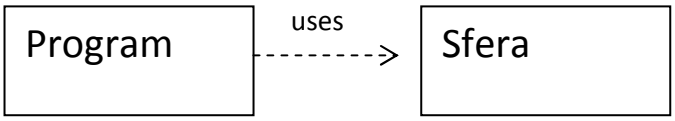
Introduzione alla Programmazione in C#

con approccio Object-First + Project-Driven

Per cominciare si pensa ad un programma che calcoli il volume di una sfera.

Innanzitutto si pensa ai moduli componenti di questo programma, che tecnicamente vengono detti CLASSI; uno è il Program che dà l'avvio all'esecuzione del programma stesso e l'altro è Sfera che rappresenta la singola sfera di cui si vuole calcolare il volume:

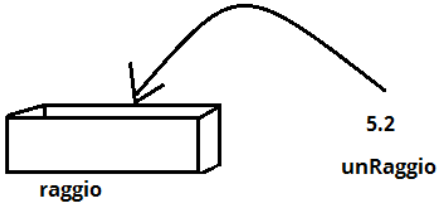


Il suddetto diagramma rappresenta le classi del programma.
Ogni classe del programma è costituita da un Nome, un elenco di Campi per memorizzare i valori utili per le elaborazioni, un Costruttore per inizializzare i valori dei campi, un elenco di Metodi per codificare le operazioni di interesse.
La codifica in C# della classe Sfera è la seguente:

Le righe che iniziano con // sono dei commenti che descrivono le istruzioni del linguaggio

```
public class Sfera
{
      // campi o field
      private double raggio;

      // costruttore ➔ per inizializzare i campi con un dato fornito in input
      // in questo caso in input c'è il valore denominato unRaggio
      public Sfera(double unRaggio)
      {
          raggio = unRaggio; // assegnazione di un valore al campo raggio
      }

      // metodi ➔ operazioni effettuate sui campi
      public double Volume()    // il risultato è di tipo double!
      {        4.0
         return 4/3*raggio*raggio*raggio*3.1415926;
}
}
```

I campi sono PRIVATI in quanto si possono usare solo dentro i metodi della classe medesima.
I metodi sono PUBBLICI in quanto si possono usare sia all'interno della classe che in altre classi.



Esempio di assegnazione

Tipi numerici

- int o Int32 occupa 32 bit = 4 byte di memoria e consente di rappresentare numeri interi da circa -2 miliardi a +2 miliardi.
- double o Double occupa 8 byte di memoria e consente di memorizzare numeri reali approssimati con massimo 16 cifre significative. Ad esempio 1.23456789 E+90 = $1.23456789 * 10^{90}$
- decimal o Decimal occupa 16 byte di memoria e consente di memorizzare numeri con la virgola di al massimo 28 cifre. Sono ottimi per la memorizzazione di importi monetari come ad esempio 10000000000000000000.01 (dieci miliardi di miliardi e un centesimo)

La classe Program gestisce il dialogo con l'utente e utilizza le funzionalità della classe Sfera:

```
public class Program
{
      public static void Main()
      {
            Console.WriteLine("BUONGIORNO");
            Console.WriteLine("Dammi il raggio della sfera");

            double unRaggio = Convert.ToDouble( Console.ReadLine() );

            // creo un oggetto di tipo Sfera
            Sfera s = new Sfera(unRaggio);
            // … ne calcolo il volume
            double v = s.Volume();
            Console.WriteLine("Il volume è " + v);
      }
}
```

L'input da tastiera è sempre considerato come una stringa di caratteri di tipo testo e pertanto risulta necessario effettuare una conversione al tipo double

Esecuzione del programma nella finestra Console:

```
BUONGIORNO
Dammi il raggio della sfera
5.2
Il volume è 441.7330523008
```

Notare che il linguaggio è CASE SENSITIVE, ovvero distingue tra Maiuscolo e minuscolo. In generale i nomi delle classi e quelli dei metodi iniziano in Maiuscolo, mentre quelli dei campi sono in minuscolo.

