البرمجة بلغة c#

إعداد: خالد أبوزيد

برمجة غرضية ال توجه oop:(object oriented programing)

البنية الأساسية لأي برنامج غرضي التوجه هي ال class
بنية ال classعلى الشكل:

Class name class
{// begin class
أعضاء بيانية
أعضاء دالية
}//end class

هذا الشكل العام للكلاس حيث تمثل الأعضاء البيانية (data member) البطاقة الشخصية للكائن الذي سيتم إنشائه
مثال الأعضاء البيانية للشخص هي بطاقته الشخصية تحوي الاسم واسم الاب والميلاد ومكان الولادة ولا وجود لشخص دون بطاقة شخصية
كذلك الكائن دون الأعضاء البيانية ليس كائن
تكتب الاعضاء البيانية داخل الكلاس على هذا الشكل

class اسم الكلاس
{ بدية الكلاس
data member: اسم البيان نمط المعطيات محدد الوصول
} نهاية الكلاس

الأعضاء الدالية(intimacy member) تمثل صلة الوصل بين الكائن المنشئ و بين الأعضاء البيانية لأن الأعضاء البيانية في أغلب الأحيان يكون محدد الوصول لها private ولا يمكن الوصول له إلا من أعضاء الصف نفسه أي الأعضاء الدالية تستطيع الوصول للبيانات حيث تكون الأعضاء الدالية ذات محدد وصول public أي يمكن الوصول لها من خارج الكلاس .
حيث لدينا محددات الوصول الاتية:

المحدد	خاص private	عام public	محمي protracted
خواصه	لا يمكن الوصول لبياناته الا من داخل الصف نفسه .	يمكن الصول لبياناته من داخل وخارج الصف .	سنتناوله في درس الوراثة .

تكتب الاعضاء الدالية داخل الكلاس على هذا الشكل

class اسم الكلاس
{ بدية الكلاس
data member : اسم البيان نمط المعطيات محدد الوصول
{ begin } end
intimacy member : محدد الوصول نمط الاعادة اسم الدالة()
{ begin } end
} نهاية الكلاس

حيث نمط الاعادة اما يكون void لا يعيد قيمة واما نمط معطيات int-string-double-bool حيث تقوم بإرجاع قيمة .

مثال سوف نبني كلاس للوقت حيث يتألف من الساعات والدقائق

```
Class time
{
عضو بياني Private int hours;
عضو بياني Private int minutes;
عضو دالي Public set (int h,int m)
بداية للعضو الدالي{
This.houers=h;
This.minutes=m;
نهاية للعضو الدالي}
Public void print()
{console.writeLine(hours+":"+minutes);}
}//end class
```

استدعاء الكلاس وأعضائه ضمن main

```
Static void Main(string[]args)
{
```

للتصريح عن كائن نستدعي اسم الكلاس ثم اسم الكائن time x;
لإنشاء الكائن اسمه نسنده لكلاس جديد X=new time ();
وممكن دمج الخطوتين ب: time x=new time();

```
x.set(2,13);
x.print();
}//end main
```

** ملف البرنامج يحوي على الاقل كلاس وحيد هو كلاس ال main **
بناء كلاس employee يحوي بيانات موظف
١- الأعضاء البيانية:

1-first name
2-last name
3-salary

٢-الأعضاء الدالة:
1-initإسناد قيم الوسطاء للأعضاء البيانة
2-Printطباعة الأعضاء البيانية
3-full nameدالة تعود بالاسم الكامل للموظف
4-Salaryاعادة قيمة وقراءة قيمة من لوحة المفاتيح
ولها ثلث خواص: تبدأ بحرف كبير وتملك نفس اسم العضو البياني ولا تملك أقواس الدالة()
وممكن تطبيق دالة مثلها لكل عضو بياني كما هنا :

```
namespace ConsoleApplication1
{
    class employee
    {
        private string firstname;
        private string lastname;
        private int salary;
        public void init(string firstname, string lastname, int salary)
        {
            this.firstname = firstname;
            this.lastname = lastname;
            this.salary = salary;
        }//end init
        public void print()
        {
            Console.WriteLine(firstname + lastname + ":" + salary);
        }//end print
        public string fullname()
        {
            return firstname + lastname;
        }//end fullname
        public int Salary
        {
            get { return this.salary;}
            set {this.salary=value;}
        }//end salary

         public string Firstname
        {
         get {return this.firstname;}
         set {this.firstname =value;}
         }//end firstname
           public string Lastname
        {
          get { return this.lastname;}
```

```
            set {this.lastname=value;}
        }//end lastname
        }//end class employee


    class Program
    {
        static void Main(string[] args)
        {
            employee x = new employee();
            x . init("khaled", " abozaib", 20000);
                    x.print();//    طباعة الاسم مع الراتب                -
            Console.WriteLine(x.fullname());//طباعة الاسم
                  Console.WriteLine(x.Salary);//طباعة القيمة الثابتة   -
            string thes = Console.ReadLine();
            x.Salary = Convert.ToInt32(thes);
            x.print();
            Console.WriteLine("input firstname");
            x.Firstname=Console.ReadLine();
            Console.WriteLine("input lastname");
            x.Lastname=Console.ReadLine();
            x.print();//تنفيذ print
            Console.ReadLine();

        }
    }
}
```

المشيدات

Constructors

المشيد :هو عضو دالي خاص يملك نفس اسم الصف ولا يملك نمط إعادة ولا حتى من النوع void يعمل على حجز مساحة ذاكرة من أجل إنشاء الكائن وتمهيده عند القيم الابتدائية:

مثال:

Class employee

(١) الأعضاء البيانية:١- name

٢- age

٣- count

حيث (name && age تمثل أعضاء بيانية مثيلة)

أما (count هي عضو بياني لكن ليس لها تأثر بالكائنات المنشئة لكنها تمثل عدد الكائنات المنشة حيث count متعلق بالفئة وليس بالكائن فهو عضو بياني ساكن static member).

