Chapter 6

Classes & Object Oriented Programming

كثير من الناس عبارة عن مجرد ظاهرة صوتية.

(*Introduction to classes*)

write a program that reads 3 integers and prints the minimum and the maximum of them.

```
enter 3 integers:  10   3   7
minimum is:  3
maximum is:  10
```

```
# include <iostream>
  using namespace std;

int  main ( )
{
   int x, y, z, min, max;

   cout << "enter 3 integers:  ";

   cin >> x >> y >> z;


   min = x;

  if (y < min)
       min = y;

   if (z < min)
       min = z;


    max = x;

   if (y > max)
       max = y;

   if (z > max)
       max  = z;


   cout << "minimum is:  "  <<  min   << endl;

   cout << "maximum is:  " <<  max  << endl;

   return 0;
}
```

حل مباشر بدون استخدام دوال وبدون classes.

عيب هذه الطريقة
صعوبة تقسيم العمل وصعوبة إعادة استخدام أجزاء منه ببرامج أخرى.

كل الناس يكتمون الأسرار، عدا أولئك الذين نفضي لهم بأسرارنا.

(*Introduction to classes*)

write a program that reads 3 integers, and calls functions that return their minimum and maximum, the main function then calls a void function to print the results.

```
enter 3 integers:  10  3  7
minimum is:  3
maximum is:  10
```

```
# include <iostream>
  using namespace std;

int minimum (int, int, int);

int maximum (int, int, int);

void print (int, int);


int  main ( )
{
    int  x, y, z, min, max;

    cout << "enter 3 integers:  ";

    cin >> x >> y >> z;

    min = minimum (x, y, z);

    max = maximum (x, y, z);

    print (min, max);

    return 0;

}
```

function prototype

حل نفس السؤال السابق
مع استخدام دالة.

function name

function call

عيب هذه الطريقة
تمرير قيم أثناء استدعاء الدوال
يستغرق وقتاً ويتطلب مخازن جديدة.

لا تجعل مساعدتك الآخرين مجرد
فعل ما يتوجب عليهم فعله.

return type

function parameters

```
int  minimum (int  a,  int  b,  int  c)
```

function header

```
{
    int  min;

    min = a;

    if (b < min)
        min = b;

    if (c < min)
        min = c;

    return min;
}
```

return value

هذه طريقة لحساب min أو max.

```
int  maximum (int  x,  int  y,  int  z)
{
    if (x > y && x > z)
        return x;

    if (y > z)
        return y;

    return z;
}
```

function header

وهذه طريقة بديلة لحساب min أو max.

```
void print (int min, int max)
{
    cout << "minimum is:  " << min << endl;

    cout << "maximum is:  " << max << endl;
}
```

لا توقف طفل عند حدود تعلمك، فهو مولود في زمن مختلف.

(Converting to classes)

```
# include <iostream>
  using namespace std;
class values
{
    private:
        int x, y, z;

    public:
         int minimum ( )
        {
                if (x < y && x < z)
                        return x;
                if (y < z)
                        return y;
                return z;
         }

        int maximum ( )
        {
                int max = x;
                if (y > max)
                        max = y;
                if (z > max)
                        max = z;
                return max;
        }

        void print (int min, int max )
        {
                cout << "minimum is: " << min << endl;
                cout << "maximum is: " << max << endl;
        }

       values (int a, int b, int c)
        {
                x = a;
                y = b;
                z = c;
        }
 };
```

مزايا الكلاسس
التغلب على عيوب طريقتي البرنامج
الواحد والدوال، ومزايا أخرى كثيرة.