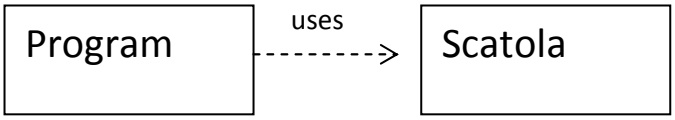
ESERCIZIO 1

Scrivere un programma per calcolare Volume, Superficie e lunghezza della Diagonale di una Scatola di forma rettangolare.

La classe Program dovrà anche confrontare tra loro i volumi di due diverse scatole per dire quale delle due è maggiore.

Soluzione

Il diagramma delle classi:




```
public class Scatola
{
 // campi
 private double altezza;
 private double larghezza;
 private double profondità;

// costruttore  con tre variabili in input public Scatola(double una Altezza,
double una Larghezza, double unaProfondità)
{
  altezza = una Altezza;
  larghezza = una Larghezza;
  profondità = unaProfondità;
}

// metodi
public double Volume()
{
  return Altezza*Larghezza*Profondità;
}

public double Superficie()
{
  return 2*Altezza*Larghezza+2*Altezza*Profondità+2*Larghezza*Profondità;
}
```

```
public double Diagonale()
{
return Math.Sqrt(Altezza*Altezza+Larghezza*Larghezza+Profondità*Profondità); }
}
```

Si noti l'uso di una funzione matematica della libreria Math per calcolare la square root (radice quadrata).

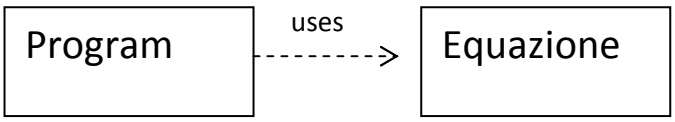
```
public class Program
{
  public static void Main()
  {
    Console.WriteLine("Dammi la Altezza della Scatola");
    double unaAltezza = Convert.ToDouble(Console.ReadLine());
    Console.WriteLine("Dammi la Larghezza della Scatola");
    double unaLarghezza = Convert.ToDouble(Console.ReadLine());
    Console.WriteLine("Dammi la Profondità della Scatola");
    double unaProfondità = Convert.ToDouble(Console.ReadLine());
    // attenzione a rispettare l'ordine dei parametri da fornire in input
    // al costruttore della Scatola!
    Scatola s1 = new Scatola(una Altezza, una Larghezza, una Profondità);
    // calcoli
    double v1 = s1.Volume();
    Console.WriteLine("Il volume è " + v);
    // ora costruisco un'altra scatola
    // posso riutilizzare le variabili dichiarate prima
    Console.WriteLine("Dammi la Altezza della seconda Scatola");
    unaAltezza = Convert.ToDouble(Console.ReadLine());
    Console.WriteLine("Dammi la Larghezza della Scatola");
    unaLarghezza = Convert.ToDouble(Console.ReadLine());
    Console.WriteLine("Dammi la Profondità della Scatola");
    unaProfondità = Convert.ToDouble(Console.ReadLine());
    // creo la seconda scatola
    Scatola s2 = new Scatola(una Altezza, una Larghezza, una Profondità);
    double v2 = s2.Volume();
    // confronto i volumi
    if (v1 > v2)  // se la condizione è vera
    { Console.WriteLine("La prima scatola è più grande della seconda"); }
    else   // altrimenti
    { Console.WriteLine("La prima scatola non è più grande della seconda"); }
  }
}
```

ESERCIZIO 2

Scrivere un programma per risolvere una equazione di primo grado posta nella forma ax + b = 0
Si ricorda che se a ≠ 0 si ha x = -b/a
Se a = 0 e b ≠ 0 l'equazione è Impossibile, se a = 0 e b = 0 è Indeterminata

