

 while(x<=1200)

 { cout<<"Enter the values\t";

   cin>>x;

   s+=x; }

 cout<<"\nsummation is\t"<<s<<'\n';}
```

كما يمكن استخدام الصيغة التالية لوضع شرط الاستمرار في آخر الحلقة:

```
Do
{statements;
}While (condition)
```

**مثال 19.6** برنامج لحساب وطباعة متوسط قائمة من القيم بحيث ينتهي الإدخال بقيمة سابة.

```
#include <iostream>
```

```
using namespace std;

main()

{int i;

float x,s=0,av;

 i=0;

 do

 {cout<<"Enter the values\t";

    cin>>x;

    s+=x;

    i++;

} while(x>=0);

 av=s/i;

 cout<<"\naverage is\t"<<av;}
```

# الباب السابع

# المصفوفات

## Arrays

تعرف المصفوفة على أنها مجموعة من البيانات (تسمى العناصر) من نفس النوع تخزن تحت اسم واحد. ويستفاد من هذه الآلية في إمكانية معالجة هذه البيانات برمجيا من خلال الحلقات كما سيأتي.

تنقسم المصفوفات إلى نوعين أساسيين هما مصفوفات أحادية البعد One dimensional array ومصفوفات متعددة الأبعاد Multi dimensional array ومن النوع الثاني سوف نتناول في هذا المنهج مصفوفات ثنائية البعدTwo dimensional array .

تحتوي المصفوفات الأحادية على صف واحد أو عمود واحد من العناصر كما في المثال التالي:

A=[1 4 5 6 5 -3 -45 0]

يشار إلى عناصر المصفوفة A السابقة كما يلي:

A[0]=1 , A[1]=4 , A[2]=5 , A[3]=6 , A[4]=5 , A[5]=-3 , A[6]=-45 , A[7]=0

وعند الإعلان عن نوع المصفوفة يجب تحديد عدد عناصرها مثلا:

```
int A[8];
```

بينما تحتوي المصفوفات الثنائية على عدة صفوف وأعمدة من العناصر كما في المثال التالي:

$$B=\begin{bmatrix} 1.2 & 3.0 & 0.6 \\ -5.2 & 4.4 & 0.6 \\ 7.9 & 3.5 & -0.9 \end{bmatrix}$$

يتم التعامل مع المصفوفات سواء من حيث الإدخال أو الإخراج أو المعالجة عن طريق الحلقات،بحيث يتم استخدام حلقة واحدة لأحادية البعد وحلقتين متداخلتين لثنائية البعد وهكذا. ويشار إلى عناصر هذا النوع من المصفوفات بترتيبه من خلال الصف والعمود. مثلا:

B[1][2]=0.6 ,  B[2][1]=3.5  B[2][2]=-0.9

وعند الإعلان عن نوع المصفوفة يجب تحديد عدد صفوفها وأعمدتها فمثلا:

Float B[3][3];  double mat[4][13];

**أمثلة مختلفة لبرامج تستخدم المصفوفات الأحادية والثنائية:**

**مثال 1.7** تخزين مرتبات مائة موظف في مصفوفة وحساب عدد المرتبات التي تزيد عن ألف دينار.

```
#include <iostream>
using namespace std;
 main()
{
 float salary[100];
 int i,k=0;
  for(i=0;i<100;i++)
    {
      cout<<"Enter the Salary of person:"<<i<<'\t';
      cin>>salary[i];
      if (salary[i]>1000)k++;
    }
  cout<<"result is:  "<<k;}
```

**مثال 2.7** استخدم المصفوفات لإدخال أسماء ودرجات وأرقام قيد 15 طالبا طباعة اسم ورقم قيد الطالب المتحصل على أعلى درجة.