(٢)- الأعضاء الدالية:

١- مشيدات ولها ثلاثة أنواع:

a- مشيد افتراضي (لا يملك وسطاء)وظيفته إعطاء قيم ابتدائية للكائن .

مشيد يملك وسطاء(بنفس عدد الأعضاء البينية المثيلة)وظيفته إسناد قيم للأعضاء البيانية من داخل دالة ال main.

c- مبدأ النسخ ووسيطه كائن حيث ينسخ معلومات كائن (أعضائه بيانية) ويعطيها لكائن اخر .

٢- Count: دالة تعود بعدد الكائنات المنشئة .
٣- Name: دالة تقرأ اسم من لوحة المفاتيح وتعود بالاسم .

```
namespace ConsoleApplication1
{
    class employee
    {
        private string name;
        private int age;
        public static int count = 0;
        public employee()//مشيد إفتراضي
        {

            count++;
            Console.WriteLine("count="+count);
        }
        public employee(string name, int age)// مشيد وسطاء
        {
            this.name = name;
            this.age = age;
        }
        public employee(employee z)//مشيد النسخ
        {
            this.name = z.name;
            this.age = z.age;
        }
        public int Count
        {
            get { return this.Count; }
        }
        public string Name
        {
            get { return this.name; }
            set { this.name = value; }
        }
        public void print()
        {
            Console.WriteLine("name:"+name+"/age:"+age);
        }
    }//end class enployee
    class Program
    {
        static void Main(string[] args)
        {
            employee a = new employee();
            employee a1 = new employee("khaled",19);
            a1.print();
            Console.WriteLine("enter name");
            string c = Console.ReadLine();
            Console.WriteLine("enter age");
            int c1 = Int32.Parse(Console.ReadLine());
            employee a2 = new employee(c, c1);
            a2.print();
                employee a3 = new employee(a2);// استدعاء مشيد النسخ     -
```

```
            a3.print();
            Console.ReadLine();

        }
    }
}
```

سـؤال :

- اكتب صف لفئة كتاب يحوي على الأعضاء البيانية التالية :

١- اسم الكتاب ٣- سعره ٥- عدد الموجود منه

٢- رقمه ٤- اسم المؤلف ٦- count

ويحوي على الدوال الأتية :

١- مشيد افتراضي ومشيد الوسطاء .

٢- خصائص get && set

٣- دالة للبحث عن كتاب بحسب اسمه .

٤- دالة لبيع كتاب عن طريق رقم الكتاب وتعيد هذه الدالة كمية الكتب المتبقية .

الحل:

```
namespace ConsoleApplication1
{
    class book
    {
        private int bookid;//رقم الكتاب
        private string bookname;//إسم الكتاب
        private string publishername;//المؤلف
        private int price;//السعر
        private int amount;//الكمية
        public static int num = 0;//النوع من الكتب عدد
        public book()
        {
            num++;
            Console.WriteLine("count=" + num++);

        }
        public book(int bookid, string bookname, string publishername,
int price, int amount)
        {
           this.bookid = bookid;
            this.bookname = bookname;
            this.publishername = publishername;
            this.price = price;
            this.amount = amount;
           Console.WriteLine("count=" + num++);
        }//end b
        public void print()
        {
       Console.WriteLine(bookid + "   " + bookname + "   "+
publishername + "   " + price + "   " + amount);
         }//end print
        public string Bookname
        {
            get { return this.bookname;}
```

```
        set { this.bookname = value; }
    }
    public int Bookid
    {
        get { return this.bookid; }
        set { this.bookid = value; }
    }
    public int Price
    {
        get { return this.price; }
        set { this.price = value; }
    }
    public string Publishername
    {
        get { return this.publishername; }
        set { this.publishername = value; }
    }
    public int Amount
    {
        get { return this.amount; }
        set { this.amount = value; }
    }
    public static int Num
    {
        get { return num; }
       }
public void search(string search, book[] b, int n)
    {
        for (int i = 0; i < n; i++)
        {
            if (b[i].bookname == search)
                b[i].print();
            else
            { Console.WriteLine("no book"); }
        }//end for

    }//end search
    public int sell(book[] s,int bookid1,int n)
    {
        int bo = 0;
        for (int i = 0; i < n; i++)
        {
            if (s[i].bookid==bookid1)
            {
                bo = (s[i].amount - 1);
                s[i].amount = bo;
                s[i].print();
            }//end if
            else
                Console.WriteLine("noo book");
        }//end for
        return bo;
```

```
        }//end sell
           }


    }//end class book
class Program
    {
        static void Main(string[] args)
        {
            book z = new book();
            Console.WriteLine("enter book count");
            int n = Int32.Parse(Console.ReadLine());
            book[] e1 = new book[n];
            for (int i = 0; i < n; i++)
            {

                Console.WriteLine("enter bookid :"+(i+1));
                int a = Int32.Parse(Console.ReadLine());
                Console.WriteLine("enter bookname :"+(i+1));
                string b = Console.ReadLine();
                Console.WriteLine("enter publisher :" + (i + 1));
                string c = Console.ReadLine();
                Console.WriteLine("enter price :" + (i + 1));
                int d = Int32.Parse(Console.ReadLine());
                Console.WriteLine("enter amount :" + (i + 1));
                int e = Int32.Parse(Console.ReadLine());
                e1[i]=new book (a,b,c,d,e);
                e1[i].print();
            }
                Console.WriteLine("enter name book");
                string name1 = Console.ReadLine();
                book b2 = new book();
                b2.search(name1, e1, n);
                Console.WriteLine("enter book id");
                int l = Int32.Parse( Console.ReadLine());
                z.sell(e1, l, n);
                Console.ReadLine();
        }
    }
                  }
```

التحميل الزائد

Over load

١- اما اعادة تحميل الدوال .

