برمجة الكائنات

(Object-Oriented Programming)

الـــــدوال

المحاضرة الثانية

```
addition ();
}
```

2- النوع الثاني من كتابه الدوال .

وهذا النوع من الدوال يستقبل قيم ولا يرجع اي قيمة .

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

حيث :

function-name:اسم الدالة والذي يتبع في تسميته قواعد تسمية المعرفات .

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

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

ex1.

```
#include <iostream>
addition (int a)
{
 int r;
```

```
 r=a*2;
cout << r;
}
 main ()
{
 addition (٥);
}
```

ex2

```
#include <iostream>
addition (int a)
{
 int r;
 r=a+٤;
 cout << r;
}
 main ()
{
 int x;
 cout << ""x";
 cin >> x;
 addition (x);
}
```

ex3

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

```
{
 int x,y;
 cout << ""x,y";
 cin >> x >>y;
 addition (x,y);
 sub(x,y);
 Muilt(x,y);
 Div(x,y);

}


addition (int a,int b)
{
 int r;
 r=a+b;
cout <<"the sum = " << r;
}
sub (int a, int b)
{
 int r;
 r=a-b;
 cout <<"the sub = " << r;
}

Muilt (int a, int b)
```

```
{
 int r;
 r=a*b;
 cout <<"the Muilt = " << r;
}

Div (int a, int b)
{
 int r;
 r=a/b;
 cout <<"the Div = " << r;
}
```

```
# include <iostream>

    main() {
    ... .. ...
          add(num1, num2);  // Actual parameters: num1 and num2
    ... .. ...
}

    add(int a, int b) {     // Formal parameters: a and b
    ... .. ...
     add = a+b;
    ... .. ...
}
```

الشكل يوضح كيفية استدعاء القيم واستخدمها داخل الداله الثانوية