المحاضرة الخامسة

الدوال الصديقة والأصناف الصديقة

Friends Functions and Friend Classes

الدوال الصديقة Friend Functions

تعريف الدالة الصديقة:-

هي عبارة عن دالة لاتنتمي لصنف معين ولكننا نحتاج إليها للوصول إلى خصائص ذلك الصنف

أو هي دالة معرفة خارج الصنف مع امتيازها بالوصول إلى أعضاء الصنف

مثال توضيحي:-

```
class ff
{
private:
int x;
public:
ff( )
{
x=0;
}
void set(int y)
{
x=y;
}
};
```

إذا كتبنا قبل البرنامج الرئيسي مايلي:-

```
void plus(ff   f)
{
   //  ff  كائن من نوع الصنف  f  ;
   cout<<f.x+5;
  //    (متغير ) خاصية x ;
}
```

فهذه الطريقة خاطئة لأنه لا يمكن الوصول إلى المتغير **x** وذلك لان الدالة **plus** ليست عضو في الصنف **ff**

وحتى تتمكن الدالة **plus** من الوصول إلى خصائص الصنف يجب الإعلان عنها كدالة صديقة داخل الصنف **ff**

ويتم الإعلان عنها داخل الصنف باستخدام الكلمة المحجوزة **friend** كالتالي:-

```
class ff
{
"""""
"""""
"""""
"""""
friend    void plus(ff )
// إعلان فقط دون كتابة تفاصيل
// لا تكتب تفاصيل الدالة الصديقة داخل الصنف
};
```

كيفية استدعاء الدالة الصديقة:-

```
int main( )
{
   // اشتقاق كائن من نوع الصنف
   ff    f1;
   // استدعاء دالة عضو في الصنف
   f1.set( 50 );
   // استدعاء الدالة الصديقة ووسيطة
   // الاستدعاء عبارة عن كائن من نوع الصنف
     plus(f1)
```

☑	1. الدالة الصديقة لها امتيازات الدالة العضو أي لها القدرة على الوصول إلى خصائص الصنف 2. الدالة الصديقة يتم تعريفها داخل الصنف باستخدام الكلمة المحجوزة **friend** 3. يتم استدعاء الدالة الصديقة داخل البرنامج الرئيسي عن طريق اسمها والوسيطة (الوسيطة عبارة عن كائن من نفس الصنف) 4. لا يستخدم المؤشر **This** مع الدالة الصديقة لأننا أصلاً لن نستدعي الدالة عن طريق الكائن 5. لا تعتبر الدالة الصديقة عضو من أعضاء الصنف ولذلك من الخطأ استدعاؤها باستخدام اسم الكائن وأداة الوصول (.)

وألان نكتب البرنامج بصورة متكاملة:-

```
#include<iostream.h>
class ff
{
private:
int x;
public:
ff()
{
x=0;
}
void set(int y)
{
x=y;
}
friend void plus(ff);
} ;
void plus(ff   f)
{
cout<<f.x+5;
}
int main(  )
{
ff  f1;
f1.set(50);
plus(f1);
return 0;
}
```

مثال آخر:-

برنامج يقوم بمقارنة نصفي قطري دائرتين باستخدام الدالة الصديقة

```
#include<iostream.h>
class circle
{
private:
float ar,c;
public:
int radius;
circle(  )
{
radius=0;
}
circle(int r)
{
```

```
radius=r;
}
void area( );
void cir( );
void display( );
friend compare(circle,circle);
};
void circle::area( )
{
ar=radius*radius*3.14;
}
void circle::cir( )
{
c=2*radius*3.14;
}
void circle::display( )
{
cout<<"area ="<<ar<<"\n";
cout<<"cir  ="<<c<<"\n";
}
compare(circle c1,circle c2)
{
return(c1.radius>c2.radius);
}
int main( )
{
int r1, r2;
cout<<"enter radius 1\n";
cin>>r1;
cout<<"enter radius 2\n";
cin>>r2;
circle c1(r1),c2(r2);
c1.area( );
c1.cir( );
c2.area( );
c2.cir( );
c1.display( );
c2.display( );
if(compare(c1,c2))
cout<<c1.radius<<"is greater";
else
cout<<c2.radius<<"is greater";
return 0;
}
```