```
#include <iostream>
using namespace std;
 main()
{
 string names[15],IDs[15];
 int i,k;
 float max=0,grades[15];
  for(i=0;i<15;i++)
    {
      cout<<"Enter the student name :"<<'\t';
      cin>>names[i];
      cout<<"Enter the student ID :"<<'\t';
      cin>>IDs[i];
      cout<<"Enter the student grade :"<<'\t';
      cin>>grades[i];
      if (grades[i]>max)
      {max=grades[i];      k=i;}
    }
  cout<<names[k]<<'\t'<<IDs[k];}
```

**مثال 3.7** حساب مجموع ومتوسط 5 قيم ثم طباعتها بعكس ترتيب الإدخال.

```
#include <iostream>

using namespace std;

main()

{int i ,sum,avg;

int a[5];

sum=0;

cout<<"enter the matrixs\n";

for(i=0;i<5;i++)

cin>>a[i];

for(i=0;i<5;i++)

sum+=a[i];

avg=sum/5;

cout<<"\nsum="<< sum <<"\navg="<<avg<<endl;

cout<<"\nthe matrixs invers is\n";

for(i=4;i>=0;i--)

cout<<a[i]<<ends;}
```

**مثال 4.7** دمج مصفوفتين مدخلتين في مصفوفة واحدة.

```
#include <iostream>
using namespace std;
main()
{int i,j,n;
n=0;
int a[5];
int b[5];
int c[10];
cout<<"enter first matrixs\n";
for(i=0;i<5;i++)
cin>>a[i];
for(i=0;i<5;i++)c[i]=a[i];
cout<<"enter second matrixs\n";
for(j=5;j<10;j++)
cin>>b[j];
for(j=5;j<10;j++)
c[j]=b[j];
cout<<endl<<"the matrex we got:";
for(n=0;n<10;n++)
cout<<c[n]<<ends;}
```

**مثال 5.7** ترتيب عناصر مصفوفة (n عنصرا) تصاعديا.

```
#include <iostream>
using namespace std;
main()
{int i,j,n,t;
cout<<" how many element\n";
cin>>n;
int arr[n];
cout<<"enter the matrixs\n";
for(i=0;i<n;i++)
 cin>>arr[i];
for(i=0;i<n-1;i++)
  for(j=i+1;j<n;j++)
   if(arr[j]<arr[i])
   {t=arr[i];
    arr[i]=arr[j];
    arr[j]=t;}
cout<<endl<<"the arranged matrex is:";
for(i=0;i<n;i++)
 cout<<arr[i];
}
```

**مثال 6.7** إيجاد الحرف الأكثر تكرارا في مصفوفة حرفية من 20 عنصرا.

```
#include <iostream>
using namespace std;
main()
{int i,j,n=0,max=0;
char a[20];
int b[20];
cout<<"enter the matrixs\n";
for(i=0;i<20;i++)
 cin>>a[i];
for(i=0;i<20;i++)
  {n=0;
   for(j=0;j<20;j++)
    if(a[i]==a[j])n++;
    b[i]=n;}
for(i=0;i<20;i++)
  if(b[i]>max)
   {max=b[i];
    j=i;}
cout<<endl<<"\nthe most repeated element:"<<a[j];
cout<<endl<<"\nit was repeated:  "<<b[j]<<"  times\n";}
```

**مثال 6.7** مصفوفة إدخال مصفوفة (nxm) وطباعة محورتها.

```
#include <iostream>
using namespace std;
main()
{int i,j,n,m;
cout<<"Enter rows then coloms";
cin>>n;
cin>>m;
float mat1[n][m],mat2[m][n];
cout<<"enter the matrixs\n";
for(i=0;i<n;i++)
  for(j=0;j<m;j++)
    cin>>mat1[i][j];
for(i=0;i<m;i++)
  {for(j=0;j<n;j++)
    {mat2[i][j]=mat1[j][i];
  cout<<mat1[j][i]<<'\t';}
      cout<<endl<<endl;}
}
```

**مثال 7.7** مصفوفة لإدخال درجات 6 طلبة في 6 مواد وحساب وطباعة الآتي:

-متوسط كل طالب.

-متوسط كل مادة.

-المتوسط الكلي للمواد.

-المتوسط القطري (القطر الرئيسي).

-المتوسط القطري (القطر الفرعي).

-طباعة المصفوفة مع كل هذه المخرجات بشكل مناسب.