Constructor

لا يتوقف الناس عن اللعب
لأنهم كبروا، بل يكبرون
لأنهم توقفوا عن اللعب.

find the exact output:

```
int main ( )
{
        values  p1 (10, 3, 7), p2 (8, 6, 9);

        int  min, max;

        min = p1.minimum ( );

        max = p1.maximum ( );

        p1.print (min, max);

        min = p2.minimum ( );

        max = p2.maximum ( );

        p2.print (min, max);

        return 0;
 }
```

	x	y	z
p1	10	3	7

min ~~3~~ 6 max ~~10~~ 9

	x	y	z
p2			

output

```
minimum is: 3
maximum is: 10
```

```
minimum is: 6
maximum is: 9
```

لا يســتطيع أحــد ركــوب ظهرك إلا إذا كنت منحنياً.

(Standard classes)

// A simple class to show constructor, default values, set and get functions.

```
# include <iostream>
  using namespace std;

class easy
{
    private:

        int mark;

    public:

        void  print ( )
        {
            cout << "mark is: " << mark << endl;
        }


        void  set (int x)
        {
            mark = x;
        }


        int  get ( )
        {
            return mark;
        }


        easy (int x = 0)
        {
            mark = x;
        }
};
```

Print
* تستخدم لطباعة قيم الـ attributes.
* نوعها void.
* فيها cout.
* لا تستقبل معاملات ( ).

set
* تستخدم لتبديل قيم الـ attributes.
* تستقبل معاملاً واحداً.
* نوعها void.
* ليس فيها return.

get
* تستخدم لإعادة قيم الـ attributes.
* لها نوع خلاف void.
* لا تستقبل معاملات ( ).
* تعيد قيمة.

constructor
* يستخدم لإعطاء قيم أو أولية للـ attributes.
* ليس له نوع.
* له نفس اسم class.
* يتم تنفيذه تلقائياً عند تعريف object من نوع الـ class.
* يستخدم لإعطاء قيم أولية.

لا يمكن لأحد أن يُشعرك بالوضاعة من دون موافقتك؟

find the exact output:

```
int main ( )
{
        easy  m1 (93), m2;

        int y;

        m1.print ( );


        y = m1.get ( ) + 4;

        cout << "after curve mark is: " << y << endl;


        m2.print ( );


        m2.set (71);

        m2.print ( );


        return 0;
}
```

m1 mark 93

m2 mark ~~0~~ 71

y 97

output

```
mark is: 93

after curve mark is: 97

mark is: 0

mark is: 71
```

(class definition and class implementation)

Sept 2013

```
# include <iostream>
  using namespace std;
class easy
{
      private:
          int mark;
      public:
        void print ( );

        void set ( int );

        int get ( );

        easy ( int = 0 );
};
```

class definition
غالبا يتم فصله في ملف مستقل عن
class implementation
لثلاثة أسباب رئيسية.

```
void easy::print ( )
{
        cout << "mark is: " << mark << endl;
}

void easy::set (int x)
{
        mark = x;
}

int easy::get ( )
{
        return mark;
}

easy::easy (int x)
{
        set (x);
}
```

لا يوجد سبب يحبك الناس من أجله
أفضل من عدم التطلع لما في أيديهم.

find the exact output:

```
int main ( )
{
        easy  m1 (93);

        int  y;

        m1.print ( );

        y = m1.get ( ) + 4;

        cout << "after curve mark is: " << y << endl;


        easy m2;

        m2.print ( );

        m2.set (71);

        m2.print ( );

        return 0;
}
```

mark

m1 | 93

y | 97

mark

m2 | ~~0~~

71

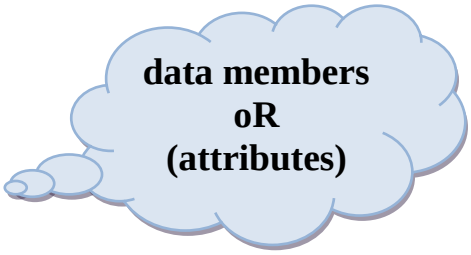
output

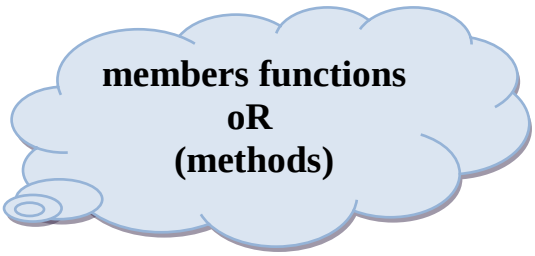
```
mark is: 93

after curve mark is: 97

mark is: 0

mark is: 71
```

