

دولة ليبيا

وزارة التعليم العالي والبحث العلمي

جامعة الزاوية

كلية التربية - العجيلات

مقدمة في

البرمجة بلغة C++

السنة الأولى - تخصص الحاسوب

إعداد

أ. عدنان عون الله شكرو

دبلوم عالي برمجة - المعهد العالي للمهن الشاملة غدامس 2001م

ماجستير نظم معلومات - أكاديمية الدراسات العليا طرابلس - 2013م

للعام الجامعي

2014-2013

المحاضرة الأولى

مدخل إلى البرمجة

1 . 1 مفاهيم أساسية :

تعريف البرمجة Definition Programming

هي عملية كتابة تعليمات وتوجيه أوامر لجهاز الحاسوب ، لتوجيه هذا الجهاز وإعلامه بكيفية التعامل مع البيانات أو كيفية تنفيذ سلسلة من الأعمال المطلوبة تسمى خوارزمية.

تعريف البرنامج Definition program

هو مجموعة من التعليمات والأوامر التي توضح للحاسب تسلسل الخطوات التي ينبغي القيام بها لأداء مهام معينة لحل المشكلة المطروحة واستخراج النتائج. ويخزن البرنامج في الذاكرة الرئيسية للحاسوب.

تعريف البرمجيات Definition Software

وهي الجانب المعنوي من مكونات الحاسوب وتعد وسيلة للتفاهم والتخاطب والاتصال بين الحاسوب والانسان .

1 . 2 أنواع البرمجيات Types of Software

تتعدد تقسيمات البرمجيات فمنهم من يقسمها إلى برامج نظم وبرامج تطبيقية ومنهم من يقسمها إلى التالي :-

1. نظم التشغيل Operating Systems : وهي البرامج المسؤولة عن تشغيل وإدارة

الحاسوب والتفاهم بين مكوناته المادية مثل الطابعات ومكوناته البرمجية مثل برنامج معالجة النصوص ، ومن أمثلة نظم التشغيل Dos , Windows , Linux .

2. البرامج التطبيقية Application Programs : هي مجموعة من البرامج التي

صممت لأداء وظيفة معينة مثل حزمة برامج أوفيس Microsoft Office وأهم برامجها برنامج معالجة النصوص (الطباعة) Microsoft word ، وبرنامج الجداول الالكترونية Microsoft Excel ، إضافة إلى المنظومات الجاهزة التي يتم تصميمها بلغات البرمجة مثل منظومات حساب الرواتب والمنظومات المصرفية.

3. لغات البرمجة Programming Languages : عبارة عن مجموعة من الأوامر،

تكتب وفق مجموعة من القواعد تحدد بواسطة لغة البرمجة، ومن ثم تمر هذه الأوامر

بعدة مراحل إلى ان تنفذ على جهاز الحاسوب مثل لغة C++ وتنقسم إلى عدة أقسام من أهمها :

(أ) لغات المستوى الواطي (Low Level Language (L.L.L) : وهي اللغة التي تستخدم (0 ، 1) للتعبير عن الأوامر المختلفة التي يتكون منها البرنامج ، وهي لغة صعبة لا يحسنها إلا من صمم الحاسبة نفسها، وتسمى لغة الآلة (Machine Language) .

(ب) لغات المستوى المتوسط (Meddle Level Language (M.L.L) : هي لغة تتميز بأنها وسط بين لغة الماكنة واللغة العالية وتستخدم خليط من العلامات والرموز وتسمى لغة التجميع (Assembly Language) .

(ج) لغات المستوى العالي (High Level Language (H.L.L) : وهي اللغات الحديثة المستخدمة في أجهزة الحاسوب وهي قريبة من لغة الانسان وتتميز بسهولة الكتابة وسهولة اكتشاف الأخطاء ومن أمثلتها لغة البيسك (Basic) ولغة الفورتران (Fortran) ولغة الباسكال (Pascal) ولغة السي (C) والسي بلس بلس (C++) ، وغيرها إلى جانب اللغات المرئية الحديثة التي تتلاءم مع الوسائط المتعددة وتسمى اللغات المصورة مثل الفيچوال بيسيك Visual Basic والدلفي Delphi .

1. 3 خطوات حل المسائل Steps Of Problem Solving

هناك عدد من الخطوات التي تسهل عملية حل المسألة في لغة البرمجة من أهمها :

1. **تعريف وتحليل المسألة Analysis and Definition Problem** : ويقوم فيها

المبرمج بمعرفة ما إذا كانت المسألة قابلة للحل مع تحديد عناصر المدخلات والعمليات والمخرجات.

2. **الخوارزميات Algorithm** : الخوارزمية هي مجموعة من الخطوات الرياضية والمنطقية

والمتسلسلة اللازمة لحل مشكلة ما، وقد تكون هذه الخطوات باللغة العادية وعند تتبعنا لهذه الخطوات نصل للحل النهائي، وسميت الخوارزمية بهذا الاسم نسبة إلى العالم المسلم أبو جعفر محمد بن موسى الخوارزمي الذي ابتكرها في القرن التاسع الميلادي.

3. مخطط سير العمليات Flowchart : يسمى أيضاً بالمخطط الانسيابي ، وهو عبارة عن طريقة تخطيطية تمثل تتابع الأحداث بأشكال رمزية وخطوط تمثل مسار عمليات البرنامج المنطقية ، كل شكل من هذه الأشكال يعبر عن نوع الأمر أو التعليمة في حل المسألة، وهي ترجمة تخطيطية للخوارزمية .

4. البرمجة Programming : وهي القيام بترجمة الخطوات السابقة إلى مجموعة من الأوامر المكتوبة بلغة يستطيع الحاسب فهمها وهي لغات البرمجة وإدخالها إلى الحاسوب عن طريق أحد وسائل الإدخال مثل لوحة المفاتيح وشاشة العرض.

5. اختبار البرنامج Program Testing : ويعني أن نقوم بتنفيذ البرنامج وإدخال عينة من البيانات ، فإذا كان الناتج صحيحاً يعني أن البرنامج صحيح ، وإذا كان غير صحيح فينبغي مراجعة البرنامج وتصحيحه .

1 . 4 الخوارزميات Algorithms :

قبل كتابة خوارزمية أي مسألة يجب تحديد عناصر المدخلات وعناصر المخرجات وبينهما العملية المطلوبة ولا ننسى أن كل خوارزمية تبدأ بالبداية Start وتنتهي بالنهاية End وترتب الخوارزمية على هيئة خطوات متسلسلة تساهم في ترجمة المسألة إلى مخطط انسيابي أو برنامج يكتب بأحد لغات البرمجة .

مثال 1 : أكتب خوارزمية لجمع عددين وطباعة الناتج النهائي ؟

قبل أن نبدأ في الحل نعرف المتغيرات لنفرض أن العددين هما A , B وهما يمثلان المدخلات ولنفرض أن المجموع هو SUM وهو يمثل المخرجات ، والعملية هي $SUM = A + B$ والخوارزمية تكون على النحو التالي :

1. البداية

2. أدخل القيم A , B

3. أجمع $SUM = A + B$

4. أطلع الناتج SUM

5. النهاية





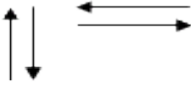

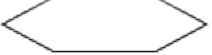
مثال 2 : أكتب خوارزمية لإيجاد مساحة الدائرة ؟

بداية لابد أن نعرف أن مساحة الدائرة = πR^2 وبما أن البرمجة بالغة الانجليزية فلنفرض أن المعادلة تكون بالشكل $A = (\text{PIE}) * R^2$ ولابد أن نحدد المدخلات والمخرجات في المعادلة ، فالمدخلات هي PIE وهي ثابت يساوي 3.14 ، و R وهي عبارة عن متغير ، أما المخرجات التي سيتم طباعتها وكتابتها هي A والخوارزمية تكون على النحو التالي :

1. البداية .
2. اقرأ قيمة R .
3. ضع قيمة $\text{PIE}=3.14$
4. احسب المساحة (A) من المعادلة $A = (\text{PIE}) * R * R$
5. اطبع المساحة A
6. النهاية.

1 . 5 المخططات الانسيابية Flowchart :

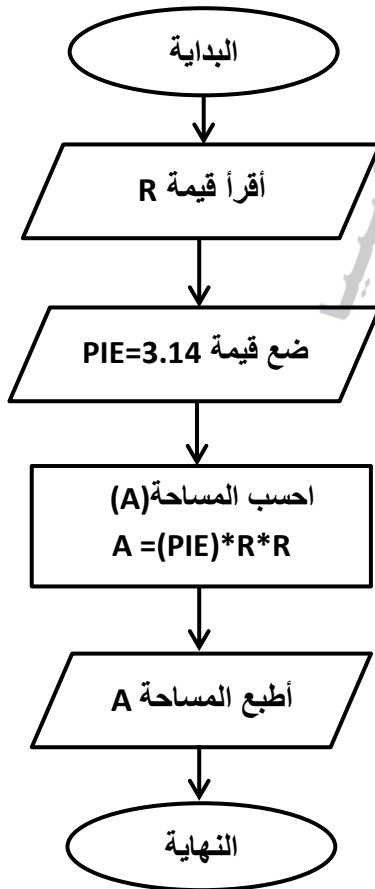
تمثل خريطة سير العمليات وصفاً تصويرياً لخطوات الخوارزمية تكون أكثر وضوحاً. وخريطة سير العمليات تقوم مقام الخوارزمية ويمكن بواسطتها ملاحظة تتبع التسلسل المنطقي لحل المسألة بكل سهولة، وغالباً ما تكون استخراج الخوارزمية من خريطة سير العمليات أسهل بكثير من كتابة الخوارزمية مباشرة. وعند رسم خريطة سير العمليات لمسألة معينة فإننا نستخدم مجموعة من الأشكال الرمزية الاصطلاحية المبينة في الجدول التالي:

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

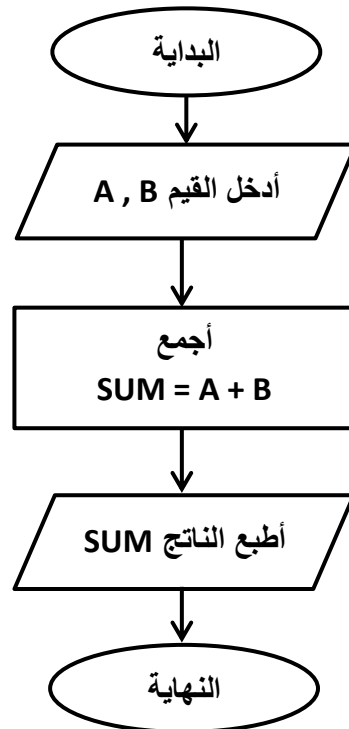
من أهم فوائد استخدام خرائط سير العمليات قبل كتابة البرنامج لمسألة ما، ما يأتي:

1. تمكن المبرمج من الإلمام الكامل بالمسألة المراد حلها و السيطرة على كل أجزائها بحيث تساعده على اكتشاف الأخطاء المنطقية (Logic Error) و التي تعتبر من أهم الأخطاء التي تجهد المبرمج.
2. تساعد بيسر و سهولة على تعديل البرامج الموضوعه بمجرد النظر.
3. يعتبر الاحتفاظ برسوم خرائط سير العمليات لحلول مسائل معينة أمراً مهماً إذ يكون مرجعاً عند إجراء تعديلات عليها أو استخدامها لحل مسائل أخرى مشابهة دون الحاجة إلى الرجوع إلى المبرمج الأول باعتبار أن الحلول الأولى قد صيغت في خطوات واضحة بسيطة و مفهومة.
4. توفير وسيلة مناسبة ومساعدة في كتابة البرامج ذات التفرعات الكثيرة .