```
#include <iostream>
using namespace std;
main()
{int i,j;
float s1,s2,s3,s4,st;
float degree[7][7];
st=0;
cout<<"enter the matrixs\n";
for(i=0;i<6;i++)
  for(j=0;j<6;j++)
    cin>>degree[i][j];
s3=s4=0;
for(i=0;i<6;i++)
  {s1=s2= 0;
  for(j=0;j<6;j++)
    {s1+=degree[i][j];
     s2+=degree[j][i];
     s3+=(i==j)?degree[i][j]:0;
     s4+=(i+j==5)?degree[i][j]:0;}
   degree[i][6]=s1/6;
   degree[6][i]=s2/6;
   st+=s1;}
```

```
for(i=0;i<=6;i++)
  {for(j=0;j<=6;j++)
    cout<<degree[i][j]<<'\t';
  cout<<endl<<endl;}
cout<<endl<<"the main diagonal average is :"<<s3/6;
cout<<endl<<"the sub diagonal average is :"<<s4/6;
cout<<endl<<"the total average is :"<<st/36;
}
```

**مثال 8.7** لديك مصفوفتان A(2x4) و B(2x4) احسب واطبع المصفوفة C حيث:

C=3A- 0.5B

```
#include <iostream>
using namespace std;
main()
{int i,j;
float A[2][4],B[2][4],C[2][4];
cout<<"enter the matrixs A\n";
for(i=0;i<2;i++)
  for(j=0;j<4;j++)
    cin>>A[i][j];
cout<<"enter the matrixs B\n";
for(i=0;i<2;i++)
  for(j=0;j<4;j++)
    cin>>B[i][j];
```

```
for(i=0;i<2;i++)
  {for(j=0;j<4;j++)
    {C[i][j]=3*A[i][j]-0.5*B[i][j];
    cout<<C[i][j]<<'\t';}
    cout<<endl<<endl;}
}
```

**مثال 9.7** حساب حاصل ضرب مصفوفتين b(2x4) و a(3x2).

```
#include <iostream>
using namespace std;
main()
 {
 int i,j,k;
 int a[3][2];
 int b[2][4];
 int c[3][4]={0};  //put zero in every location to sum with other value
 cout<<"enter first matrixs\n" ;
 for(i=0;i<3;i++)
 for(j=0;j<2;j++)
 cin>>a[i][j];
cout<<"enter second matrixs\n" ;
 for(i=0;i<2;i++)
```

```
 for(j=0;j<4;j++)
   cin>>b[i][j];
for(i=0;i<3;i++){
   cout<<"\n";
 for(j=0;j<4;j++){
   for(k=0;k<2;k++)
     c[i][j]+=a[i][k]*b[k][j];
  cout<<c[i][j]<<"\t" ;
 } } }
```

تمرين 1.7 اكتب برنامجا لحساب وطباعة مجموع العناصر التي فوق القطر الرئيسي ومجموع العناصر التي فوق القطر الثانوي لمصفوفة مدخلة (7x7).

تمرين 2.7 اكتب برنامجا لإيجاد وطباعة مصفوفة أحادية مكونة من أكبر عنصر في كل صف من مصفوفة ثنائية مدخلة.

# الباب الثامن

# الدوال

## functions

توفر لغة C++ - كما هو الحال في سائر لغات البرمجة - إمكانية التفرع من البرنامج الرئيسي المكتوب داخل الدالة main إلى مقاطع برامج أخرى (برامج فرعية subprograms)؛ الأمر الذي يقلل الخطوات المكررة في البرنامج وينظم عملية البرمجة أكثر. يتم ذلك في لغة C من خلال الدوال.

ويمكن فهم الدوال من خلال مجموعة الأمثلة التالية:

**مثال 1.8** برنامج يستخدم دالة لحساب مضروب أي مدخل.