٢- أو اعادة تحميل العوامل operator

للدوال : تعني وجود أكثر م دالة في نفس الصف تحمل نفس الاسم ولكن هنالك إخلاف بلائحة وسطاء الدالة (اما بعدد الوسطاء أو نوعها)

Ex

Ad(int x,int y)

Ad(int x1,int y1)
Ad(string x3,int y3)

Ex2:
Employee
Employee (string name,int age)
Employee(employee)

يتم التحميل الزائد في الدوال وذلك بوضع كلمة static بعد محدد الوصول واضافة كلمة (operator+رمز)لاستعداء الدالة باستخدام الرمز مثل دالة الجمع نستدعيها ب(+)وكل ذلك بعد اسم الدالة .

مثال:

```
namespace ConsoleApplication1
{
   class complex
    {
        private int image;
        private int real;
        public complex()
        {
            real = 0;
            image = 0;
        }
        public complex(int real, int image)
        {
            this.real = real;
            this.image = image;
        }
        public int Real
        {
            get { return this.real; }
            set { this.real = value; }
        }
        public int Image
        {
            get { return this.image; }
            set { this.image = value; }
        }
        public static complex operator +(complex i, complex j)
        {
            complex e = new complex();
            e.image = i.image + j.image;
            e.real = i.real + j.real;
            return e;
        }
        public static complex operator *(complex i1, complex j1)
        {
            return new complex((i1.real * j1.real) - (i1.image *
j1.image),
```

```
            (i1.real * j1.image) + (j1.real * i1.image));
        }
        public void print()
        {
            Console.WriteLine(real + "+i" + image);
        }
    }//end class

    class Program
    {
        static void Main(string[] args)
        {
            complex c0=new complex ();
            Console.WriteLine("enter real");
            int x1=Int32.Parse(Console.ReadLine());
            Console.WriteLine("enter image");
            int y1=Int32.Parse(Console.ReadLine());
            complex c1=new complex (x1,y1);
            c1.print();
            Console.WriteLine("enter real");
            int x2=Int32.Parse(Console.ReadLine());
            Console.WriteLine("enter image");
            int y2=Int32.Parse(Console.ReadLine());
            complex c2=new complex (x2,y2);
            c2.print();
            c0=c1+c2;
            Console.WriteLine("c1+c2=");
            c0.print();
            c0 = c1 * c2;
            Console.WriteLine("c1*c2=");
            c0.print();
            Console.ReadLine();}}}
```

سؤال:

اكتب كلاس لجمع وقسمة وضرب ومقارنة كسرين وقلب كسر مع تحميل زائد و مشيدات بنوعيها مع الطباعة .

الحل:

```
namespace ConsoleApplication1
{
    class fraction
    {
        private double numerator;
        private double denominator;
        public fraction()
        {
            numerator = 0;
            denominator = 0;
        }
        public fraction(double numerator, double denominator)
        {
            this.numerator = numerator;
            this.denominator = denominator;
```

```
        }
        public double Numerator
        {
            get { return this.numerator; }
            set { this.numerator = value; }
        }
        public double Denominator
        {
            get { return this.denominator; }
            set { this.denominator = value; }
        }
        public static fraction operator +(fraction i, fraction j)
        {
            fraction sum = new fraction();
           if (i.denominator == j.denominator)

            {
                sum.numerator = i.numerator + j.numerator;
                sum.denominator = i.denominator;
            }
            else
            {
                sum.numerator = ((i.numerator * j.denominator) +
(j.numerator * i.denominator));
                sum.denominator = (i.denominator * j.denominator);
            }
            return sum;
        }
        public static fraction operator *(fraction i1, fraction j1)
        {
            return new fraction(i1.numerator * j1.numerator,
i1.denominator * j1.denominator);
        }
        public fraction turned(fraction i2)
        {
            return new fraction(i2.denominator, i2.numerator);
        }

            public void print()
            {
                Console.WriteLine(numerator + "/" + denominator);
            }
            public static fraction   operator ^(fraction i3, fraction
j3)
        {

                double w=((i3.numerator) /
(i3.denominator));Console.WriteLine(w);
                double e = ((j3.numerator) /( j3.denominator));
Console.WriteLine(e);
                if (w > e)
                { Console.WriteLine(i3.numerator + "/" + i3.denominator
+ ">" + j3.numerator + "/" + j3.denominator); }
                else if (w < e)
```

```
                { Console.WriteLine(i3.numerator + "/" + i3.denominator
+ "<" + j3.numerator + "/" + j3.denominator); }
                else
                { Console.WriteLine(i3.numerator + "/" + i3.denominator
+ "=" + j3.numerator + "/" + j3.denominator); }
                return j3 ;

        }

    }//end class
    class Program
    {
        static void Main(string[] args)
        {
            fraction c = new fraction();
            Console.WriteLine("enter numerator");
            double x = double.Parse(Console.ReadLine());
            Console.WriteLine("enter denominator");
            double y=double.Parse(Console.ReadLine());
            fraction a = new fraction(x, y);
            a.print();
            Console.WriteLine("enternumerator");
           double x1 = double.Parse(Console.ReadLine());
            Console.WriteLine("enter denominator");
            double y1 = double.Parse(Console.ReadLine());
            fraction b = new fraction(x1, y1);
            b.print();
            c = a + b;
            Console.WriteLine("sum=");
            c.print();
            c = a * b;
            Console.WriteLine("a*b=");
            c.print();
            c = c.turned(c);
            Console.WriteLine("turned");
            c.print();
            c = a ^ b;
            Console.ReadLine();

        }
    }
}
```

سـؤال:

اكتب كلاس لإحداثيات نقطة point

١- بيانات(x,y).