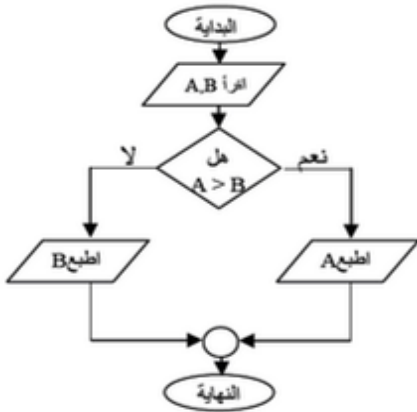
مثال 4 : ارسم المخطط لإيجاد مساحة الدائرة وطباعة الناتج النهائي (مثال 2)



مثال 3 : ارسم المخطط الانسيابي لعملية جمع عددين وطباعة الناتج النهائي (مثال 1):



مثال 5 : أكتب الخوارزمية وأرسم المخطط الانسيابي لطباعة العدد الأكبر بين عددين ؟
نفرض أن العددين هما A,B وبالتالي تكون الخوارزمية على النحو التالي :



1. البداية
2. أدخل قيمة A , B
3. إذا كان $A > B$ اذهب إلى الخطوة 5.
4. أطلع A
5. أطلع B
6. النهاية

1 . 6 أسبقية تنفيذ العمليات

هي قاعدة تستخدم لتوضيح أي العمليات الحسابية يجب تنفيذها أولاً في جملة حسابية معينة، ويرتب تنفيذ العمليات على النحو التالي :-

1. تنفيذ ما داخل الأقواس.
2. تنفيذ عمليات الأسس والجذور.
3. الضرب والقسمة.
4. الجمع والطرح.

مثال 6 : أحسب ناتج هذه المعادلة بناءً على القواعد السابقة $4*2+3*(2-6/4+1)$

تنفيذ العمليات داخل الأقواس: $(2-6/4+1)$

أولويات الضرب والقسمة حسب موقعها في التعبير: $6/4=1.5$

أولويات الجمع والطرح حسب موقعها في التعبير $2-1.5=0.5$ و $0.5+1=1.5$

أولويات الضرب والقسمة $3*1.5=4.5$ و $4*2=8$

أولويات الجمع والطرح $4.5+8=12.5$:

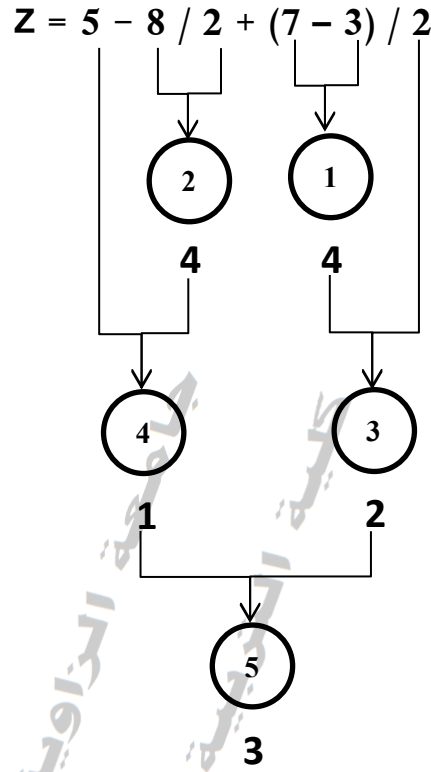
ويكون التعبير النهائي بالتالي:

$$4*2+3*(2-6/4+1)=12.5$$

مثال 7 : أحسب ناتج هذه المعادلة بناءً على قواعد أسبقية تنفيذ العمليات

$$Z = 5 - 8 / 2 + (7 - 3) / 2$$

الحل



إذا الناتج النهائي للمعادلة = 3 : $Z = 5 - 8 / 2 + (7 - 3) / 2 = 3$

مراجع المحاضرة الأولى :

1. بشير القايد ، اساسيات البرمجة ، منشورات ELGA ، 2001 .
2. عدنان عون الله ، مقدمة عامة في الحاسب الآلي ، 2007 ، متاح في هذا الموقع www.boosla.com .
3. <http://mokh.forumegypt.net/t24-topic> .
4. <http://komanda.yoo7.com/t949-topic> .
5. <http://ar.wikipedia.org> .
6. http://computer.atlas4e.com/Project_E1/Project/chapter12/chapter12.htm .
7. http://t3lmtechnologya.blogspot.com/2012/09/blog-post_7437.html .

المحاضرة الثانية

أساسيات لغة ++C

2 . 1 مقدمة

هي لغة برمجة متعددة الاستخدام، وتعتبر لغة برمجة كائنية. يعتبرها الكثيرون اللغة الأفضل لتصميم التطبيقات ذات الواجهة الكبيرة وللتعامل مع البنية الصلبة للحاسب، وذلك لسرعتها في التنفيذ والتي لا تختلف كثيراً عن السي بل هي تطوير للغة السي ، وفي المقابل توفر تعامل أكثر تعقيداً مع البيانات. وتعتبر من لغات البرمجة العالية المستوى وفي نفس الوقت قريبة من لغة التجميع (بالإنجليزية Assembly) ذات المستوى المتوسط .

وقد اخترع هذه اللغة المبرمج بشركة التليفونات الأمريكية بيران ستراوستراب في الثمانينات من القرن المنصرم ، حيث تم إضافة بعض النواقص والعيوب في اللغة الأصلية سي (C) وتسميتها بنفس الحرف السابق مع إضافة ++ الذي يعني في هذه اللغة "خطوة إلى الأمام" وأصبحت تعرف باسم سي ++ (C++)، وبالتالي جرى استدراك بعض النواقص والعيوب في اللغة سي حتى تسهل مهمة المبرمج .

2 . 2 أساسيات لغة سي ++

لأي لغة مجموعة من الأساسيات التي تنطلق منها ومن أهم أساسيات لغة سي ++:

1. الأرقام : وهي من 0 - 9.

2. الحروف الهجائية اللاتينية الكبيرة والصغيرة a-z ، A-Z .

3. الرموز الخاصة مثل + ، - ، / ، * ، ؛ ... وغيرها .

2 . 3 الرموز Characters

هو عبارة عن حرف أو رقم أو رمز خاص بشرط أن يكون موضوع بين علامتي تنصيص مفردة.

مثال 1: 'n' 'A' '5' '&'

إضافة إلى الحرف توجد السلسلة string ، والسلسلة تتكون من مجموعة من الرموز التي توضع بين علامتي تنصيص مزدوجة "" .

مثال 2: "/My job is teacher/" "#10 go to step" .

2 . 4 الأعداد Numbers

وتسمى أيضا الثوابت الرقمية ، وهي اعداد ثابتة لا تتغير قيمتها وتنقسم إلى :

2 . 4 . 1 الاعداد الصحيحة :

العدد الصحيح هو عدد سالب أو موجب أو صفر بشرك الا يحتوي على الفاصلة العشرية .

مثال3: هذه الاعداد صحيحة مقبولة 123 0 -598

مثال4: بين لماذا هذه الأرقام تعتبر غير مقبولة كأعداد صحيحة : 12.89 ، 45,000 ، 100£

الحل : 12.89 لوجود الفاصلة العشرية

45,000 لاحتوائها على رمز خاص هو ،

100£ لا يعتبر رقما لوجود علامة اليورو

ومن أنواع الأعداد الصحيحة ما يلي :

1. العدد الصحيح (Integer) : حيث يخصص 16 او 32بت .

2. العدد الصحيح القصير (Short Integer) : يخصص له 16 بت ومداه من -32768 إلى 32767 .

3. العدد الصحيح الطويل (Long Integer) ويخصص له عادة 32 بت ومداه من -2147483648 إلى 2147483647 .

وفائدة القاعدة والتقسيم أعلاه أنه أثناء البرمجة حين يكون الرقم قصير يحجز المترجم مساحة أقصر ، وإن كان طويل كذلك .

2 . 4 . 1 الاعداد الحقيقية :

وهي الاعداد التي تحتوي على العلامة العشرية ، ويمكن تمثيلها بأسلوبين:-

1. النقطة الثابتة (Fixed point) ويحتوي على :

- الجزء الصحيح.

- الفاصلة العشرية (.)

- الجزء الكسري ما بعد النقطة

مثال5: هذه الاعداد حقيقية مقبولة تحتوي على نقطة ثابتة 5.48 -85.23 66.00 0.079

مثال6: بين لماذا هذه الأرقام تعتبر غير مقبولة كأعداد حقيقية \$100.85\$ 99.75.2 546 ؟

100.85\$ لوجود علامة الدولار

99.75.2 لوجود أكثر من نقطة عشرية

546 لعدم وجود نقطة عشرية

2. النقطة السائبة أو العائمة (Floating point) ويحتوي على :

- الجزء الصحيح

- الحرف E

- الخانة الاسية

مثال6: مثل العدد 14000000000.0 تمثيل أسي :

14E9 0.14E11 1.4E10

مثال7: مثل الاعداد التالية الحقيقية بالقوة الأسية 34500.0 0.000345 -3456.7

34500.0 يساوي 3.45E4

0.000345 يساوي 3.45E-4

-3456.7 يساوي -3.4567E3

2 . 4 الكلمات المحجوزة :

وهي كلمات تستعمل في لغة سي ++ كأوامر وأسماء لا يمكن استعمالها كمتغيرات لأنها

قد تسبب ريكة للمترجم (compiler) .