```
#include <iostream>
using namespace std;
int f=1,i,n;
int fact()
{f=1;
 for(i=1;i<=n;i++)
   f*=i;
return f;
}
main()
{
```

```
 cout<<"Enter the number";
 cin>>n;
 fact();
cout<<endl<<"factor of :"<<n<<"  is"<<f;}
```

**مثال 2.8** برنامج يستخدم دالة لطباعة الأعداد الموجبة فقط من قائمة تنتهي بصفر.

```
#include <iostream>
using namespace std;
int v;
int test()
{(v>=0)?cout<<v<<'\n':cout<<"  \n";}
main()
{
 cout<<"Enter the value\n";
 cin>>v;
 do{
 test();
 cin>>v;
 }while(v!=0);}
```

**ملاحظة:** الدالة لا يشترط أن ترجع قيمة ما للبرنامج الرئيسي كما في المثال السابق.

**مثال 3.8** استخدام الدوال في اختبار العدد الزوجي والفردي باختبار البت من قائمة مكونة من 14 قيمة.

```
#include<conio.h>
#include <iostream>
using namespace std;
string even_odd();
int i,value;
string m;
 main()
{for(i=1;i<=14;i++)
{cout<<"Enter the value";
cin>>value;
cout<< "the value  "<<value<<"  is   "<<even_odd()<<'\n';}}
string even_odd()
{m=((value&1)==0)?"even":"odd";
return m;}
```

**ملاحظة**:1. توجد صيغة يستخدم فيها الاختصار (Macro) عن طريق الأمر (define) وذلك لاختصار الدوال الفرعية كما في الأمثلة التالية:

```
#define smaller(a,b) (a<b?a:b);
#define sign(a) (a<0)?'-':'+';
```

2.آمل من الطلبة الرجوع لبعض المراجع للتعرف على أكبر قدر من الدوال الجاهزة التي توفرها ++c.

## دوال التحكم في الأحرف والسلاسل الحرفية **Character manipulation functions**

1 - دوال اختبار الأحرف Character testing functions *(ctype.h)*

| | |
|---|---|
| isdigit(x) | هل x رقم من 0 إلى 9؟ |
| isalpha(x) | هل x حرف من A إلى z؟ |
| isalnum(x) | هل x حرف أو رقم ؟ |
| isxdigit(x) | هل x رقم ست عشري ؟ |
| islower(x) | هل x حرف صغير؟ |
| isupper(x) | هل x حرف كبير؟ |
| isspace(x) | هل x مسافة (فراغ)؟ |
| ispunct(x) | هل x رمز تشكيل؟ |

2 - دوال التحويل من صغير إلى كبير وبالعكس *Character conversion functions*

| | |
|---|---|
| tolower(x) | لتحويل x من حرف كبير إلى حرف صغير |
| toupper(x) | لتحويل x من حرف صغير إلى حرف كبير |

3 - دوال معالجة السلاسل *String manipulation functions (string.h)*

| | |
|---|---|
| strlen(x) | لتحديد طول السلسلة الحرفية |
| strcat(x1,x2) | لربط سلسلتين حرفيتين |
| strncat(x1,x2,n) | لربط السلسلة الحرفية الأولى مع أول n حرف من الثانية |
| strcpy(x1,x2) | لنسخ السلسلة الحرفية الثانية في الأولى |
| strncpy(x1,x2,n) | لنسخ أول n حرف من السلسلة الحرفية الثانية في الأولى |
| strcmp(x1,x2) | إرجاع قيمة موجبة إذا كان x1>x2 وقيمة سالبة إذا كان x2>x1 وصفر إذا كان X1=x2 |
| strncmp(x1,x2.n) | مقارنة nحرف من السلسلتين |

4 - دوال تحويل السلسلة *String conversion functions*

| | |
|---|---|
| strtod(x) | للتحويل من string إلى double |
| strtol(x) | للتحويل من string إلى long |
| strtoul(x) | للتحويل من string إلى unsigned long |

**مثال 4.8 :**

توقع مخرجات البرنامج التالي ثم قم بتنفيذه للتحقق من ذلك.

