المحاضرة العاشرة

الدوال Functions

البرامج السابقة كتبت على أنها وحدة واحده وهذا قد يكون غير مناسب في بعض الأحوال حيث يصعب متابعة وإصلاح البرنامج خصوصا عندما يكون البرنامج طويلاً, ولذلك من الأفضل تجزئة البرنامج إلى أجزاء صغيره كل جزء يؤدي مهمة معينة ثم تختبر هذه الأجزاء وترتبط مع بعضها لتكون البرنامج الكامل.

أيضا في بعض الأحيان يتحتم عليك إعادة تنفيذ عدد من الجمل المتكررة في البرنامج الواحد .وعليه يمكن تحويل الجمل المتكررة إلى دوال.

تعريف الدالة

عبارة عن برنامج فرعي يتم تعريفه قبل الدالة main() ويتم استدعاؤها داخل الدالة main() ليقوم بوظيفة معينة

الفائدة من استخدام الدوال

- سهولة متابعة وإصلاح البرنامج .
- المساعدة في فهم البرنامج.
- عدم تكرار الجمل المراد إعادة تنفيذها .

الصيغة العامة للدالة

```
data type      function name (par1,par2,….)
{
   local variables declaration ;
     function body;
   return(expression);
}
```

حيث:

1. data type: نوع قيمة الدالة (نوع مردود الدالة)
2. function name : اسم الدالة
3. (par1,par2…) : معاملات أو وسائط الدالة
4. local variables declaration: المتغيرات المعلن عنها في الدالة وهي محليه أي تستخدم بالدالة فقط.
5. function body : جسم الدالة(التعليمات)
6. return: ترجع قيمة التعبير expression إلى مكان استدعاؤها.

استخدام الدالة

عند استخدام أي داله يجب القيام بالاتي:

1. الإعلان عن الدالة Function Declaration

ويكون قبل الدالة الرئيسية main() ويشمل الأتي :

أ – نوع مردود الدالة.

ب –اسم الدالة.

ج –أنواع المعاملات (الوسائط) التي تستقبلها الدالة.

☑	ينتهي الإعلان بفاصله منقوطة.
☑	الدالة التي ليس لها مردود (الدالة الفارغة) يكون نوع بياناتها void

أمثله على بعض الإعلانات للدوال:

```
1-void display();
2-int example();
3- int mul (int ,int);
4- float div (int,int,int);
5-float average( int[],int);
6-void square_number(int);
```

2. تفاصيل الدالة Function Details

ويتم كتابة تفاصيل الدالة أسفل البرنامج الرئيسي (الدالة main()) وتحتوي التفاصيل على الأتي:

أ – نوع مردود الدالة.

ب –اسم الدالة .

ج –أنواع وأسماء الوسائط التي تستقبلها الدالة .

د –جسم الدالة (التعليمات).

مثال:

إذا أعلنا عن الدالة بالصورة التالية:

```
int addition (int,int);
```

فيمكن أن تكون التفاصيل كما يلي :

```
int addition (int a,int b)
{
    int r;
    r=a+b;
   return  r;
}
```

3. استدعاء الدالة Function Calling

يتم استدعاء الدالة داخل البرنامج الرئيسي (الدالة main()) عن طريق اسمها مع مراعاة نوع وعدد الوسائط التي تستقبلها .

مثال:

الدالة التي كتبت تفاصيلها أعلاه يمكن استدعاؤها بالصور الثلاث التالية :

1. addition(5,7); استدعاء بقيم محددة
2. addition (x,y); استدعاء بقيم مدخلة من قبل المستخدم
3. Z= addition(x,y); استدعاء الدالة وإسناد الناتج للمتغير z

أمثله توضح استخدام الدوال في البرنامج:

1. أكتب برنامج يحتوي على داله تقوم بطباعة الجملة : Welcome to C++ language على الشاشة.

```
#include<iostream.h>
void display();
int main()
{
  display();
  return 0;
}
void display()
{
cout<<"welcome to  C++ language \n";
}
```