مثال8: char case delete int if for long

2 . 5 المعارف

هو ذلك الاسم أو المعارف الذي تخزن فيه قيمة المتغيرات مثل الثابت او المتغير ومن

شروطه:-

1. أن يتكون من حرف أو مجموعة حروف أو حرفاً وأرقام وعلامة () .

2. يجب أن يبدأ بحرف من اليسار .

3. يجب أن يكون خالياً من الرموز الخاصة فيما عدا () .

4. يسمح باستخدام الحروف الصغيرة والحروف الكبيرة .

مثال 9: المعارف التالية معارف صحيحة :

Message Total_price stud_name Area5

مثال 10: المعارف التالية معارف غير صحيحة للأسباب المذكورة بمحاذااتها:

Vitamin C	لوجود الفراغ
Counlev#	لوجود علامة خاصة
9time	بدأت برقم
double	كلمة محجوزة

2 . 6 المتغيرات Variables

هي أسماء رمزية يخصص لها أماكن تخزين في ذاكرة الحاسب ، والتي تتحول قيمتها وتتغير من قيمة لأخرى ، حيث يمكن الرجوع لهذه القيم عن طريق هذه الأسماء وذلك أثناء تنفيذ البرنامج. ولاستخدام متغير داخل برنامج سي ++ لابد من الاعلان عنه وتعريف في بداية البرنامج على النحو التالي :

Type var1, var2,

حيث Type تعني نوع المتغير المراد الاعلان عنه هل هو متغير صحيح أو حقيقي أو حرفي. ومن أشهر أنواع المتغيرات أو التعبيرات التي نستعملها للإعلان عن المتغيرات ما يلي:

int	لتعريف متغير صحيح
long	لتعريف متغير صحيح طويل
float	لتعريف متغير حقيقي
char	لتعريف متغير حرفي
double	لتعريف متغير مضاعف الدقة

على أن تكتب التعريفات السابقة بحروف صغيرة .

كما يمكن مع الاعلان عن المتغير تخصيص قيمة له منذ البداية. ومن أكثر أنواع المتغيرات استعمالاً :

2 . 6 . 1 المتغيرات الحرفية Characters Variables

وهي المتغيرات الحرفية التي تحتوي حرف أو أكثر ويتم الاعلان عنها كمتغيرات في بداية البرنامج قبل استخدامها داخله، وتستوعب خانة واحدة فقط لكل متغير ويعلن عنها بواسطة .char

مثال 11:

```
Char a,b;
a = '?';
b = '&';
```

الاعلان السابق يبين أن المتغيرين حرفيين ويسمح بتخصيص رمز واحد لكل منهما.

2 . 6 . 2 المتغيرات الصحيحة Integer Variables

وهي متغيرات تسمح بتخزين العدد الصحيح فيها سواء كان سالب او موجب، ويعلن عن المتغير من هذا النوع بالعبارة int وإذا كانت المتغير الصحيح طويل نستخدم long.

مثال 12: المثال التالي يوضح أن المتغيرات a,b من النوع الصحيح القصير :

```
Int a , b ;
```

في حين الاعلان التالي :

```
long c
```

يعني أن المتغير c هو من النوع الصحيح الطويل وله سعة أكبر ومداه أكبر من العدد الصحيح القصير.

مثال 13: يبين هذا المثال الاعلان عن المتغير الصحيح مع تخصيص قيم صحيحة لكل منها :

```
Int i = 500000;
```

```
Int j = 600000;
```

```
Int k=i+j;
```

لو تم وضع هذه الجمل في برنامج مع امر الطباعة وبالتالي تنفيذه سنحصل على قيم غير صحيحة ، لأن المتغير صحيح طويل وينبغي تعريفه باستخدام long كالتالي :

```
long i = 500000;
```

```
long j = 600000;
```

```
long k=i+j;
```

وفي هذه الحالة يكون ناتج خزن ما سبق صحيحاً ويظهر بالشكل التالي:

$l=500000$ $j=600000$ $k=1100000$

2 . 6 . 3 المتغيرات الحقيقية Float Variables

هي التي تخزن فيها المتغيرات الرقمية التي تحتوي على فاصلة عشرية ويجب أن يعلن عنها بواسطة العبارة float .

مثال 14:

float A , B ;

A=10.5 ;

B=15.4 ;

2 . 6 . 4 المتغيرات مضاعفة الدقة Double variables

هي تلك المتغيرات التي تحتوي أعداداً صحيحة وحقيقية لكنها مضاعفة أو دقيقة جداً ، ويعلن عن هذا النوع بالكلمة (double) .

مثال 15:

double a , b ;

a=100.9 ;

b=300.7 ;

2 . 7 التعليقات Comments

هي عبارة عن بعض الأوامر التوضيحية ينظر القارئ إليها وكأنها مرشد ولا يكون لها أي تأثير داخل البرنامج ، لأنها ليست جزء منه بقدر ماهي وسيلة إيضاحية، وفي الغالب يبدأ التعليق بالعلامتين (//) وينتهي بنهاية السطر وإذا كان التعليق أطول فإنه يبدأ ب(/*) وينتهي ب(/*).

مثال 15:

```
// This program is for add two numbers
```

العبارة السابقة إذا كتبت في البرنامج لا تدخل في تنفيذه بل للتوضيح فقط.

مراجع المحاضرة الثانية :

1. بشير القايد ، اساليب البرمجة بلغة ++C، منشورات ELGA ، 2005 .

2. <http://ar.wikipedia.org>

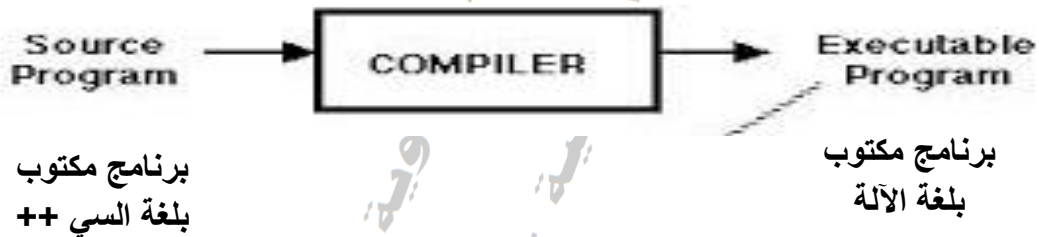
3. <http://vc4arab.com/showthread.php?t=854>

المحاضرة الثالثة

أساسيات كتابة برنامج بلغة ++C

3 . 1 شكل البرنامج

علمنا في المحاضرة الأولى أن البرنامج هو عبارة عن مجموعة من الاوامر والتعليمات المكتوبة بشكل منطقي من قبل المبرمج أو مجموعة من المبرمجين ، كل أمر أو تعليمية هي توجيهة للحاسوب لأداء عملية معينة من المسألة المعطاة، وعند تنفيذ هذا البرنامج يتم ترجمته إلى لغة الآلة (Machine language) عن طريق المترجم (Compiler)، وبالتالي كشف وتسجيل الأخطاء الواردة فيه وتبليغ المبرمج بهذه الأخطاء .



والبرنامج في لغة سي ++ يأخذ الشكل العام التالي:-

```

<header files>   ملفات العناوين
Main ( )
{
Variables Declarations   إعلانات داخلية
    statement_1;
    statement_2;
    .....
    statement_last;
return 0;
}
  
```

مثال 1: أكتب برنامج يطبع العبارة : C++ is good language

```
// This is program 1
#include <iostream.h>
main()
{
cout<< "c++ is good language";
return 0 ;
}
```

حيث

السطر الاول كتبت به العبارة (//This is program 1) وهي مسبوقة ب // وبالتالي هي جملة تعليق لا تدخل في البرنامج وإنما تعتبر توضيح على البرنامج.

السطر الثاني(> # include <iostream.h) وهو سطر يسمح باستخدام قناتي الادخال (istream) والايخراج (ostream) تحت اسم (iostream.h).

السطر الثالث الدالة الرئيسية main والتي يبدأ بها أي برنامج وتدل على بداية جمل البرنامج. أما أمر الاخراج cout فيقوم بطباعة ما موجود بين علامتي التنصيص على الشاشة ، مع ملاحظة أن كل جملة يجب أن تنتهي بفاصلة منقوطة .

السطر الرابع جملة (return) والتي تدل على نهاية البرنامج وعادة ما تنتهي بالقيمة 0 ، وينتهي البرنامج بالقوس المغلق .

3 . 2 قنوات الادخال والايخراج

تعتمد لغة سي ++ في تصميم أدوات الادخال والايخراج على أساس القنوات (streams) والتي يمكن توصيلها بأحد الأطراف مثل لوحة المفاتيح والشاشة والطابعة .

ومن أشهر القنوات الخاصة بالإدخال والايخراج على ثلاثة فصائل وهي :

1. فصيلة (istream) لعمليات الادخال .

2. فصيلة (ostream) لعمليات الاخراج.

3. فصيلة (iostream) لعمليات الإدخال والإخراج معا.

3 . 2 . 1 هدف قناة الإدخال (عبارة cin)

وهي عبارة عن أمر او دالة لإدخال قيم وقراءتها عن طريق لوحة المفاتيح ، وإسناد قيم إلى المتغيرات المعروفة داخل البرنامج من خلال ادخال المستخدم لقيمتها لغرض معالجتها والقيام بالعمليات المطلوبة.

ومن أشهر دوال الإدخال هي العبارة cin والشكل العام لها :

```
cin >>var;
```

حيث cin هي دالة إدخال المتغيرات.

>> الاسم المزدوجة المتجهة لليمين تعني مؤثر قناة الإدخال ويستخدم جنباً إلى جنب مع cin . var وهي متغير لقيم يتم استقبالها من خلال لوحة المفاتيح ومن ثم تخزينها في ذاكرة الحاسب الآلي تحت أسماء هذه المتغيرات.

مثال 2 : مثلاً cin>>a; تقرأ متغير باسم a .

3 . 2 . 2 هدف قناة الإخراج (عبارة cout)

وهي عبارة عن أمر او دالة تستخدم لإخراج وعرض نواتج العمليات أمام المستخدم في شاشة التنفيذ .