```
#include <iostream>
using namespace std;
#include<ctype.h>
// a program to test the chracter:
main(void)
{
char x;
cout<<"Enter a charecter please :";
cin>>x;
if(isdigit(x))cout<<endl<<"the charecter you gave is :1";
if( isalpha(x))cout<<endl<<"the charecter you gave is :2";
if(isalnum(x))cout<<endl<<"the charecter you gave is :3";
if(isxdigit(x))cout<<endl<<"the charecter you gave is :4";
if(islower(x))cout<<endl<<"the charecter you gave is :5";
if(isupper(x))cout<<endl<<"the charecter you gave is :6";
```

```
if(isspace(x))cout<<endl<<"the charecter you gave is :7";
if(ispunct(x))cout<<endl<<"the charecter you gave is :8";
return 0;}
```

**مثال 5.8:** برنامج لحساب عدد الأحرف الكبيرة من بين 10 أحرف مدخلة.

```
#include <iostream>
using namespace std;
#include<ctype.h>
// aprogram to test the chracter:
main(void)
{
char x;
int c=0,i;
for(i=1;i<=10;i++)
{
cout<<"Enter a charecter please :";
cin>>x;
if(isupper(x))c+=1;}
cout<<endl<<"ther is   "<<c<<"   capital characters";
return 0;
}
```

**مثال 6.8 :** برنامج لإدخال أي رقم وتحديد عدد الخانات التي يتكون منها بما في ذلك الفاصلة.

```
#include <iostream>

using namespace std;

#include<ctype.h>

#include<string.h>

int main()

{

char c[20];

cout<<"Enter any nomber:"<<'\t';

cin>>c;

cout<<"digits in yor nomber is\t"<<strlen(c);

return 0;

}
```

**تمرين 1.7** : ماذا لوكان المطلوب في المثال السابق عدد الجزء الصحيح لوحده والجزء الكسري لوحده؟

**تمرين 2.7 :** قم بتنفيذ برامج أخرى لاختبار باقي الدوال.

# الباب التاسع

# المؤشرات

# Pointers

المؤشر هو متغير يشير إلى عنوان موقع في الذاكرة. ويستخدم لهذا الغرض البارامترات التالية:

* :تلحق المتغير الذي يحتوي على العنوان وتستخدم للإشارة إلى القيمة التي بهذا العنوان.

& : تشير إلى العنوان في الذاكرة لقيمة المتغير الذي تلحقه.

تجدر الإشارة هنا إلى أن البيانات مخزنة في الذاكرة على هيئة كم هائل من الصفوف المكونة من 8 خانات ثنائية (8bits=1 byte)، ويخصص لكل موقع عنوان ثنائي (binary address) يعبر عنه بصيغة العشري المشفر ثنائيا (binary coded decimal BCD)

| | |
|---|---|
| Address 0 | Value 0 |
| Address 1 | Value 1 |
| Address 2 | Value 2 |
| . | . |
| . | . |
| Address n | Value n |

وعند الإعلان عن أنواع المتغيرات في أول البرنامج فإن معالج الحاسب (microprocessor)

سوف يقوم بحجز عدد من المواقع لهذا المتغير موافق لذاك النوع.

| | |
|---|---|
| Char | 1 byte |
| Int | 2 byte |
| Long | 4 byte |
| Float | 4 byte |
| double | 8 byte |

**مثال1.9:** لاحظ مخرجات البرنامج التالي:

```
#include<conio.h>
#include <iostream>
using namespace std;
 main()
{int i=0,v,*k;
while(i<=5)
{cout<<"Enter any value";
cin>>v;
k=&v;
cout<< "the value yuo entered is: "<<*k<<'\n';
cout<< "its address in the memory is: "<<k<<'\n';
i++;}}
```

**مثال2.9:** لاحظ مخرجات البرنامج التالي:

```
#include<conio.h>
#include <iostream>
using namespace std;
int a,b,c;
void change(int *a,int *b)
{int c=*a;
*a=*b;
*b=c;}
main(){
cout<<"Enter the twovalues\n";
cin>>a>>b;
cout<<"a="<<a<<" ,   b="<<b<<endl;
change(&a,&b);
cout<<"a="<<a<<" ,   b="<<b;}
```