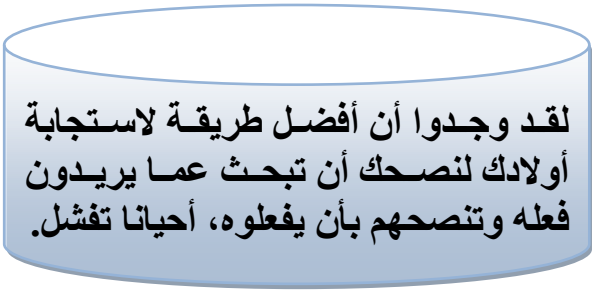
لقد لاحظت أن كل من يدعون
للإجهاض قد وُلِدُوا بالفعل.

(Complex Class)

```
# include <iostream>
# include <cstring>
  using namespace std;

class Student
{
      private:
            int id;
            string name;
            double mark;

      public:
            void  print ( )
            {
                  cout << "ID =  " << id << endl;
                  cout << "Name =  " << name << endl;
                  cout << "Mark =  " << mark << endl;
            }

            void  setId (int  d)
            {
                  if  (d < 0)
                  {
                     cout  <<  "Invalid ID.\n";
                     id = 0;
                   }

                  else
                     id = d;
             }

             void  setName (string  s)
             {
                   name = s;
             }
```







```
void  setMark (double  m)
{
        if  (m >= 0 && m <= 100)
                mark = m;

        else
        {
                cout << "Invalid mark.\n";
                mark = - 99;
        }
}

 int  getId ( )
{
        return id;
}

string  getName ( )
{
        return name;
}

 double  getMark ( )
{
        return mark;
}

 Student (string s =  " ", int  d = 0, double  m = 0)
{
        setName (s);
        setId (d);
        setMark (m);
}
};
```

أحياناً يفضل وضع قيمة غريبة في المتغير للدلالة على حدوث خطأ في عملية الإدخال مثل (99-).

يفضل إعطاء قيم ابتدائية مناسبة للمتغيرات في حال عدم توفر قيمها بالوقت الحالي.

لو لم تكن جزءاً من الحل فعلى الأقل لا تكن جزءا من المشكلة.

```
int main ( )
{
        Student  s1, s2;

        s1.setId (10);

        s1.setName ("Abdul rahman");

        s1.setMark (735);

        s1.print ( );

        int d;
        string s;
        double m;

        cout << "Enter ID, name & mark for student 2: ";
        cin >> d >> s >> m;          // -20  Somaiya  81.3


        s2.setId (d);
        s2.setName (s);
        s2.setMark (m);

        s2.print ( );
        return 0;
}
```

	id	name	mark
s1	10	Abdul rahman	-99

d	s	m
-20	Somaiya	81.3

	id	name	mark
s2	0	Somaiya	81.3

output

```
Invalid mark.
ID = 10
Name = Abdul rahman
Mark = -99

Enter ID, name & mark for student 2: -20  Somaiya  81.3
Invaid ID.
ID = 0
Name = Somaiya
Mark = 81.3
```

ليست الحقيقة دوماً التي
يموت من أجلها الناس.

Classes (class Rectangle)

Create a class *Rectangle* that has two attributes *Length* & *Width* of type (integer) and the following member functions:

Rectangle that calls the *setLength* & *setWidth* functions.
(default Length = 1, default Width = 1).

getLength that returns the *Length* value.

getWidth that returns the *Width* value.

setLength that sets the *Length* according to the passed parameter if it is positive, otherwise it shows an error message and sets it to 0.

setWidth that sets the *Width* to the passed parameter if it is positive, otherwise it shows an error message and sets it to 0.

