Local and Global variables:

The variables in general bay be classified as local or global variables.

(a) Local variables: Identifiers, variables and functions in a block are said to belong to a particular block or function and these identifiers are known

as the local parameters or variables. Local variables are defined inside a function block or a compound statement. For example,

```
Void func (int I, int j)
{
Int k,m;          // local variables
……                // bodyofthe function
}
```

Local variables are referred only the particular part of a block or a function. Same variable name may be given to different parts of a function or a block and each variable will be treated as a different entity.

(b) Global variables: these are variables defined outside the main function block. These variables are referred by the same data type and by the same name through out the program in both the calling portion of the program and in the function block.

EX// write a program that uses two functions, one to find the area of the rectangle and the other to find the perimeter of the rectangle.

```
#include<iostream .h>
int rec1( )
{int x, y ,t;
  x=4;y=6 ;
  t=x*y ;
 cout<< t
return 0;
}
int rec2( )
{ int x,y,t ;
```

```
x=4 ; y=6;
 t=(x+y)*2;
cout<<t;
return 0;
}
main()
{ rec1();
  rec2();
return 0;
}
```

نلاحظ ان المتغيرات (x,y,t) المستخدمة في الدا لة االاولى والثانية محلية اي خاصة بتللك الدالة اي ان x,y,t في الدالة الثانية هي ليست نفسها في الدالة الاولى لذلك ادخلناها مرة اخرى في الدالة الثانية لان المتغيرات المحلية بمجرد انتهاء تنفيذ الدالة تلغى القيم بداخلها 0

```
int x,y
int rec1()
  {int t;
x=4 ; y=6;
t=x* y;
cout<<t;
return0;
}
int rec2()
 {int t ;
   t=(x+y)*2 ;
cout<< t;
return 0;
}
main()
 { rec1();
   rec2() ;
  return 0;
}
```

هي متغيرات عامة فعرفت خارج حدود اي دالة فتكون x,y معروفة في كل البرنامج فلا نحتاج الى تعريفها مرة اخرى عندما نستخدمها فى اى دالة وتكون محتفضة بقيمها الحاليه

لانها متغيرات 6 هي y وقيمة ال 4 هيxعند الخروج من الدالة تبقى قيمة ال فعندما نستخدمها في اي مكان اخر في البرنامج تبقى عامة تحتفظ بقيمها النهائية على قيمتها التي تحتويها

y و ال x لانها نفس ال 6 هنا ايضا قيمته y وال 4 هنا قيمتها ايضا xال المستخدمة في الدالة الاولى لذلك لم نقم بادخالها مرة اخرة لانها متغيرات عامة تبقى محتفظة بقيمها النهائية

***برنامج اخر يبين استخدام دالة لايجاد مساحة دائرة حيث نقوم بادخال نصف قطر الدائرة بالبرنامج الرئيسي ونستخدمه نفسه في الدالة بدون تعريفه مرة اخرى او ادخال قيمته مره اخرى0 بما ان نصف القطر لا نريد تعريفة مره اخرى في اي جزء من البرنامج ويكون محتفظ بقيمته اينما يستخدم يجب ان يعرف كعام (Global) اي يكون البرنامج كالاتي 0
قانون مساحة الدائرة هو النسبة الثابتة في نصف القطرتربيع .

```
int rad ;
int areacricle( )
 float area;
area=3.14*red *red ;
cout<< "area="<< area <<"\n" ;
return 0
}
main()
{rad=10;
 areacricle()
return 0;
}
```

ولا نحتاج الى ادخال قيمته 10 هنا قيمتة redال مرة اخر او الى تعريفه لانه معرف في البداية