ومن أشهر دوال الإخراج هي العبارة cout والشكل العام لها :

```
cout <<var;
```

حيث cout هي دالة إخراج وطباعة المتغيرات.

الاسم المزدوجة المتجهة لليساار << تعني مؤثر قناة الإخراج وتستخدم جنباً إلى جنب مع cout . var وهي تعبيرات يمكن أن تكون ثوابت عددية أو قيم لمتغيرات من النوع الصحيح او الحقيقي أو الحرفي المطلوب إخراجها على شاشة العرض.

مثال 3 : cout<<a; طباعة قيمة المتغير a على الشاشة.

3 . 3 الاعلان عن المتغيرات

في المحاضرة السابقة درسنا أنواع المتغيرات وكيف يتم الاعلان عنها ، وما يجب أن نفهمه جيداً، أن أي متغير يتم استخدامه داخل البرنامج لابد من الاعلان عنه وتعريفه في بداية جسم

البرنامج بعد main مباشرةً ، وإذا تم استخدام متغير داخل البرنامج ولم يتم الاعلان عنه فإن البرنامج لن ينفذ وسيظهر خطأ.

والجدول التالي يوضح كيفية الاعلان عن أهم انواع المتغيرات في لغة ++C

تعريف المتغير	استخدامه
Int var;	يستخدم لتعريف المتغير من نوع integer أي رقمي مثلا (int x=5)
Float var;	يستخدم لتعريف المتغير من نوع كسري مثلا (Float var=5.4;)
Char var;	يستخدم لتعريف المتغير من نوع حرفي مثلا (Char var="a;")

مع ملاحظة انه يمكن تعريف أكثر من متغير واحد من نوع واحد حسب البرنامج وما يحتوي من متغيرات.

3 . 4 تمثيل العمليات الرياضية في لغة سي ++

تمثل العمليات الرياضية برمجيًا بطريقة مشابهة لطريقة تمثيلها رياضياً مع تغيير طفيف بالرموز الرياضية لما يكافأها من الرموز البرمجية ، ولاحظ هذا الجدول التوضيحي للعمليات وتمثيلها رياضياً وبرمجياً.

لنفرض أن لدينا متغيران (a,b) ونتاج العملية الرياضية يخزن في c .

الرمز و الوظيفة	تمثيله رياضيا	تمثيله برمجيا
الجمع (+)	C=a+b	C=a+b;
الطرح (-)	C=a-b	C=a-b;
القسمة (/)	$C=\frac{a}{b}$	C=a/b;
الضرب (*)	C=a*b	C=a*b;
باقي القسمة (%)	C=a mod b	C=a%b;

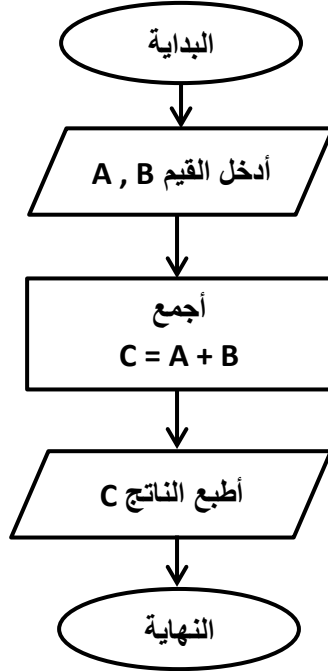
3 . 5 الرموز العلائقية في لغة سي ++

هي رموز تستخدم لمعرفة العلاقة بين الرقمين أي هل يساوي أو أكبر أو أصغر أو لا يساوي والجدول التالي يوضح هذه الرموز ونتيجة المقارنة إما تكون (True) أو (False).

الرمز	الوظيفة	مثال
>	علامة اكبر	(a>b)
>=	علامة اكبر أو يساوي	(a>=b)
<	علامة أصغر	(a<b)
<=	علامة أصغر أو يساوي	(a<=b)
==	علامة اليساوي	(a==b)
!=	علامة لا يساوي	(a!=b)

مثال 4: أكتب خوارزمية وارسم المخطط الانسيابي مع كتابة برنامج بلغة السي ++ لإيجاد ناتج جمع عددين صحيحين (a , b) والناتج يكون c .

ثانياً : المخطط الانسيابي



أولاً : الخوارزمية

1. البداية
2. أدخل القيم A , B
3. أجمع $C = A + B$
4. أطبع الناتج C
5. النهاية

ثالثاً : البرنامج

```

// This is program 2
#include <iostream.h>
main()
{
int a,b,c;
cin>>a>>b;
c = a + b;
cout<< c;
return 0 ;
}
  
```

مراجع المحاضرة الثالثة :

1. بشير القايد ، اساليب البرمجة بلغة ++C، منشورات ELGA ، 2005 .
2. <https://www.cs.uaf.edu/~cs631/node4.html>
3. حسين الربيعي ، خطوة بخطوة لتعلم لغة ++C & C ، متاح في هذا الموقع <http://www.kutub.info/library/book/12961>

المحاضرة الرابعة

بعض المهارات في جمل الإدخال والإخراج في لغة ++C

4 . 1 حروف الهروب

هي رموز خاصة يطلق عليها حروف الهروب حيث تستخدم لأغراض خاصة مع قناة الإخراج (cout) للتحكم في المخرجات على شاشة العرض وتبدأ هذه الرموز بالخط المائل (\) على أن يكون ضمن علامة التنصيص المزدوجة (") ومن أمثلة حروف الهروب :-

1. الرمز \n : ويستخدم للقفز إلى سطر جديد.

2. الرمز \t : التقدم 7 مسافات عمودية قبل الطباعة.

3. الرمز \b : مسافة إلى الخلف .

مثال 1 : البرنامج التالي يحتوي على أكثر من قناة إخراج .

```
#include <iostream.h> // for cout
Void main(void)
{
cout<< "My name is Ahmed";
cout<< "her name is Asma";
cout<< "What is ypur name please?";
return 0 ;
}
```

البرنامج السابق سيطبع My name is Ahmedher name is AsmaWhat is ypur name please?

والملاحظ في السطر السابق أنها ظهرت في سطر واحد ولا يوجد فراغ بين مخرجات الجمل .

وليكون البرنامج يقوم بكتابة كل جملة في سطر نعيد كتابة البرنامج بالشكل التالي :

```
#include <iostream.h> // for cout
Void main(void)
{
cout<< "My name is Ahmed \n";
cout<< "her name is Asma \n";
cout<< "What is ypur name please?";
return 0 ;
}
```

سيقوم البرنامج السابق بطباعة الجمل السابقة كل جملة في سطر نظراً لاستخدام أحد حروف الهروب والطباعة ستكون على النحو التالي :

My name is Ahmed
her name is Asma
What is ypur name please?

وقد تكتب جمل البرنامج السابق بالشكل التالي :

```
cout<< "My name is Ahmed \nher name is Asma \nWhat is ypur name please?";
```

وتقوم بطباعة الشكل كل سطر على حدة .

4 . 2 أدوات التشكيل

تقدم لنا لغة سي ++ مجموعة من أدوات التشكيل تستخدم في إدخال وإخراج البيانات ومن أهمها أداة endl وهي تستخدم مع قنوات الاخراج لفتح سطر جديد .
نفس المثال السابق يمكن إعادة كتابته بالشكل التالي :

```
#include <iostream.h> // for cout
Void main(void)
{
cout<< "My name is Ahmed"<<endl
    << "her name is Asma "<<endl
    << "What is ypur name please?";
return 0 ;
}
```

سيطبع البرنامج التالي :-

My name is Ahmed
her name is Asma
What is ypur name please?

حيث تشير endl إلى الحاسب بالذهاب إلى سطر جديد على شاشة العرض .

4 . 2 جمل التخصيص

وهي تأخذ الشكل العام التالي :

VariableName = Expression;

حيث VariableName الواقع على يسار رمز التخصيص = يجب ان يكون متغير .
Expression : هو نوع من التعابير الأولية أو المعقدة أو الثابتة أو المتغيرة.

قيمة ثابتة A = 3.14;

قيمة متغيرة Result = total;

تعبير أولي H = R + 2.3 + W;

تعبير معقد N = (q*(b*k+m))/f;

مثال 2 : لنفرض أن لدينا المتغيرات A,B,C من النوع الحقيقي وتم تخصيص 5.55 و 7.77 إلى المتغيرين A,B على التوالي والمتغير C هو حاصل جمع المتغيرين A,B .

```
#include <iostream.h> // for cout
main()
{
float A=5.55;
float B=7.77;
float C=A+B;
cout<< "A="<<A<<"B="<<b<<"C="<<c;
return 0 ;
}
```

سوف يقوم البرنامج بطباعة الناتج التالي :

A= 5.55 B=7.77 C=13.32

مراجع المحاضرة الرابعة :

1. بشير القايد ، اساليب البرمجة بلغة ++C، منشورات ELGA ، 2005 .

المحاضرة الخامسة

جمل الاختيارات في لغة ++C

تناولنا في المحاضرات السابقة مقدمة عن لغة سي ++ وبعض جمل الادخال والايخارج وكيف نكتب برنامج باستخدام هذه الجمل ، وفي هذه المحاضرة نتناول جمل الاختيارات والتي تعد جمل (If) إذا الشرطية والتي يمكن أن تكون في عدة صور ويستفاد منها برمجياً في حل العديد من المشكلات وتصميم العديد من البرامج التي تحتوي شروطاً معينة.

5 . 1 جملة إذا The If statement

الهدف منها تنفيذ جملة في حالة تحقق الشرط والشكل العام لها

if (Logical Expression)

Yes_statement

Next statement

يتم تنفيذ الجملة الثانية Yes_statement إذا كانت نتيجة الشرط Logical Expression والتي يجب أن توضع بين قوسين () صحيحة ، أما إذا كانت النتيجة خاطئة فيذهب إلى الجملة التالية (Next statement) مع ملاحظة أن هذه الجملة تنفذ سواء أكان الشرط صحيحاً أو خاطئاً .