٢- مشيدات و طباعة و get&&set

الحل:

```
namespace ConsoleApplication1
```

```
{
    class point
    {
        private int x;
        private int y;
        public point()
        {
            x = 0;
            y = 0;
        }
        public point(int x, int y)
        {
            this.x = x;
            this.y = y;
        }
        public int X
        {
            get { return this.x; }
            set { this.x = value; }
        }
        public int Y
        {
            get { return this.y; }
            set { this.y = value; }
        }
        public void printe()
        {
            Console.WriteLine("(" + x + "," + y + ")" );
        }
    }//end point

       class circle :point
        {
           public int r;
           public circle()
           {

              this. r = 0;
            }//constroctor
            public circle( int x, int y,int r):base(x,y)
           {

               this.r = r;
                }
             public int R
                {
                    get{return this.r;}
                    set{this.r=value;}
                }
             public void print()
             {
                 Console.WriteLine("(" +base.X + "," + base.Y + ")" +
"r=" + r);
             }
```

```
            public double mohet()
            {return 2*3.14*r;}
            public double t()
            {return 3.14*2*r;}
            public double s()
            {return 3.14*r*r;}


        }//end class
class Program
    {
        static void Main(string[] args)
        {
            point bu = new point(3,2);
            bu.printe();
            circle m = new circle();
            Console.WriteLine("enter x");
            int a = Int32.Parse(Console.ReadLine());
            Console.WriteLine("enter y");
            int b = Int32.Parse(Console.ReadLine());
            Console.WriteLine("enter r");
            int c = Int32.Parse(Console.ReadLine());
            circle s = new circle(a, b,c);
            s.print();
            Console.WriteLine("mhet="+s.mohet());
            Console.WriteLine("s=" + s.s());
            Console.ReadLine();
        }
    }
}
```

الـــــــــوراثـــــــــــــة

Inheritance

الوراثة في c# : تعني وجود صف أب base class نشتق منه الصف الابن حيث يرث الابن من الأب الأعضاء البيانية التي تملك محدد الوصول public أو protected .

Protected : (وصول محمي) أي دوال الصف نفسه ودوال الصف الابن (أي الصف المشتق) تستطيع الوصول إلى البيانات ويرث الأعضاء الدالية .

مثال : بناء الشجرة الهرمية لـ point ونشتق منها صف الدائرة circle

حيث الـ point له إحداثيات (x && y) .

والـ circle له إحداثيات (x&&y&&r) حيث r هي نصف القطر .

حيث ترث الدائرة إحداثيات الأب وتبني لنفسها نصف القطر .

مثال : وراثة دائرة لـ point وبناء كلاس الدائرة مع خصائص get&&set وحساب مساحة الدائرة ومحيطها:

```
namespace ConsoleApplication1
{
    class point
    {
        private int x;
        private int y;
        public point()
        {
            x = 0;
            y = 0;
        }
        public point(int x, int y)
        {
            this.x = x;
            this.y = y;
        }
        public int X//get&&set خصائص
        {
            get { return this.x; }
            set { this.x = value; }
        }
        public int Y//get&&set خصائص
        {
            get { return this.y; }
            set { this.y = value; }
        }
        public void printe()
        {
            Console.WriteLine("(" + x + "," + y + ")" );
        }
    }//end point

       class circle :point
        {
           public int r;
           public circle()
           {

              this. r = 0;
            }//constroctor
            public circle( int x, int y,int r):base(x,y)
           {

               this.r = r;
                }
             public int R//get&&set خصائص
                {
                    get{return this.r;}
                    set{this.r=value;}
                }
             public void print()
             {
```

```
                     Console.WriteLine("(" +base.X + "," + base.Y + ")" +
"r=" + r);//الدائرة احداثيات طباعة
                }
            public double mohet()//المحيط
            {return 2*3.14*r;}
            public double s()//المساحة
            {return 3.14*r*r;} }


        class Program
    {
        static void Main(string[] args)
        {
            point bu = new point(3,2);
            bu.printe();
            circle m = new circle();
            Console.WriteLine("enter x");
            int a = Int32.Parse(Console.ReadLine());
            Console.WriteLine("enter y");
            int b = Int32.Parse(Console.ReadLine());
            Console.WriteLine("enter r");
            int c = Int32.Parse(Console.ReadLine());
            circle s = new circle(a, b,c);
            s.print();
            Console.WriteLine("mhet="+s.mohet());
            Console.WriteLine("s=" + s.s());
            Console.ReadLine();
        }
    }

}
```

------------------------------ وظيـــفـــــــــــــة ------------------------------

- بناء الشجرة الوراثية لصف employee أب أعضاءه البيانية الاسم والعمر وله وريثان
- ١- موظف براتب شهري .(يرث من الأب الاسم والعمر ويبقى الراتب)
- ٢- موظف براتب يومي . .(يرث من الأب الاسم والعمر ويبقى الراتب)