لا نحتاج الى تعريفه لانه معرف في البداية

EX: Writ a program to enter array A[5] in main program and then use two functions .the first function find the minimum number in it and the second function find the maximum number in same array .and print the minimum and maximum in main program .

```
#include<iostream.h>
int A[5];
int minfun()
{int  i ,min ;
 for(i=0 ; i<5 ; i++)
      if( i==1)
         min=A[i];
    else
   if (A[i]< min )
       min=A[i];
return min ;
}
int maxfun()
{int i, max;
 for(i=0 ; i<5 ; i++)
      if( i==1)
```

for هي جملة واحدة داخل ال ifجملة ال

```
max=A[i];
    else
   if (A[i]> max )
        max=A[i];
return max ;
}
main ()
{int I  ;
 for(i=0; i< 5 ; i++)
  {cout<<"enter the number"<<endl;
   cin>> A[i] ;
}
minfun();
maxfun();
cout<<"the minimum no ="<<mimfun() <<"\n" ;
cout<<" the maximum no ="<<maxfun() <<"\n" ;
return 0 ;
}
```

for هي جملة واحدة داخل ال ifجملة ال

نلاحظ ان المصفوفة تم تعريفها في البداية لذلك هي اصبحت عامة اي يمكن استخدامها في اي جزء في البرنامج بدون تعريفها مرة اخرى او ادخالها مرة اخرى لانها تبقى محتفظة بقيمها الحالية اثناء تنفيذ البرنامج فعندما ادخلنا المصفوفة في البرنامج الرئيسي واستدعينا الدالة الاولى فان الدالة الاولى وكذلك الثانية سوف تتعرف على هذه المصفوفة ومع قيمها لذلك لا نحتاج الى تعريف المصفوفة مرة ثانية او ادخال قيمها مرة اخرى في الدالتين لانها عامة

EX: Write a program to enter array a[5] in main program and then use function to square the element in it and print the array in main program

```
#include<iostream.h>
 #include<math.h>
int a[5];
 int powarray( )
{ int i ;
 for(i=0 ;i< 5 ;i++)
  a[i]= pow(a[i],2) ;
return 0;
}
main()
{int i, a[5];
for(i=0 ; i<5 ; i++)
  cin>> a[i];
powarray( );
for(i=0 ;i<5 ; i++)
  cout<<a[i];
```

هي مصفوفة return وذللك لان القيمة بعد ال return(a[])هنا لا يمكن ان نقول لذلك لايمكن اخراج مصفوفة لانه intونوع القيمة التي تنتجها الدالة هي يتعارض مع نوع القيمة الموجودة في راس الدالة وهذا يحدث فقط مع المصفوفة لذلك البرنامج الرئيسي سوف يعرف المصفوفة بعد تربيع عناصرها وبدون الحاجة) اوحتى اذ لم تكن معرفة بشكل عام global (الى اخراجها لانها معرفة بشكل عام ايضا يتعرف عليها كما ذكرنا سابقا

طباعة المصفوفة التي تغيرت قيمها داخل الدالة وهنا سوف يعرفها البرنامج بدون returnاخراجها ب ال

```
return 0;
}
```

EX: Write a program to enter array a[5] in main program and then use two functions, the first function use to square the element in it and the second use to print the result array in it.

هنا سوف نقوم بادخال المصفوفة في البرنامج الرئيسي ثم نستدعي الدالة الاولى لكي تقوم بتربيع كل عنصر قي المصفوفة ثم تقوم الدالة الثانية بطباعة المصفوفة الناتجة بعد التربيع لذلك يجب ان تكون المصفوفة معروفة ومحتفظة بقيمها الجديده لكي تسطيع الدالة الثانية التعرف عليها وطباعتة قيمها المربعة الجديدةا لذا يجب ان تعرف كمصفوفة عامة (globle)

```
#include<iostream.h>
 #include<math.h>
 double  a[5];
void  power()
  { int  i;
    for(i=0 ; i< 5; i++)
      a[i]=pow(a[i] ,2);
  }
void printarray ();
{int  i;
 for(i=0 ; i<5 ; i++)
   cout<<a[i ]<<"\n";
 return 0;
}
main()
{ int  i;
for(i=0 ; i< 5; i++)
  cin>> a[i] ;
power() ;
printarray ();
return 0;
}
```

هنا لم نقم بادخال المصفوفة في كلمة الاستدعاء لان المصفوفة ستكون معرفة ومحتفظة بقيمها اينما تستخدم بالبرناج لانها عامة لذلك الدالتين سوف تعرفها بدون الحاجة الى ادخالها لهما