مثال 1 : أكتب برنامج يقوم بقراءة درجة الطالب وطباعة (PASS) ودرجة الطالب إذا كانت درجة الطالب أكبر من 50 ، وطباعة درجة الطالب فقط إذا كانت أقل من 50.

```
#include <iostream.h>
main()
{
float grad;
cout<<"Enter the grad";
cin>>grad;
if (grad>=50)
cout<< "PASS\n";
cout<<grad;
return 0 ;
}
```

عند تنفيذ البرنامج يطبع الجملة Enter the grad وهي جملة تم طباعتها باستخدام الامر cout قبل أمر إدخال الدرجة في البرنامج والغرض منها التوضيح فقط ، ثم نقوم بإدخال الدرجة.

على فرض أننا ادخلنا 70 ففي هذه الحالة الشرط الذي بين القوسين صحيح (أي أن الدرجة هي أكبر من 50) وبالتالي سيقوم بطباعة الكلمة (PASS) وطباعة الدرجة (70) في السطر الذي يليها باعتبارنا استخدمنا \n كحرف هروب للسطر الذي يليه (وقد درسنا حروف الهروب في المحاضرة السابقة).

أما إذا أدخلنا بعد التنفيذ الرقم 40 فلن يتم تنفيذ الشرط ولن ينفذ الجملة التي بعد الشرط مباشرة بل سينفذ الجملة التي تليها مباشرة وهي طباعة الدرجة فقط.

5 . 2 جملة إذا - فإن - وإلا The if - else statement

تسمى هذه الجملة بالجملة المتكاملة وتعني أنه إذا تحقق الشرط فافعل كذا وإلا فأفعل كذا والصيغة العامة لها :

```
if (Logical Expression)
    Yes_statement;
else
    No_statement;
Next statement;
```

هنا لو تحقق الشرط (Logical Expression) عندها سيتم تنفيذ الجملة Yes_statement وإذا لم يتحقق الشرط فسيتم تنفيذ الجملة No_statement أما الجملة Next statement فتتفد في كلا الحالتين، وكلمة else هنا تعني وإلا ، أي أن هذه الجملة كاملة يتم استخدامها بالصيغة أعلاه في حالة كانت المشكلة تحتوي على حالتين لا ثالث لهما.

مثال 2 : أكتب برنامج يقوم بقراءة درجة الطالب وطباعة (PASS) إذا كانت درجة الطالب أكبر من 50 ، وطباعة (FAIL) إذا كانت درجة الطالب أقل من 50.


```
#include <iostream.h>
main( )
{
float grad;
cout<<"Enter the grad";
cin>>grad;
if (grad>=50)
    cout<< "PASS ";
else
    cout<< "FAIL";
return 0 ;
}
```

عند تنفيذ البرنامج يطلب منا إدخال الدرجة (Enter the grad) ويتم إدخال الدرجة .
على فرض أننا أدخلنا الرقم 70 فسيتم طباعة العبارة (PASS) أي أن الشرط (if (grad>=50) قد تحقق .
أما إذا قمنا بإدخال الدرجة 35 فسيتم طباعة العبارة (FAIL) أي أن الشرط (if (grad>=50) لم يتحقق
وبالتالي الذهاب للجملة التي تلي العبارة else .

5 . 3 جملة إذا المتداخلة The Nested if statement

وهي عبارة عن تآلف مجموعة من جمل if مع عدد من جملة if else ، بمعنى أن تكون
جملة if بداخلها if أخرى وهكذا ، والصيغة العامة لها :

```
if (condition1)
statement1;
else if(condition2)
statement2;
:
:
else if(condition n)
statement n;
else
statement;
```

والصيغة السابقة تعني إذا تم تنفيذ الشرط الاول (condition1) فنذهب لتنفيذ statement1 وإلا فنذهب لاختبار الشرط الثاني حتى نصل إلى الشرط (n) ، وإذا لم يتم تنفيذ الشرط (n) فيتم تنفيذ الجملة الأخيرة بعد else .

مثال 3 : أكتب برنامج لإدخال عدد ثم طباعة (Positive) إذا كان موجب و (Negative) إذا كان سالب وطباعة (0) إذا كان العدد صفر .

```
#include <iostream.h>

main( )
{
int x;
cin>>x;
if(x>0)
    cout<<"x is positive";
else if (x<0)
    cout<<"x is negative";
else
    cout<<"x is 0";
return 0 ;
}
```

الملاحظ في المثال السابق أن لدينا ثلاث خيارات وبالتالي فإننا لا نستطيع استخدام عبارة if التي تحتل خيار واحد فقط أو عبارة if else والتي تحتل خيارين ، وما علينا هنا إلا استخدام عبارة if المتداخلة لأن الخيارات في البرنامج ثلاثة هي :

1. العدد موجب.
2. العدد سالب
3. العدد صفر

وبالتالي إذا أدخلنا الرقم 9 سيتم طباعة العبارة x is positive لأن العدد موجب ، وإذا أدخلنا الرقم 5- فسيقوم بطباعة العبارة x is negative ، وإذا أدخلنا الرقم صفر فسيقوم بطباعة العبارة x is 0 .

5 . 4 جملة التحويل The switch Statement

كما لاحظنا في جملة if أن المقارنة تتم بين قيمتين حيث تكون النتيجة إما صحيحة أو خاطئة، ويمكن أن تكون المقارنة بين ثلاثة قيم أو أكثر باستخدام if المتداخلة ، لكن ذلك يتطلب دقة عالية وصعوبة مع زيادة حجم الاختيارات أو القيم المقارنة، خاصةً أن حل المسألة قد يفرض على المبرمج المقارنة بين عدد من القيم تبعاً لشروط مختلفة .

ولحل هذه المعضلة يوجد أمر التبديل أو التحويل switch الذي يقوم بعدة تحويلات مختلفة.

والشكل العام لهذه الجملة ما يلي :-

```
switch( Variable )
```

```
{
```

```
    case Value1 : Statements;
```

```
    case Value2 : Statements;
```

```
    case Value3 : Statements;
```

```
    default : statements;
```

```
}
```

حيث

- ✓ Variable: متحول ما مهما كان نوعه (صحيح أو حقيقي أو منطقي ...) و يوضع بين قوسين بعد العبارة. switch
- ✓ Value: قيمة ما نريد معرفة إذا كان المتغير يحتويها .. و توضع بعد العبارة (case انتبه للقيم الحرفية بحيث يجب أن تضع الحاصرات إما المزدوجة أو الفردية .
- ✓ Statements : ممكن أن تكون تعليمة واحدة أو عدة تعليمات.
- ✓ default: يتم تنفيذها فقط عند عدم وجود أي قيمة من تلك القيم في المتحول.
- ✓ يجب وضع : الأقواس ({ }) إذا أردنا التحقق من أكثر من قيمة .. أما لو كنا نريد التحقق من قيمة واحدة فممكن أن لا نضع تلك الأقواس (مثل العبارة .. (if و لكن إن تحققنا من قيمة واحدة و وضعنا أيضاً العبارة default فعندها أصبحت switch تحتوي على أكثر من تعليمة وبالتالي يجب وضع الأقواس ({ }) .

مثال 4: بفرض أننا جعلنا المستخدم يدخل قيمة لعدد صحيح ضمن المتحول n لتحديد أيام الأسبوع بحيث أن العدد 1 ليوم السبت و 2 للأحد ... إلخ:

```
#include <iostream.h>

main()
{
int n;
cin>>n;

switch( n )
{
case 1 : cout << "Sat\n"; break;
case 2 : cout << "Sun\n"; break;
case 3 : cout << "Mon\n"; break;
case 4 : cout << "Tue\n"; break;
case 5 : cout << "Wed\n"; break;
case 6 : cout << "Thu\n"; break;
case 7 : cout << "Fri\n"; break;

default : cout << "Error...\n";
}
}
```

ملاحظة 1/ من أجل توضيح عمل .. break بفرض أن المستخدم أدخل الرقم 4 و بفرض أننا لم نكتب الأمر break عندها سيتم كتابة أيام Tue و Wed و Thu و Fri على الشاشة كما قلنا .. هنا تكمن أهمية هذا الأمر فهو يجعل المترجم يتجاهل باقي القيم التي تلي القيمة التي نريدها و يخرج إلى خارج كتلة (عبارة switch) و سيتم تنفيذ التعليمة التي تلي تلك البنية أو الكتلة.. وبالتالي ستظهر Tue فقط على الشاشة.

ملاحظة 2/ يستخدم الأمر break للخروج من الكتلة التي يوجد فيها فقط .. و يمكن أن يستخدم في أي من الأوامر و العبارات التي هي موضوع درسنا ما عدا البنية if .

5 . بعض البرامج والتمارين على جملة if و switch

1. أكتب برنامج يقوم بإدخال القيمة الصحيحة X ويطبع العبارة large value إذا كانت القيمة أكبر من 100.

```
#include<iostream.h>
main( )
{
int x;
cout<<"enter x:";
cin>>x;
if ( x>=100)
cout<<"large value";
}
```

2. أكتب برنامج لإدخال ثلاثة قيم صحيحة ثم طباعة القيمة الأكبر.

```
#include<iostream.h>
main( )
{
int x,y,z;
cout<<"enter x,y,z";
cin>>x>>y>>z;
if(x>y && x>z)
cout<<"max="<<x;
else if (y>x && y>z)
cout<<"max="<<y;
else
cout<<"max="<<z;
}
```

3. أكتب برنامج لإدخال عدد ثم تحديد ما إذا كان العدد فردي أم زوجي .

```
#include<iostream.h>
main()
{
int number;
cout<<"enter number :";
cin>>number;
if (number % 2 ==0)
cout<<"the"<<number<<"is even number ";
else
cout<<"the"<<number<<"is odd number";
}
```

4. أكتب برنامج لإدخال قيمتين صحيحتين x , y ثم إجراء العمليات الحسابية (الجمع، الطرح، الضرب، القسمة، باقي القسمة) على هاتين القيمتين، مستخدماً الجملة `switch`.
مثلاً : إذا كانت المدخلات $x=5$ ، $y=10$ ، والمؤثر $op = +$ ، فإن الناتج سيكون $15=10+5$.

```
#include<iostream.h>
main()
{
int x,y;
char op;
cout<<"enter two numbers\n";
cin>>x>>y;
cout<<"enter the operator\n";
cin>>op;
switch(op)
{
case'+':cout<<x<<"+"<<y<<"="<<x+y;
break;
case'-':cout<<x<<"-"<<y<<"="<<x-y;
break;
case'*':cout<<x<<"*"<<y<<"="<<x*y;
break;
case'/':cout<<x<<"/"<<y<<"="<<x/y;
break;
case'%':cout<<x<<"%"<<y<<"="<<x%y;
break;
default:cout<<"i don't know the operator:"<<op;
}
}
```

مراجع المحاضرة الخامسة :

1. بشير القايد ، اساليب البرمجة بلغة ++C ، منشورات ELGA ، 2005 .
2. حذيفة عبدالرحمن ، البرمجة بلغة ++C وتراكيب البيانات ، متاح في هذا الموقع <http://faculty.ksu.edu.sa> .
3. دروس كاملة للمبتدئين في أساسيات ++C ، متاح في هذا الموقع : موقع فيجوال سي للعرب.