2. أكتب برنامج يحوي داله تقوم بحساب مكعب أي قيمة صحيحة مدخلة من قبل المستخدم ؟

```
#include<iostream.h>
int cube(int n)
{
 return(n*n*n);
}
 int main()
{
  int x;
  cout<<"enter x \n";
  cin>>x;
  cout<<"x^3="<<cube(x)<<endl;
  return 0;
}
```

☑	يمكن الاستغناء عن الإعلان عن الدالة عندما تكتب تفاصيلها قبل البرنامج الرئيسي (الدالة main()).

3. أكتب برنامج لحساب متوسط n من القيم الصحيحة المدخلة من قبل المستخدم على أن يتم حساب المتوسط في دالة مستقلة؟

```
#include<iostream.h>
float average (int);
int main()
{
int n;
float avg;
cout<<"enter size of the numbers :\n";
cin>>n;
avg=  average(n);
cout<<"the average of all numbers ="<<avg<<endl;
return 0;
}
float average(int a )
{
  int i, value;


float sum=0;
cout<<"enter "<<a<<"values please=:"<<endl;
for(i=1;i<=a;i++)
{
  cout<<"enter value "<<i<<"\n";
  cin>>value;
  sum+=value;
 }
return sum/a;
}
```

الدوال ذاتية الاستدعاء Recursion Functions

يمكن للدوال أن تستدعي دوال أخرى كما يمكنها أيضا أن تستدعي نفسها (بمعاملات أخرى مثلاً) ولكن حتى نضمن أنها تعمل يجب أن نوفر حالة مؤكدة الحدوث بأن عملية الاستدعاء هذه ستتوقف عند مرحلة معينة قبل أن يحدث طفح overflow وإذا أمكن الاستغناء عن هذه الطريقة باستعمال الحلقات التكرارية فهذا أفضل لأن الأخيرة أسرع ولا تستترف الذاكرة .

مثال1:

برنامج يقوم بحساب مضروب عدد من النوع الصحيح باستخدام الاستدعاء الذاتي للدوال(بدون حلقة تكرارية)

حيث مضروب العدد n هو

$$n! = n(n-1)(n-2)(n-3).....3.2.1$$

فمثلاُ مضروب 5 هو 5.4.3.2.1

ومضروب 4 هو 4.3.2.1

حيث يمكن حساب مضروب 5 بشكل مختصر كما يلي 5!=5.4! (أي 5 ضرب مضروب 4)

وبشكل عام :- مضروب أي عدد يساوي العدد ضرب مضروب العدد الذي قبله

أي أن n!=n.(n-1)! علماً بان مضروب 0 و1 هو 1

```
#include<iostream.h>
int f(int n)
{
 if (n==0 || n==1)
 return 1;
else
 return n*f(n-1);
}
int  main( )
{
 int n;
cout<<"enter n:\n";
 cin>>n;
cout<<"factorial "<<n<<"is "<<f(n)<<endl;
return 0;
}
```

تدريب:

1. اكتب برنامج يقوم بحساب قيمة المعادلة

$$sum=x^2+x^4+x^6+\ldots+x^n$$

على أن تتم عملية حساب المعادلة في دالة حيث x قيمة صحيحة مدخلة من قبل المستخدم

2. صف مخرجات البرنامج التالي

```
#include<iostream.h>
int fun( );
int main( )
{
int i;
i=9;
while(i<=14)
{
cout<<fun()<<"\n";
++i;
}
return 0;
}
int fun( )
{
int i=6;
int j=10;
return(i++ + ++j);
}
```

3. اكتب برنامجاً محتوياً على دالة وظيفتها حساب المجموع التالي sum ثم طباعته على ا لشاشة حيث

$$sum = \frac{2}{3} + \frac{3}{5} + \frac{4}{7} + ... + \frac{n+1}{2n+1}$$