الحــل :

```
namespace ConsoleApplication1
{
    class employee
    {
        private string n;
        private int a;
        private int count = 0;
        public employee()
        {

            a = 0;
            count++;
            Console.WriteLine("count=" + count);
        }
        public employee(string n, int a)
```

```
        {
            this.n = n;
            this.a = a;
        }
        public string N
        {
            get { return this.n; }
            set { this.n = value; }
        }
        public int A
        {
            get { return this.a; }
            set { this.a = value; }
        }
        public void printe()
        {
            Console.WriteLine("(" + n + "," + a + ")");
        }
    }//end employee

    class mensal : employee//شهري براتب موظف
    {
        public int s;
        public mensal()
        {

            this.s = 0;
        }//constroctor
        public mensal(string n, int a, int s)
            : base(n, a)
        {

            this.s = s;
        }
        public int S
        {
            get { return this.s; }
            set { this.s = value; }
        }
        public void print1()
        {
            Console.WriteLine("name:" + base.N + "-age:" + base.A + ","
+ "salare=" + s);
        }
    }
    class day : employee//يومي براتب موظف
    {
        public int d;
        public day()
        {

            this.d = 0;
        }//constroctor
        public day(string n, int a, int d)
```

```
            : base(n, a)
        {

            this.d = d;
        }
        public int S
        {
            get { return this.d; }
            set { this.d = value; }
        }
        public void print2()
        {
            Console.WriteLine("name:" + base.N + "-age:" + base.A + ","
+ "salare day=" + d);
        }
    }
class Program
    {
        static void Main(string[] args)
        {
            employee x = new employee();
            mensal c1 = new mensal();
            Console.WriteLine("enter count mensal");
            int n = Int32.Parse(Console.ReadLine());
            mensal[] m = new mensal[n];
            for (int i = 0; i < n; i++)
            {
                Console.WriteLine("enter name");
                string r = Console.ReadLine();
                Console.WriteLine("age");
                int z = Int32.Parse(Console.ReadLine());
                Console.WriteLine("salary");
                int e = Int32.Parse(Console.ReadLine());
                m[i] = new mensal(r, z, e);
                m[i].print1();
            }
            Console.WriteLine("enter count salary day");
            int u = Int32.Parse(Console.ReadLine());
            day[] m2 = new day[u];
            for (int i = 0; i < u; i++)
            {
                Console.WriteLine("enter name");
                string r1 = Console.ReadLine();
                Console.WriteLine("age");
                int z1 = Int32.Parse(Console.ReadLine());
                Console.WriteLine("salary day");
                int e1 = Int32.Parse(Console.ReadLine());
                m2[i] = new day(r1, z1, e1);
                m2[i].print2();
            }
            Console.ReadLine();
        }
    }
```

مفهوم over ride :

أي أن الصف الابن يمكنه كتابة دالة لها نفس اسم دالة ونفس نمط الاعادة ومحدد الوصول أكبر أو نفسه موجودة في كلاس الاب

- من أجل اجبار كلاس الابن على كتابة الدالة بنفس الاسم نضع قبل دالة الاب التي سوف نأخذ اسمها كلمة محجوزة virtual ووضع قبل نمط إعادة دالة الابن التي سوف تملك نفس الاسم كلمة محجوزة over ride وداخل هذه الدالة يمكننا استدعاء دالة الأب واضافة الدالة الجيدة هذا ما يسمى بـover ride .
- مثال:

```
namespace ConsoleApplication1
{
    class point
    {
        private int x;
        private int y;
        public point()
        {
            x = 0;
            y = 0;
        }
        public point(int x, int y)
        {
            this.x = x;
            this.y = y;
          }
        public int X
        {
            get { return this.x; }
            set { this.x = value; }
        }
        public int Y
        {
            get { return this.y; }
            set { this.y = value; }
        }
        public virtual void print()
        {
            Console.WriteLine("(" + x + "," + y + ")");
        }
    }//end point

    class circle : point
    {
        public int r;
        public circle()
        {

            this.r = 0;
        }//constroctor
          public circle(int x, int y, int r)
              : base(x, y)
          {
```

```
            this.r = r;
        }
        public int R
      {
            get { return this.r; }
            set { this.r = value; }
        }
      public override void print()
      {
          base.print();
          Console.WriteLine("r=" + r);
      }
      public double mohet()
        { return 2 * 3.14 * r; }
        public double s()
        { return 3.14 * r * r; }


    }//end class circle
    class cylinder: circle
    {
        private int rise;//الارتفاع
        public cylinder()
        { this.rise=0; }
        public cylinder (int x,int y, int r,int rise):base(x, y,
r)
        {
            this.rise = rise;
        }
        public int Rise
        {
            get { return this.rise; }
            set { this.rise = value; }
        }
        public override void print()
        {
          base.print();
          Console.WriteLine( "rise ="+ rise);
      }
      public double volume()
      { return 3.14 * r * r * rise; }//الحجم


  }//end lass cylinder class Program
  {
        static void Main(string[] args)
      {
          Console.WriteLine("enter coordinate point" );//إحداثيات
النقطة
          Console.WriteLine("enter x");
          int r1 = Int32.Parse(Console.ReadLine());
          Console.WriteLine("enter y");
          int r2 = Int32.Parse(Console.ReadLine());
          point q1 = new point(r1, r2);
```

```
            q1.print();
            Console.WriteLine("enter coordinate circle");//إحداثيات
الدائرة
            Console.WriteLine("enter x");
            int r3 = Int32.Parse(Console.ReadLine());
            Console.WriteLine("enter y");
            int r4 = Int32.Parse(Console.ReadLine());
            Console.WriteLine("enter r");
            int r5 = Int32.Parse(Console.ReadLine());
            circle q2 = new circle(r3, r4, r5);
            q2.print();
            Console.WriteLine("mohet="+q2.mohet());
            Console.WriteLine("s="+q2.s());
            Console.WriteLine("enter coordinate cylinder");// أحداثيات
الاسطوانة
            Console.WriteLine("enter x");
            int r6 = Int32.Parse(Console.ReadLine());
            Console.WriteLine("enter y");
            int r7 = Int32.Parse(Console.ReadLine());
            Console.WriteLine("enter r");
            int r8 = Int32.Parse(Console.ReadLine());
            Console.WriteLine("enter rise");
            int r9 = Int32.Parse(Console.ReadLine());
            cylinder q3 = new cylinder(r6, r7, r8, r9);
              q3.print();
            Console.WriteLine("volume=" + q3.volume());
            Console.ReadLine();
        }
    }
}
```

في هذا المثال : جعلنا دالة الطباعة بنفس الاسم للأب و للأبناء واستعينا دالة الطباعة الخاصة بالأب في الابناء .