المحاضرة السادسة

الإجابة النموذجية لامتحان النصفى




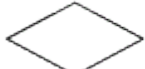
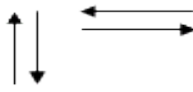
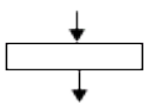

أجب عن جميع الأسئلة الآتية:

س1. حسب فهمك ما الفرق بين كلاً من :

أ. المخطط الانسيابي والخوارزمية.

6. الخوارزميات **Algorithm** : الخوارزمية هي مجموعة من الخطوات الرياضية والمنطقية والمتسلسلة اللازمة لحل مشكلة ما، وقد تكون هذه الخطوات باللغة العادية وعند تتبعنا لهذه الخطوات نصل للحل النهائي، وسميت الخوارزمية بهذا الاسم نسبة إلى العالم المسلم أبو جعفر محمد بن موسى الخوارزمي الذي ابتكرها في القرن التاسع الميلادي.

7. **مخطط سير العمليات Flowchart** : يسمى أيضاً بالمخطط الانسيابي ، وهو عبارة عن طريقة تخطيطية تمثل تتابع الأحداث بأشكال رمزية وخطوط تمثل مسار عمليات البرنامج المنطقية ، كل شكل من هذه الأشكال يعبر عن نوع الأمر أو التعليم في حل المسألة، وهي ترجمة تخطيطية للخوارزمية . وعند رسم خريطة سير العمليات لمسألة معينة فإننا نستخدم مجموعة من الأشكال الرمزية الاصطلاحية المبينة في الجدول التالي:

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

ب. لغات المستوى العالي ولغات المستوى الواطي.

1. لغات المستوى الواطي (L.L.L) Low Level Language : وهي اللغة التي تستخدم

(0,1) للتعبير عن الأوامر المختلفة التي يتكون منها البرنامج ، وهي لغة صعبة لا

يحسنها إلا من صمم الحاسبة نفسها، وتسمى لغة الآلة (Machine Language) .

2. لغات المستوى العالي (H.L.L) High Level Language : وهي اللغات الحديثة

المستخدمة في أجهزة الحاسوب وهي قريبة من لغة الانسان وتتميز بسهولة الكتابة وسهولة

اكتشاف الأخطاء ومن أمثلتها لغة البيسيك (Basic) ولغة الفورتران (Fortran) ولغة

الباسكال (Pascal) ولغة السي (C) والسي بلس بلس (C++) ، وغيرها إلى جانب

اللغات المرئية الحديثة التي تتلاءم مع الوسائط المتعددة وتسمى اللغات المصورة مثل

الفيجوال بيسيك Visual Basic والدلفي Delphi .

ج. المتغيرات الصحيحة والمتغيرات الحقيقية من حيث الاعلان عنها كمتغيرات في بداية البرنامج.

1. المتغيرات الصحيحة Integer Variables

وهي متغيرات تسمح بتخزين العدد الصحيح فيها سواء كان سالب او موجب، ويعلن عن

المتغير من هذا النوع بالعبارة int وإذا كانت المتغير الصحيح طويل نستخدم long.

مثال: المثال التالي يوضح أن المتغيرات a,b من النوع الصحيح :

```
Int a , b ;
```

```
a = 15
```

```
b = 20
```

2. المتغيرات الحقيقية Float Variables

هي التي تخزن فيها المتغيرات الرقمية التي تحتوي على فاصلة عشرية ويجب أن يعلن عنها

بواسطة العبارة float .

مثال:

```
float A , B ;
```

```
A=10.5 ;
```

```
B=15.4 ;
```


د. قناة الإدخال وقناة الإخراج في البرنامج.

1. قناة الادخال (عبارة cin)

وهي عبارة عن أمر او دالة لإدخال قيم وقراءتها عن طريق لوحة المفاتيح ، وإسناد قيم إلى المتغيرات المعروفة داخل البرنامج من خلال ادخال المستخدم لقيمها لغرض معالجتها والقيام بالعمليات المطلوبة.

ومن أشهر دوال الادخال هي العبارة cin والشكل العام لها :

```
cin >>var;
```

حيث cin هي دالة إدخال المتغيرات.

>> الاسهم المزدوجة المتجهة لليمين تعني مؤثر قناة الادخال ويستخدم جنباً إلى جنب مع cin .

var وهي متغير لقيم يتم استقبالها من خلال لوحة المفاتيح ومن ثم تخزينها في ذاكرة الحاسب الالى تحت أسماء هذه المتغيرات.

مثال : مثلاً cin>>a; تقرأ متغير باسم a .

2. قناة الاخراج (عبارة cout)

وهي عبارة عن أمر او دالة تستخدم لإخراج وعرض نواتج العمليات أمام المستخدم في شاشة التنفيذ .

ومن أشهر دوال الاخراج هي العبارة cout والشكل العام لها :

```
cout <<var;
```

حيث cout هي دالة إخراج وطباعة المتغيرات.

الاسهم المزدوجة المتجهة لليسار << تعني مؤثر قناة الاخراج وتستخدم جنباً إلى جنب مع cout .

var وهي تعبيرات يمكن أن تكون ثوابت عددية أو قيم لمتغيرات من النوع الصحيح او الحقيقي أو الحرفي المطلوب إخراجها على شاشة العرض.

مثال: cout<<a; طباعة قيمة المتغير a على الشاشة.

س2. أ. أكتب برنامج بلغة سي ++ لإيجاد المتوسط الحسابي لثلاث أعداد؟

```
# include <iostream.h >
main()
{
int a,b,c;
float avg;
cout<<"Enter three number";
cin>>a>>b>>c;
avg = a+b+c/3;
cout<<avg;
return 0;
}
```

ب. أكتب برنامج يقوم بطباعة اسمك وعمرك على الشاشة على أن تكون في سطرين؟

```
# include <iostream.h >
main()
{
cout<<"My name is Ahmed"<<"\n" ;
cout<<"I am 25";
return 0;
}
```

ج. أكتب برنامج بلغة السي ++ لإيجاد قيمة y إذا كانت:

$$x \geq 10 \quad y = x - 10$$

$$x < 10 \quad y = x + 10$$

```
# include <iostream.h >
main()
{
int x,y;
cin>>x;
```

```

if (x>=10)
    y= x-10;
else
    y=x+10;
cout<<y;
}

```

س3. أ. أوجد حل المعادلة التالية مستخدماً قواعد أسبقية العمليات إذا كانت $b=2$, $c=3$, $d=4$

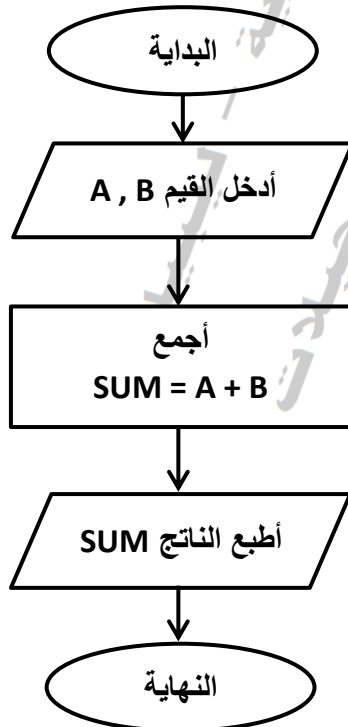
$$A = b^2 + 2c + 4/d$$

$$A = 2*2 + 2*3 + 4/4$$

$$A = 4 + 6 + 1$$

$$A = 11$$

ب. أكتب الخوارزمية وارسم المخطط الانسيابي لبرنامج يقوم بحساب مجموع عددين؟



1. البداية
2. أدخل القيم A , B
3. أجمع SUM = A + B
4. أطبع الناتج SUM
5. النهاية

ج. ما هو ناتج تنفيذ البرنامج التالي :

```
# include <iostream.h >
main()
{
int a,b;
a=18;
b=5;
cout<<a+b<<"\n";
cout<<a-b<<"\n";
cout<<a*b<<"\n";
cout<<a/b<<"\n";
cout<<a%b<<"\n";
return 0;
}
```

23

13

90

3.6

3

انتهت الاسئلة

المحاضرة السابعة

الحلقات التكرارية (1)

6 . 1 جملة لأجل The for statement

يقوم هذا الأمر بتنفيذ تعليمة أو عدة تعليمات أكثر من مرة و لعدد محدد من المرات. بحيث يتم تعريف متحول لعدد صحيح يمثل عدد المرات التي سيتكرر فيها تنفيذ مجموعة من التعليمات. والشكل العام لها :

```
for(Expression1; Expression1; Expression1)
    statement;
next statement;
```

حيث

Expression1 : القيمة الابتدائية التي تحدد للمتغير على أنه عداد.

Expression2 : شرط استمرار حلقة التكرار .

Expression3 : جملة الزيادة والنقصان للعداد في الحلقة .

أما في حالة إحتواء جملة for على أكثر من جملة عندها يتم ضمها بين قوسي الفئة .

Ex: for(int i=1; i<10; i++)

في المثال السابق i هي بداية العداد وقيمتها الابتدائية هي 1 .

الشرط يجب ألا تزيد قيمة i عن 9 أي اصغر من 10 .

الإضافة ++i إضافة 1 لـ i حتى ينتهي الشرط.

مثال 1: أكتب برنامج لطباعة الأرقام من 1 - 10 تصاعديا .

```
#include <iostream.h>
main( )
{
int s;
for(s=1;s<=10;s++)
    cout<<'S="<<s;
return 0 ;
}
```

سيقوم البرنامج بطباعة التالي :

S=1 S=2 S=3 S=4 S=5 S=6 S=7 S=8 S=9 S=10

حيث أن :

التعبير الأول s=1 أي الاعلان عن المتغير كعداد للحلقة وتخصيص القيمة الابتدائية له وهي 1.
التعبير الثاني s<=10 يمثل شرط الحلقة ، ويعني نفذ الجملة الموالية لجملة for أي طباعة قيمة المتغير s طالما أن قيمة العداد s لم تتجاوز العدد 10 .
التعبير الثالث s++ يعني زيادة قيمة العداد s بالقيمة 1 بعد كل خطوة تنفذ فيها جملة الطباعة.