مثال آخر عن الدالة الصديقة :-

```
// friend function
#include <iostream.h>
 class crectangle
 {
    int width, height;
  public:
    void set_values (int, int);
    int area (void)
   {
     return (width * height);
   }
    friend crectangle duplicate (crectangle);
};
void crectangle::set_values (int a, int b)
{
  width = a;
  height = b;
}
crectangle duplicate (crectangle rectparam)
{
  crectangle rectres;
  rectres.width = rectparam.width*2;
  rectres.height = rectparam.height*2;
  return (rectres);
}
int main( )
{
  crectangle rect, rectb;
  rect.set_values (2,3);
  rectb = duplicate (rect);
  cout << rectb.area( );
  return 0;
}
```

خلاصة القول:-

1. الدالة الصديقة لها نفس امتياز الدالة العضو حيث يتم تعريفها داخل الصنف باستخدام الكلمة المحجوزة **friend** ويتم استدعاؤها بالطريقة التالية:-

Function name(Object name);

2. تعرف الدالة الصديقة في أي مكان داخل الصنف

الأصناف الصديقة Friend Classes

تعريف الصنف الصديق:-

هو عبارة عن صنف تستطيع كل دواله الأعضاء أن تستخدم خصائص الصنف الآخر

مثلاً:-

إذا كان لدينا الصنف **x** صديق لصنف آخر وهو **y** هذا يعني أن كل الدوال الأعضاء في **x** لها ميزة الوصول لخصائص الصنف **y** وليس العكس

☑	للإعلان عن الصنف الصديق نستخدم الكلمة المحجوزة **friend**

مثال توضيحي:-

```
class a
class b
{
friend class a;
"""""
"""""
};
class a
{
"""""
"""""
};
```

لاحظ أن **a** صديق لـ **b**

ويمكن أن يكتب بصورة أخري:-

طالما أن **a** صديق يمكن أن يكتب أولاً بكل تفاصيله بدلاً عن الإعلان فقط كما في الحالة الأولى ويكون ذلك كما يلي:-

```
class a
{
"""""
"""""
};
class b
{
friend class a;
"""""
"""""
};
```

مثال:-

برنامج يقوم بإيجاد الوسط الحسابي لـ **n** من الأعداد الصحيحة المدخلة من قبل المستخدم

```
#include<iostream.h>
class mean;
class summation
{
private:
float n,x,sum;
public:
summation( )
{
sum=0;
cout<<"enter n \n";
cin>>n;
for(int i=1;i<=n;i++)
{
cout<<"enter x"<<i<<"\n";
cin>>x;
sum+=x;
}
}
friend class mean;
};
class mean
{
private:
float s;
public:
mean(summation ss)
{
s=ss.sum/ss.n;
cout<<"mean ="<<s<<"\n";
}
};
int main( )
{
summation s;
mean m(s);
return 0;
}
```

مثال آخر عن الأصناف الصديقة:-

```
// friend classes
#include <iostream.h>
#include <conio.h>
class csquare;
class crectangle
 {
   int width, height;
  public:
    int area (  )
      {
      return (width * height);
      }
    void convert (csquare a);
};
class csquare
 {
  private:
    int side;
  public:
    void set_side (int a)
      {
      side=a;
      }
    friend class crectangle;
};
void crectangle::convert (csquare a)
{
  width = a.side;
  height = a.side;
}
int main(  )
 {
  csquare sqr;
  crectangle rect;
  sqr.set_side(4);
  rect.convert(sqr);
  cout << rect.area(  );
  return 0;
}
```