وظيفة:

اكتب كلاس الـ employee يحوي الأعضاء البيانية:

١ـ اسم الموظف.

٢ـ رقمه التأميني.

وله الدوال:

- **خصائص get&&set لـ الاسم والرقم التاميني**
- **دالة طباعة القيم**
- **دالة getname تعود باسم الصف الذي منه الكائن.**
- **مشيدات .**

وله إبنان

١)ـ الأول براتب شهري وله الأعضاء البيانية

١ـ الراتب الشهري مع خصائص get&&set .

الدوال: مشيدات + طباعة + دالة getname تعود باسم الصف الذي منه الكائن.

٢)ـ الثاني براتب يومي حسب عدد ساعات العمل:

البيانات:
١ـ عدد ساعات العمل .
٢ـ أجرة الساعة .
دوال:
١ـ مشيدات .
٢ـ خصائص get&&set .
٣ـ دالة earning تعود بالأجر الكامل .
أخيراً استدعي كل ماسبق في دالة ال main .
الحل:

```
namespace ConsoleApplication1
{
    class employee
    {
        private string n;//الاسم
        private int s;//رقم التأمين
        public employee()
        {
          this.s = 0;
        }
        public employee(string n, int s)
        {
            this.n = n;
            this.s =s;
        }
        public string N
        {
            get { return this.n; }
            set { this.n = value; }
        }
        public int S
        {
            get { return this.s; }
            set { this.s = value; }
        }
        public virtual void print()
        {
            Console.WriteLine(" name:" + n +"/"+ "scurity:" + s );
        }
        public virtual void Getname()
        {
            Console.WriteLine("employee");
        }
    }//end employee

    class salarideemployee : employee
    {
        public int q;//راتب شهري
        public salarideemployee()
        {

            this.q = 0;
```

```
    }//constroctor
    public salarideemployee(string n, int s, int q)//يومي راتب
        : base(n, s)
    {
    this.q = q;
    }
    public int Q
    {
        get { return this.q; }
        set { this.q = value; }
    }
    public override void print()
    {
        base.print();
        Console.WriteLine( " salarideemployee=" + q );

    }
    public override void Getname()
    {
        Console.WriteLine("salarideemployee");
    }
}
class dailyemployee : employee//يومي راتب
{

   private int hours;//ساعات العمل
   private int wage;//أجرة الساعة
   public dailyemployee()
    {
        this.hours = 0;

        this.wage = 0;
    }//constroctor
   public dailyemployee(string n, int s, int hours, int wage)
        : base(n, s)
    {

        this.hours = hours;
        this.wage = wage;
    }

    public int Hours
    {
        get { return this.hours; }
        set { this.hours = value; }
    }
    public int Wage
    {
        get { return this.wage; }
        set { this.wage = value; }
    }

    public override void print()
    {
```

```
            base.print();
            Console.WriteLine ("hours="+hours+"wage="+wage+")");
        }
        public int earning()//العمل راتب ساعات العمل
        {
            return( this.wage * this.hours);
        }
        public override void Getname()
        {
            Console.WriteLine("salarideemployee");
        }
    }class Program
    {
        static void Main(string[] args)
        {
            employee x = new employee();
            Console.WriteLine("enter name");
            string n1 = Console.ReadLine();
            Console.WriteLine("enter security");
            int a1 = Int32.Parse(Console.ReadLine());
            employee c1 = new employee(n1,a1);
            c1.print();
            c1.Getname();
            salarideemployee c2 = new salarideemployee();
            Console.WriteLine("enter count salarideemployee");// عدد
الموظفين براتب شهري
            int n = Int32.Parse(Console.ReadLine());
            salarideemployee[] m = new salarideemployee[n];
            for (int i = 0; i < n; i++)
            {
                Console.WriteLine("enter name");
                string r = Console.ReadLine();
                Console.WriteLine("security");
                int z = Int32.Parse(Console.ReadLine());
                Console.WriteLine("salary");
                int e = Int32.Parse(Console.ReadLine());
                m[i] = new salarideemployee(r, z, e);
                m[i].print();
                m[i].Getname();
            }

            Console.WriteLine("enter count dailyemployee");//يومي راتب
            int u = Int32.Parse(Console.ReadLine());
            dailyemployee[] m2 = new dailyemployee[u];
            for (int i = 0; i < u; i++)
            {
                Console.WriteLine("enter name");
                string r1 = Console.ReadLine();
                Console.WriteLine("security");
                int z1 = Int32.Parse(Console.ReadLine());
                Console.WriteLine("hours");
                int e1 = Int32.Parse(Console.ReadLine());
                Console.WriteLine("wage");
                int e4 = Int32.Parse(Console.ReadLine());
```

```
                m2[i] = new dailyemployee(r1, z1, e1, e4);
                m2[i].print();
                Console.WriteLine(" earning="+m2[i].earning());
                m2[i].Getname();
            }
            Console.ReadLine();
        }
    }
}
```

مثال:
لدينا متجر لبيع الالكترونيات
١- لاب توب
اسم اللاب توب-رقمه-سعر-سرعة المعالج
٢-موبايلات
رقم الموبايل-اسمه-سعره-نوع الشاشة (لمس ام عادي)
مشيدات
get&&set
طباعة
مقارنة سعر جهازين من النوعين