أما إذا اردنا طباعة الأرقام تنازلياً من 10 - 1 فنقوم بكتابة جملة for على النحو التالي :
for(s=10;s>=1;s--)

سيطبع البرنامج التالي :

S=10 S=9 S=8 S=7 S=6 S=5 S=4 S=3 S=2 S=1

مثال 2 : أكتب برنامج لطباعة الارقام من 1-10 ومربعاتها.

```
#include<iostream.h>
main()
{
int i;
for(i=1;i<=10;i++)
cout<<i<<"\t"<<i*i<<"\n";
return 0;
}
```

نلاحظ أن جملة cout إحتوت على طباعة i أولاً ثم ازاحة مؤشر الطباعة 7 مسافات لاستخدام \t ، ثم طباعة المربعات i*i ثم الانتقال إلى سطر جديد بواسطة جملة الهروب \n .

وبالتالي سيكون شكل الطباعة كالتالي

```
1      1
2      4
3      9
4      16
.
.
.
10     100
```

مثال 3 : أكتب برنامج لإيجاد مجموع الأعداد

$$\text{Sum} = 4.5 + 5.0 + 5.5 + \dots + 10.0$$

الحل

```
#include<iostream.h>
main()
{
float sum = 0.0;
for(float a=4.5;a<=10;a+=0.5)
    sum+=a;
cout<<"THE SUM IS"<<sum
return 0;
}
```

حيث :

- . التعبير الأول $a=4.5$ ويمثل البداية بالقيمة الحقيقية float .
- . التعبير الثاني $a<=10$ ويمثل شرط هذه الجملة.
- . التعبير الثالث $a+=0.5$ وهي زيادة المتغير a بالقيمة 0.5 .

مثال 4 : أكتب برنامج لحساب مضروب n $n!=1.2.3.4.....n$

مثل :

$$4! = 4 * 3 * 2 * 1 = 24$$

$$5! = 5 * 4 * 3 * 2 * 1 = 120$$

الحل

```
#include<iostream.h>
main()
{
int n,f=1,i;
cout<<"enter n =";
cin>>n;
for (i=1;i<=n;i++)
f*=i;
cout<<"factorial="<<f;
}
```

فمثلاً لو قمنا بإدخال 5 سيكون الناتج على النحو التالي :

$$\text{factorial} = 120$$

مثال 5 : أكتب برنامج لإيجاد جمع المتسلسلة الآتية :

$$1 + \frac{1}{2} + \frac{1}{3} + \frac{1}{4} + \dots + \frac{1}{n}$$

حيث n يتم إدخالها من قبل المستخدم

```
#include<iostream.h>
main()
{
float n,i, sum=0;
cout<<"enter n:";
cin>>n ;
for (i=1 ; i<=n ; i++)
sum+=1/i;
cout<<"sum="<<sum ;
}
```

انتهت المحاضرة

مراجع المحاضرة السابعة :

1. بشير القايد ، اساليب البرمجة بلغة ++C ، منشورات ELGA ، 2005 .
2. حذيفة عبدالرحمن ، البرمجة بلغة ++C وتراكيب البيانات ، متاح في هذا الموقع <http://faculty.ksu.edu.sa> .
3. دروس كاملة للمبتدئين في أساسيات ++C ، متاح في هذا الموقع : موقع فيجوال سي للعرب.

المحاضرة الثامنة

الحلقات التكرارية (2)

8 . 1 العبارة: (while)

تقوم هذه العبارة بتكرار مجموعة من التعليمات عدد غير معروف من المرات طالما أن الشرط محقق.

والشكل العام لها كالتالي:

```
while( Condition )
    Statements;
```

حيث أن:

Condition : هو الشرط .. يكتب تماماً مثل العبارة. if

Statement: التعليمات المراد تكرارها طالما أن الشرط محقق نتيجه True .

مثال 1: أكتب برنامج لطباعة الارقام من 1 - 10 تصاعديا باستخدام عبارة While.

```
#include <iostream.h>
main( )
{
int s=1;
while(s<=10)
{
cout<<'S="<<s;
s+=1;
}
return 0 ;
}
```

سيقوم البرنامج بطباعة التالي :

S=1 S=2 S=3 S=4 S=5 S=6 S=7 S=8 S=9 S=10

في بداية البرنامج تم تخصيص القيمة 1 إلى المتغير S قبل الدخول إلى حلقة التكرار ، يأتي بعده الشرط $s \leq 10$ وطالما الشرط صحيح سيتم تنفيذ الجمل التي تلية بين القوسين، وهي زيادة قيمة العداد 1 طالما ان الشرط صحيح يتوقف التنفيذ حينما يكون الرقم المطبوع أصغر من أو يساوي 10 .

مثال 2: ما هو ناتج تنفيذ البرنامج التالي :

```
#include <iostream.h>
main()
{
int s=0;
while(s<=5)
    cout<<'S'<<s;
    s+=1;
return 0 ;
}
```

الجواب : الناتج سيكون كالتالي :-

S=1 S=1 S=1

نلاحظ أن الحل لا ينتهي يعني سيطبع S=1 إلى ما لا نهاية وهذه تسمى بظاهرة الدورة غير المنتهية Infinite Loop ويتم الخروج من هذه الحلقة بالضغط على المفاتيح Ctrl+Z ، وذلك لأن جملة الزيادة خارجة عن نطاق حلقة التكرار ولا بد من وضعها بين قوسي فئة {}.

مثال 3: أكتب برنامج لإيجاد متوسط n من الأعداد الصحيحة.

```
#include<iostream.h>
main()
{
int n , i , x, sum ;
float avg ;
sum=0;
i=1;
cout<<"enter n:";
cin>>n;
while (i<=n)
{
cout<<"enter x"<<i<<"\n";
cin>>x ;
sum+=x;
i++ ;
}
avg=sum/n;
cout<<"The average ="<<avg<<"\n";
}
```

8 . 2 العبارة: (do while)

نفس العبارة `while` تماماً و لكنها ستنفذ التعليمات الموجودة مرة واحدة على الأقل ثم بعد ذلك تختبر تحقق الشرط.

والشكل العام لها هو :

```
do
```

```
    Statements;
```

```
while( Condition );
```

حيث أن:

`Condition` : هو الشرط .. يكتب تماماً مثل العبارة. `if`.

`Statement`: التعليمات المراد تكرارها طالما أن الشرط محقق نتيجه `True` .

```
do
```

```
    x++;
```

```
while( y<0 );
```

ملاحظة 1 / لاحظ وجود الفاصلة المنقوطة في نهاية هذه العبارة .. أما في العبارة السابقة لم نضع فاصلة منقوطة بعد `while` و ذلك من أجل التمييز بين النوعين `while` و `do/while`.

ملاحظة 2 / يمكننا أيضاً في هاتين العبارتين استخدام أقواس الكتل ({ }) إذا كنا نريد تكرار عدد من التعليمات و ليس فقط تعليمة واحدة .. كما تعلمنا سابقاً.

مثال 4: أكتب برنامج يقوم بإيجاد مربعات الأعداد من 1-10

```
# include < iostream.h >
main( )
{
int i=1;
do
{
    cout << i*i<<"\n";
    i++;
}
while ( i<=10);
}
```

مثال 5: أكتب برنامج لإيجاد حاصل ضرب الاعداد الزوجية من 2-10

```
# include < iostream.h>
main( )
{
int i, product;
product=1;
i=2;
do
{
product * = i;
i + = 2;
}
while (i<=10);
cout<<" product = " << product;
}
```

انتهت المحاضرة

مراجع المحاضرة الثامنة:

1. بشير القايد ، اساليب البرمجة بلغة ++C ، منشورات ELGA ، 2005 .
2. حذيفة عبدالرحمن ، البرمجة بلغة ++C وتراكيب البيانات ، متاح في هذا الموقع <http://faculty.ksu.edu.sa> .
3. دروس كاملة للمبتدئين في أساسيات ++C ، متاح في هذا الموقع : موقع فيجوال سي للعرب.

المحاضرة التاسعة

الحلقات التكرارية (3)

9.1 جملة اذهب إلى go to

تستخدم هذه الجملة لتغيير مسار البرنامج التتبعي أو الخروج من الحلقات التكرارية مثل While أو For وغيرها وأحيانا الخروج نهائيا من البرنامج. والشكل العام لها :

go to lable;

.....

lable statement:

حيث lable اسم العنوان الذي يمكن وضعه في أي مكان من البرنامج الرئيسي أو الفرعي ، ويكون ذا اسم فريد ، وهذه الجملة قد تنفذ تحت شرط معين وتسمى if المشروطة وبدون شرط وتعرف بجملة if غير المشروطة.

ملاحظة: ينبغي التقليل من هذه الجملة في البرامج لأنها في بعض الأحيان تؤدي إلى جعل البرنامج غير مفهوم وغير واضح هيكلياً ، وخصوصاً وقت المراجعة والتكليف. مثال 1 : أكتب برنامج بلغة سي بلس بلس باستخدام جملة go to المشروطة يقوم بطباعة الاعداد من 1-10.

```
// goto example
#include<iostream.h>
main()
{
int n=10 ;
loop : cout<<n<<" ";
n-- ;
if(n>0) goto loop ;
}
```

سيقوم البرنامج بطباعة التالي : 10 , 9 , 8 , 7 , 6 , 5 , 4 , 3 , 2 , 1

مثال 1 : أكتب برنامج بلغة سي بلس بلس باستخدام جملة go to الغير مشروطة يقوم بطباعة ناتج ضرب عددين مع إضافة 1 للرقم الاول في كل مرة يتم فيها التنفيذ .

```
#include<iostream.h>
main()
{
int a , b , c ;
a=1 ;
b=2 ;
first : c=a*b ;
cout<<a<<"\t"<<b<<"\t"<<c<<"\n";
a++ ;
goto first ;
}
```

9 . 2 جملة اقطع Break

تستخدم هذه الجملة للخروج من الحلقات التكرارية المختلفة حيث يتم انهاء التكرار متى وصل التنفيذ إلى هذه الحلقة .