Area that returns the Area of the *rectangle*.

Perimeter that returns the perimeter of the *rectangle*.

```
# include <iostream>
  using namespace std;
class Rectangle
{
  private:

        int  Length, Width;

  public:

        Rectangle (int  L = 1,  int W = 1)
        {
              setLength (L);
              setWidth (W);
        }
```

ما رأيت ظالماً أشبه
بمظلوم من الحاسد.

```
void setLength (int  L)
{
   if  (L < 0)
   {
         cout << "Invalid Length\n";
         Length  = 0;
   }

   else
         Length = L;
}

int getLength ( )
{  return Length;  }

void setWidth (int  W)
{
   if  (W < 0)
   {
         cout << "Invalid Width\n";
         Width  = 0;
   }

   else
         Width = W;
}


int getWidth ( )
{  return Width;  }

int Area ( )
{  return  Length * Width;  }


int Perimeter ( )
{  return 2 * (Length + Width);  }
};
```

ما كان للذئب أن يكون ذئباً لو
لم تكن الخرافُ خرافاً.

Classes (class *Circle*)

Create a class *Circle* that has one attribute *Radius* of type (double) and the following member functions:

Circle that calls the *setRadius* function (default Radius = 0).

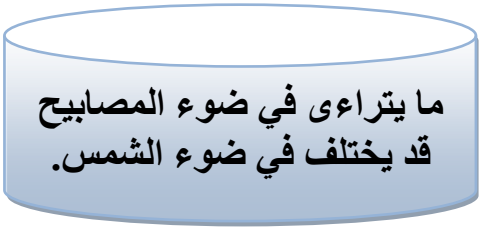
getRadius that returns the *Radius* value.

setRadius that sets the *Radius* according to the passed parameter if it is between 0 - 30, otherwise it shows an error message and sets it to 0.

Area that returns the Area of the *circle*.

Perimeter that returns the perimeter of the *circle*.

(The answer is on the next page)



```
# include <iostream>
  using namespace std;
class Circle
{
  private:
        double Radius;
  public:
        Circle (double r = 0)
        {
              setRadius (r);
        }

        void setRadius (double  r)
        {
              if (r < 0 || r > 30)
              {
                    cout << "Invalid value\n";
                    Radius = 0;
              }

              else
                    Radius = r;
        }

        double getRadius (  )
        {
              return Radius;
        }

        double Area (  )
        {
              return 3.14 * Radius * Radius;
        }

        double Perimeter (  )
        {
              return 2 * 3.14 * Radius;
        }
};
```

في حساب المساحة والمحيط لابد نستخدم (Radius) وليس (r).

معظـــم النـــاس يقضـــون حيـــاتهم محاولين إظهار محاسن ليست فيهم.

Find the errors in the following class

```
class Student

        private

                int x;

        public

                int get ( int x )
                {
                        return x;
                }



                int set ( int a )
                {
                        x = a;
                }


                void student ( int a )
                {
                        x = a;
                }


}
```

من النادر أن نفكر فيما نملك، الأغلب
أننا مشغولون بما لا نملك.

Exercise 6.7 Create a class called Rational for performing arithmetic with fractions. Write a program to test your class. Use integer variables to represent the private data of the class – the numerator and the denominator. Provide a constructor that enables an object of this class to be initialized when it is declared. The constructor should contain default values in case no initializers are provided and should store the fraction in reduced form. For example, the fraction 2/4 would be stored in the object as 1 in the numerator and 2 in the denominator. Provide public member functions that perform each of the following tasks:

a) Adding two Rational numbers. The result should be stored in reduced form.
b) subtracting two Rational numbers. The result should be stored in reduced form.
c) Multiplying two Rational numbers. The result should be stored in reduced form.
d) Dividing two Rational numbers. The result should be stored in reduced form.
e) Printing Rational numbers in the form a/b, where a is the numerator and b is the denominator.
f) Printing Rational numbers in floating-point format.

```
# include <iostream>
# include <iomanip>
  using namespace std;

class Rational
{
        private:
                int n, d;

        public:
                Rational ( int = 1, int = 1 );
                void reduced (  );
                void setN ( int );
                void setD ( int );
                int getN ( );
                int getD ( );
                Rational add ( Rational );
                Rational sub ( Rational );
                Rational mult ( Rational );
                Rational div ( Rational );
                void printR ( );
                void printF ( );
};
```

من أشد السوء أن يكون وجودك من عدمه مستويا عند كل من يعرفك.

```
Rational::Rational ( int N, int D )
{
        n = N;
        d = D;
        reduced ( );
}


void Rational::reduced ( )
{
        int min, gcd;

        if ( n < d )
                min = n;
        else
                min = d;

        for ( int i = 1; i <= min; i++ )
       if ( n % i == 0 && d % i == 0 )
          gcd = i;

    n /= gcd;
    d /= gcd;
}


void Rational::setN ( int x )
{
        n = x;
}


void Rational::setD ( int x )
{
        d = x;
}


int Rational::getN ( )
{
        return n;
}


int Rational::getD ( )
{
        return d;
}
```

من لا يملك نصف يومه على الأقل
يمكن تصنيفه على أنه عبد.

```
Rational Rational::add ( Rational R )
{
	Rational R2;

	if ( d != R.d )
	{
		n *= R.d;
		R.n *= d;
		d *= R.d;
	}

	R2.d = d;
	R2.n = n + R.n;

	R2.reduced ( );

	return R2;
}

Rational Rational::sub ( Rational R )
{
	Rational R2;

	if ( d != R.d )
	{
		n *= R.d;
		R.n *= d;
		d *= R.d;
	}

	R2.d = d;
	R2.n = n - R.n;

	if ( R2.n > 0 )
		R2.reduced ( );
	else
	{
		R2.n *= -1;
		R2.reduced ( );
		R2.n *= -1;
	}

	return R2;
}
```

نحن مجانين إذا لم نستطع أن نفكر، متعصبون إذا لم نرد أن نفكر، عبيد إذا لم نجرؤ أن نفكر.

```
Rational Rational::mult ( Rational R )
{
        Rational R2;

        R2.n = n * R.n;
        R2.d = d * R.d;

        R2.reduced ( );

        return R2;
}


Rational Rational::div ( Rational R )
{
        Rational R2;

        R2.n = n * R.d;
        R2.d = d * R.n;

        R2.reduced ( );

        return R2;
}


void Rational::printR ( )
{
        cout << "Fractional: " << n << "/" << d << endl;
}


void Rational::printF ( )
{
        cout << fixed << setprecision(3) << "Decimal: " << n * 1.0 / d << endl;
}


int main ( )
{
        int n, d;
        int x, y, z;

        cout << "Enter the first rational number, n then d: ";
        cin >> n >> d;
        Rational X (n, d);
```

نحن نحب الماضي لأنه ذهب، ولو عاد لكرهناه.

```
        cout << "Enter the second rational number, n then d: ";
        cin >> n >> d;
        Rational Y (n, d);

        Rational Z;
        cout << "\nThe addition result:\n";
        Z = X.add (Y);
        Z.printR ( );
        Z.printF ( );


        cout << "\nThe subtraction result:\n";
        Z = X.sub (Y);
        Z.printR ( );
        Z.printF ( );

        cout << "\nThe multiplication result:\n";
        Z = X.mult (Y);
        Z.printR ( );
        Z.printF ( );

        cout << "\nThe division result:\n";
        Z = X.div (Y);
        Z.printR ( );
        Z.printF ( );

        return 0;
}
```

نقضي عشرين عاما لبناء سمعة
جيدة، خمس دقائق تكفي لهدمها.