انشاء مصفوفة لاب توبات وموبايلات وتطبيق الدوال.
الحل:

```
namespace ConsoleApplication1
{
    class A//المشتركة البيانات على الحاوي الاب كلاس
    {
        private int id;//الجهاز رقم
        private string n;//الجهاز اسم
        private int price;//الجهاز سعر
        public A()
        {
            this.id = 0;
            this.price = 0;
        }
        public A(int id,string n, int price)
        {
            this.id = id;
            this.price = price;
            this.n = n;
        }
        public int Id
        {
            get { return this.id; }
            set { this.id = value; }
        }
```

```
        public string N
        {
            get { return this.n; }
            set { this.n = value; }
        }
        public int Price
        {
            get { return this.price; }
            set { this.price = value; }
        }
        public virtual void print()
        {

Console.Write("id="+id+"/name:"+n+"/parce="+price);
        }
        public virtual void getname()
        {
            Console.WriteLine("class electronic");
        }


    }//end class A
    class B:A//كلاس  تـوب  الـلاب
    {
        private float cpu;//الـمعـالـج سرعة
        public B()
        {
            this.cpu = 0;
        }
        public B(int id, string n, int price, float cpu)
            : base(id, n, price)
        {
            this.cpu = cpu;
        }
        public float Cpu
        {
            get { return this.cpu; }
            set { this.cpu = value; }
        }
        public override void print()
        {
            base.print();
            Console.WriteLine("cpu speed="+cpu);
        }
        public override void getname()
        {
```

```
            Console.WriteLine("labtop");
        }
        public void f(B r, B t)
        {
            if (r.Price > t.Price)
                Console.WriteLine(r.N + ">" + t.N);

            else if (r.Price < t.Price)
               Console.WriteLine(r.N + "<" + t.N);
            else
                Console.WriteLine(r.N + "=" + t.N);
        }
    }//end class B
    class C : A//الموبايل كلاس
    {
        private string screen;//الشاشة نوع
        public C()
        { }
        public C(int id,string n,int price, string
screen):base(id,n,price)
        {
            this.screen = screen;
        }
        public string Screen
        {
            get { return this.screen; }
            set { this.screen = value; }
        }
        public override void print()
        {
            base.print();
            Console.WriteLine("screen:" + screen);
        }
        public override void getname()
        {
            Console.WriteLine("mobile");
        }
        public  void ff(C r1, C t1)
        {
            if (r1.Price > t1.Price)
            Console.WriteLine(r1.N+">"+t1.N);
            else if (r1.Price < t1.Price)
                Console.WriteLine(r1.N + "<" + t1.N);
            else
                Console.WriteLine(r1.N + "=" + t1.N);
        }
```

```
    }//end class cclass Program
    {
        static void Main(string[] args)
        {
            A e1 = new A();
            Console.WriteLine("enter count labtop");
            int z = Int32.Parse(Console.ReadLine());
            B[] e2 = new B[z];
            for (int i = 0; i < z; i++)
            {
                Console.WriteLine("enter id labtop");
                int z1 = Int32.Parse(Console.ReadLine());
                Console.WriteLine("enter name labtop");
                string z2 = (Console.ReadLine());
                Console.WriteLine("enter price labtop");
                int z3 = Int32.Parse(Console.ReadLine());
                Console.WriteLine("enter cpu speed labtop");
                float z4 = float.Parse(Console.ReadLine());
                e2[i]=new B(z1,z2,z3,z4);
            }
            for (int i = 0; i < z; i++)
            { e2[i].print(); }
            e2[0].f(e2[1], e2[2]);
            Console.WriteLine("enter count mobile");
            int u = Int32.Parse(Console.ReadLine());
            C[] e3 = new C[u];
            for (int i = 0; i <u; i++)
            {
                Console.WriteLine("enter id mobile");
                int w1 = Int32.Parse(Console.ReadLine());
                Console.WriteLine("enter name mobile");
                string w2 = (Console.ReadLine());
                Console.WriteLine("enter price mobile");
                int w3 = Int32.Parse(Console.ReadLine());
                Console.WriteLine("enter screen mobile");
                string w4 = Console.ReadLine();
                e3[i]=new C(w1,w2,w3,w4);
               }
            for (int i = 0; i < u; i++)
            { e3[i].print(); }
            e3[0].ff(e3[1],e3[2]);
            Console.ReadLine();

        }
    }
}
```

الصف المجرد والدالة المجردة:

- الصف المجرد : هو الصف الذي يحوي دالة مجردة واحدة على الأقل ويتم تعريفه بإضافة كلمة abstrac قبل كلمة class .
- الدالة المجردة :هي دالة تكون مشتركة بالاسم فقط بين الأبناء ويتم تعريفها بالأب بوضع كلمة abstrac بعد محدد الوصول حيث أن الدالة المجردة لا يمكن تعريفها بالصف الأب(لأنها تختلف في ألية عملها في الأبناء(مجردة)) ولكن يمكن التصريح عنها في الأبناء.

ملاحظة١: لا يمكن بناء غرض من الصف المجرد ولكن يمكن التصريح عنه.

ملاحظة٢:أي صف في c# هو ابن للصف object .

مثال :اكتب صف أب(شكل هندسي) مجرد يحوي الدالة المجردة getname التي وظيفتها بالأبناء أن تعود باسم الصف الابن ويحوي على دالة getarea تعود بالمساحة ودالة جمع مساحتي شكلين من الصفوف الأبناء،وله إبن وحفيد