مثال 3: أكتب برنامج بلغة سي بلس بلس يقوم بطباعة الارقام تنازليا من 10 مع التوقف عندما يصل التنفيذ إلى الرقم 4 مستخدماً في ذلك جملة break .

```
// break loop example
#include<iostream.h>
main()
{
int n ;
for (n=10 ; n>0 ; n--)
{
cout<<n<<" , " ;
if (n==4)
{
cout<<"countdown aborted " ;
break ;
}
}
}
```

سيقوم البرنامج بطباعة التالي :

10 , 9 , 8 , 7 , 6 , 5 , 4
countdown aborted حيث يتوقف البرنامج عند الوصول للرقم 4

9 . 3 دالة الخروج exit

وهي تعني الخروج من البرنامج كلياً كما يدل اسمها ، وترجع بالقائمة صفراً إذا نفذ البرنامج ، وترجع بالقيمة غير الصفر إذا كان هناك بعض الأخطاء، ويستخدم ملف العناوين <stdlib.h> حتى يتمكن المترجم من التعرف على هذه الدالة.

مثال 4 : أكتب برنامج يقوم بطباعة مجموع 10 أعداد موجبة ، أخرج من البرنامج إذا كان العدد أصغر من أو يساوي صفر .

```
#include<iostream.h>
#include<stdlib.h>
main()
{
int i , number , postnumber ;
postnumber=0 ;
cout<<"please enter 10 values :\n";
for (i=1 ; i<=10 ; i++)
{
cout<<"enter value"<<i<<"==>";
cin>>number ;
if (number <=0)
{
cout<<"This is negative or zero number \n";
exit(0);
}
postnumber+=number ;
}
cout<<"The sum of positive " ;
cout<<"values are :"<<postnumber;
return 0;
}
```

البرنامج ينفذ في حالة ما كانت كل الأرقام المدخلة موجبة وفي حالة تم إدخال رقم سالب أو صفر يتم الخروج من البرنامج نهائياً دونما جمع حتى الأرقام الموجبة.

9 . 4 جملة الاستمرار continue

تعمل على عكس جملة الخروج exit وتعني الاستمرار في توجيه التحكم إلى نهاية الحلقة وبالتالي الرجوع إلى بداية الحلقة وإكمال تنفيذها .

مثال 5 : أكتب برنامج يقوم بطباعة مجموع 10 أعداد موجبة ، لا تخرج من البرنامج إذا كان العدد أصغر من أو يساوي صفر بل اجمع كل الأرقام ما عدا صفر والأرقام السالبة .

```
#include<iostream.h>
#include<stdlib.h>
main()
{
int i , number , postnumber ;
postnumber=0 ;
cout<<"please enter 10 values :\n";
for (i=1 ; i<=10 ; i++)
{
cout<<"enter value"<<i<<"==>";
cin>>number ;
if (number <=0)
{
cout<<"This is negative or zero number \n";
continue;
}
postnumber+=number ;
}
cout<<"The sum of positive " ;
cout<<"values are :"<<postnumber;
return 0;
}
```

يقبل البرنامج إدخال أرقام أيا كانت موجبة أو سالبة أو صفر ويعتبر هذه الأعداد ضمن الأعداد العشرة لكنه لا يقوم بحسابها أثناء الجمع .

أثناء التنفيذ وإدخال البيانات كالتالي :

Please type 10 values:

Enter value 1==> 3

Enter value 2==> 6

Enter value 3==> 10

Enter value 4==> 5

Enter value 5==> 0

This is negative or zero number

Enter value 6==> 2

Enter value 7==> 15

Enter value 8==> -7

This is negative or zero number

Enter value 9==> 11

Enter value 10==> 18

The sum of positive values are: 55

انتهت المحاضرة

مراجع المحاضرة التاسعة:

1. بشير القايد ، اساليب البرمجة بلغة ++C ، منشورات ELGA ، 2005 .
2. حذيفة عبدالرحمن ، البرمجة بلغة ++C وتراكيب البيانات ، متاح في هذا الموقع

. <http://faculty.ksu.edu.sa>

المحاضرة العاشرة

المصفوفات Arrays

10. 1 تعريف المصفوفة

هي عبارة عن منطقة في الذاكرة تتكون من عدد متجانس ومحدد من المواقع المتجاورة ، متجانسة يعني أن هذا الجزء من الذاكرة يستخدم لتمثيل نوع واحد من البيانات، ومحددة نعني بها أنها محددة في عددها برقم صحيح يستخدم لتحديد عدد المواقع المطلوبة في الذاكرة.

10. 2 المصفوفة ذات البعد الواحد

هو عبارة عن صف أو عمود يحتوي على مجموعة من عناصر البيانات متحدة النوع والاسم. والشكل العام للإعلان عنها :

Data Type Array Name [Index] ;

حيث :

Data Type هي نوع بيانات المصفوفة .

Array Name اسم المصفوفة ويراعى فيه شروط تسمية المتغيرات.

Index دليل المصفوفة وهي عبارة عن قيمة صحيحة تحدد عدد عناصر المصفوفة يمكن أن تكون ثابتة ويمكن أن تكون متغيرة .

أمثلة :

1. int x [50];

إعلان عن مصفوفة حجم بياناتها 50 من النوع الصحيح

2 . float y [20];

إعلان عن مصفوفة من النوع الحقيقي وحجم بياناتها 20

3. char name [15] ;

إعلان عن مصفوفة من النوع الحرفي وحجم بياناتها 15

الوصول إلى عناصر المصفوفة

للوصول إلى المواقع داخل المصفوفة ترقم المصفوفة بدءاً من الصفر وانتهاءً بالرقم الذي يسبق العدد الكلي لعناصر المصفوفة (n-1).

مثلاً : المصفوفة التالية مكونة من 10 عناصر واسمها C :

C[0]	6
C[1]	-5
C[2]	10
C[3]	25
C[4]	-17
C[5]	34
C[6]	12
C[7]	7
C[8]	51
C[9]	-19

مثال 1 : أكتب برنامج بلغة سي بلس بلس لقراءة مصفوفة من بعد واحد تتكون من 10 عناصر وطباعة مجموع هذه المصفوفة .

```
#include <iostream.h>
int main( )
{
int a[10];
int sum,i;
sum=0;
for(i=0;i<10;i--)
{
cin>>a[i];
sum +=a[i];
}
cout<<"sum="<<sum<<"\n";
return 0;
}
```

مثال 2 : أكتب برنامج بلغة سي بلس بلس لحساب متوسط 5 قيم مدخلة من النوع الحقيقي.

```
include<iostream.h>
int main( )
{
float x[5] , sum=0 , avg;
int i;
for(i=0;i<5;i++)
{
cout<<"enter elemant["<<i+1<<"] \n";
cin>>x[i];
sum+=x[i];
}
avg=sum/5;
cout<<"the average="<<avg<<"\n";
return 0;
}
```

10 . 3 المصفوفة متعددة الأبعاد

وعادة ما تتكون من عدة أبعاد وغالباً ما يتم التعامل مع المصفوفة ذات البعدين وذلك لقلة التعامل مع المصفوفات متعددة الأبعاد.

وهي المصفوفة التي تتكون من عدد من الصفوف Rows والاعمدة Columns وفيها يتم إعطاء المصفوفة دليلين $m*n$ حيث n تمثل الصفوف و m تمثل الأعمدة.

والشكل العام لتعريف هذا النوع من المصفوفات :

data type array name [row size] [column size];

حيث :

data type نوع بيانات المصفوفة .

array name اسم المصفوفة .

row size عدد الصفوف.

Column size عدد الأعمدة.

مثلاً :

1. int y[4][3]

الاعلان عن مصفوفة من النوع الصحيح تتكون من 4 صفوف وثلاثة أعمدة.

2. int y[4][3]={{5,0,-4},{-2,3,1},{4,7,6},{9,8,-1}};

إسناد قيم ابتدائية لعناصر المصفوفة y أثناء التصريح وعناصرها كما يلي :

العمود الاول

5	0	-4
-2	3	1
4	7	6
9	8	-1

الصف الاول

حيث يتم تجميع عناصر كل صف ضمن قوسين .

مثال 3: أكتب برنامج يقوم بقراءة مصفوفة ذات بعدين مع طباعة البيانات المدخلة على هيئة المصفوفة الثنائية .

```
#include<iostream>
const int row=3;
const int col=4;
int main( )
{
int a[row][col],i,j;
cout<<"enter the elements of array:\n";
for(i=0;i<row;i++)
{
for(j=0;j<col;j++)
{
cout<<"a["<<i<<","<<j<<"]="";
cin>>a[i][j];
}
}
cout<<"the array looks like:\n";
for(i=0;i<row;i++)
```

```

{
for(j=0;j<col;j++)
cout<<a[i][j]<<"\t";
cout<<"\n";
}
return 0;
}

```

مثال 4: أكتب برنامج يقوم بجمع مصفوفتين صحيحتين من النوع 3×3 لأعداد يتم إدخالها من قبل المستخدم ثم طباعة المصفوفة على هيئة المصفوفة الثنائية .

```

#include <iostream.h>
int main( )
{
int a[3][3],b[3][3],c[3][3],i,j;
cout<<"first array a[3][3]:\n";
for(i=0;i<3;i++)
{
for(j=0;j<3;j++)
cin>>a[i][j];
cout<<"\n";
}
cout<<"second array b[3][3]:\n";
for(i=0;i<3;i++)
{
for(j=0;j<3;j++)
cin>>b[i][j];
cout<<"\n";
}
cout<<"the first array a[3][3] look like :\n";
for(i=0;i<3;i++)
{
for(j=0;j<3;j++)
cout<<a[i][j]<<"\t";

```

```
cout<<"\n";
}
cout<<"the second array b[3][3] look like :\n";
for(i=0;i<3;i++)
{
for(j=0;j<3;j++)
cout<<b[i][j]<<"\t";
cout<<"\n";
}
cout<<"the sum of tow array c[3][3] look like:\n";
for(i=0;i<3;i++)
{
for(j=0;j<3;j++)
{
c[i][j]=a[i][j]+b[i][j];
cout<<c[i][j]<<"\t";
}
cout<<"\n";
return 0;
}
```

انتهت المحاضرة

مراجع المحاضرة العاشرة:

1. حذيفة عبدالرحمن ، البرمجة بلغة ++C وتراكيب البيانات ، متاح في هذا الموقع

<http://faculty.ksu.edu.sa>