دائرة:الأعضاء البيانية x,y,r

اسطوانة:x,y,r,h

مع خصائص get&&set

وكتابة دالة لـ Main

```
namespace ConsoleApplication1
{
        abstract class shape :Object
    {
        public abstract string getname();

        public virtual double getarea()
        {
            return 0.0;
        }
        public double sum(shape x,shape y)
        {
            return x.getarea() + y.getarea();
        }
    }
    class circle : shape
    {
        private int x ;
        private int y ;
        private double r;
        public circle()
        {
            this.x = 0;
            this.y = 0;
            this.r = 0;
        }
        public circle(int x, int y, double r)
```

```
    {
        this.x = x;
        this.y = y;
        this.r = r;
    }
    public int X
    {
        get { return this.x; }
        set { this.x = value; }
    }
    public int Y
    {
        get { return this.y; }
        set { this.y = value; }
    }
    public double R
    {
        get { return this.r; }
        set { this.r = value; }
    }
    public override string getname()
    {
        return "circle";
    }
    public override double getarea()
    {
        return Math.PI * r * r ;
    }
    public virtual string tostring()
    {
        return "("+x+","+y+","+r+")";
    }
}//end class circle
class cylinder : circle
{
    private double h;
    public cylinder()
    {
        this.h = 1;
    }
    public cylinder(int x, int y, double r, double h)
        : base(x, y, r)
    {
        this.h = h;
    }
     public double H
```

```
        {
            get { return this.h; }
            set { this.h = value; }
        }
       public override string getname()
        {
            return "cylinder";
        }
        public override double getarea()
        {
            return  2 * base.getarea() + 2 * Math.PI *
(base.R) * h;
        }
        public override string tostring()
        {
            return base.tostring() + "h="+h;
        }


    }//end class cylinder
class Program
    {
        static void Main(string[] args)
        {
            circle c = new circle(2,3,5.4);
            cylinder c1 = new cylinder(5,2,3.2,4.5);
            shape []c3=new shape[3];
            c3[0] = c;
            c3[1] = c1;
            c3[2] = new cylinder(2,4,6.5,5.4);
            for (int i = 0; i < 3; i++)
            {
                Console.WriteLine(c3[i].getname());
                Console.WriteLine(c1.getarea());
                Console.WriteLine(c.getname());
            }
            Console.ReadLine();
        }
    }}
```

مثال:لدينا شركة لارسال الطرود والحوالات يتطلب ارسال الطرود اسم المرسل و اسم المستلم ووزن الطرد والحوالات اسم المرسل و اسم الستلم و قيمة الحوالة .
ويجب حساب كلفة النقل (بالنسبة للطرود ٥٠ ليرة لكل ١كغ والحوالات ١% من قيمة المبلغ إذا كان أق من مليون و ٢% إذا كان أكثر من مليون.

إكتب لصفوف المناسبة مستخدما مفهوم الوراثة والتجريد.
الحل:

```
namespace ConsoleApplication1
{
    abstract class packet
    {
        private string name1 { get; set; }//المرسل
        private string name2 { get; set; }//المستلم
        public string Name1
        {
            get { return this.name1; }
            set { this.name1 = value; }
        }
        public string Name2
        {
            get { return this.name2; }
            set { this.name2 = value; }
        }
        public packet() { }
        public packet(string name1, string name2)
        {
            this.name1 = name1;
            this.name2 = name2;
        }
        public abstract double payment();
        public override string ToString()
        {
            Console.WriteLine("name1:" + name1 + "/name2:" +
name2);
            return "ww";
        }

    }///end class
    class porcel : packet//طرد
    {
        private int weight;
        public int Weight
        {
            get { return this.weight; }
            set { this.weight = value; }
        }
        public porcel() { this.weight = 0; }
        public porcel(string name1, string name2, int weight)
            : base(name1, name2)
        {
```

```
            this.weight = weight;
        }
        public override string ToString()
        {
            base.ToString();
            return " weight= " + weight;
        }
        public override double payment()
        {
            Console.Write("payment=");
            return 50 * weight;
        }
    }//end class
    class assignment : packet//مالية حوالة
     {
        private int mony;
        public int Mony
        {
            get { return this.mony; }
            set { this.mony = value; }
        }
        public assignment(string name1, string name2, int
mony)
            : base(name1, name2)
        {
            this.mony = mony;
        }
        public override string ToString()
        {
            base.ToString();
            return "mony" + mony;
        }
        public override double payment()
        {
            if (mony > 100000)
            {
                Console.Write("payment=");
                return 0.02*mony;
            }
            else
            {
                Console.Write("payment=");
                return 0.01 *mony;
            }
        }
```

```
 }//end class class Program
{
    static void Main(string[] args)
    {
       porcel a = new porcel("ahmed","ali", 2);//
       Console.WriteLine(a.ToString());//
       Console.WriteLine(a.payment());//
       porcel zz = new porcel("khaled","mohamad",3);//
       Console.WriteLine(zz.payment());//
       Console.WriteLine("enter count porcel");//
        int n = Int32.Parse(Console.ReadLine());//
        porcel[] a1 = new porcel[n];
        for (int i = 0; i < n; i++)
        {
            Console.WriteLine("enter name1:" + (i + 1));//
            string z1 = Console.ReadLine();//
            Console.WriteLine("enter name2:");//
            string z2 = Console.ReadLine();//
            Console.WriteLine("enter wigh:");//
            int z3 = Int32.Parse(Console.ReadLine());//
            a1[i] = new porcel(z1, z2, z3);
        }
            a1[0].ToString();//
          Console.WriteLine( a1[0].payment());//
          Console.WriteLine("enter count assignment");
          int n1 = Int32.Parse(Console.ReadLine());
          assignment[] a2 = new assignment[n1];
          for (int i = 0; i < n1; i++)
          {
              Console.WriteLine("enter name1:" + (i +
1));//
              string s1 = Console.ReadLine();//
              Console.WriteLine("enter name2:");//
              string s2 = Console.ReadLine();//
              Console.WriteLine("enter amount:");//
              int s3 = Int32.Parse(Console.ReadLine());//
              a2[i] = new assignment(s1, s2, s3);
          }
          a2[0].ToString();//
          Console.WriteLine(a2[0].payment());//
       Console.ReadLine();
    }
}
```

Khaled234580@gmail.com