Soluzione

Il diagramma delle classi:




```
using System;
public class Equazione
{
	//Campi
	private double coefficienteA;
	private double coefficienteB;
	private string tipoDiSoluzione; //Determinata, Indeterminata, Impossibile

	//Costruttore
	public Equazione (double A, double B)
	{
		coefficienteA = A;
		coefficienteB = B;
		tipoDiSoluzione = "non lo so";
	}

	//Metodi
	public double Soluzione()
	{
		if (coefficienteA != 0)
		{
			tipoDiSoluzione = "Determinata";
			return -coefficienteB/coefficienteA;
		}
		else
		{
			if (coefficienteB != 0)
			{
				tipoDiSoluzione = "Impossibile";
				return Double.NaN; // Not a Number
			}
```

```
                else
                {
                      tipoDiSoluzione = "Indeterminata";
                      return Double.NaN;
                }
          }
      }

      public string DammiTipoSoluzione()
      {
            return tipoDiSoluzione;
      }

      public string DammiEquazione()
      {
            if (coeffA > 0)
            { return coefficienteA + "x +" + coefficienteB + " = 0"; }
            else
            {
               if (coeffB == 0)
                  { return coefficienteA + "x = 0"; }
               else   // coefficienteB < 0
                 { return coefficienteA + "x " + coefficienteB + " = 0"; }
            }
      }
}


public class Program
{
      public static void Main ()
      {
            Console.WriteLine ("dammi il coefficiente della x");
            double A = Convert.ToDouble(Console.ReadLine());
            Console.WriteLine ("dammi il termine noto");
            double B = Convert.ToDouble(Console.ReadLine());
            Equazione e = new Equazione (A,B);
            string s = e.DammiEquazione();
            Console.WriteLine (s);   // scrive a video l'equazione
            double x = e.soluzione();
            if (e.DammiTipoSoluzione()=="Determinata")
            {
                  Console.WriteLine ("x =" + x);   // la soluzione
            }
            else
            {
                  Console.WriteLine(e.DammiTipoSoluzione());
            }
```

```
        }
}
```

Esecuzione del programma:

dammi il coefficiente della x

3

dammi il termine noto

4

3x +4 = 0

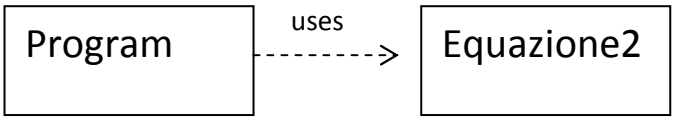
x =-1.33333333333333

ESERCIZIO 3

Scrivere un programma per risolvere una equazione di secondo grado posta nella forma $ax^2 + bx + c = 0$
Si suppone $a \neq 0$.

Soluzione

Il diagramma delle classi:




```
using System;
public class Equazione2
{
      //Campi
      private double coefficienteA;
      private double coefficienteB;
      private double coefficienteC;

      private string tipoDiSoluzione; //Determinata, Impossibile
      private double soluzione1;
      private double soluzione2;

      //Costruttore
      public Equazione (double A, double B, double C)
      {
            coefficienteA = A;
            coefficienteB = B;
            coefficienteC = C;

            tipoDiSoluzione = "non lo so";
            soluzione1 = Double.NaN;
            soluzione2 = Double.Nan;
      }

      //Metodi
      public double Risolvi()
      {
          // calcolo il delta e lo memorizzo in una variabile di appoggio
          double delta = coefficienteB * coefficienteB – 4 * coefficienteA *
                         coefficienteC;
          if (delta >= 0)
          {
              tipoDiEquazione = “determinata”;
              soluzione1 = (-coefficienteB – Math.Sqrt(delta)) /
```

Double.NaN indica che non è un numero valido (Not a Number)

```
                        (2 * coefficienteA);
            soluzione2 = (-coefficienteB + Math.Sqrt(delta)) /
                        (2 * coefficienteA);
         }
         else
         {  tipoDiEquazione = “impossibile”; }

    }


    public string DammiTipoDiEquazione()
    {  return tipoDiEquazione;  }

    public string DammiEquazione()
    {
       // usa la string interpolation
       string segnoB;
       string segnoC;
       if (b>=0) { segnoB = “+”} else { segnoB = “”; }
       if (c>=0) { segnoC = “+”} else { segnoC = “”; }

       return $”{coefficienteA}x^2 {segnoB}{coefficienteB}x
                {segnoC}{coefficienteC}”;
    }

    public double DammiSoluzione1()
    {   return  soluzione1; }

    public double DammiSoluzione2()
    {   return  soluzione2; }
}



public class Program
{
     public static void Main()
     {
         Console.WriteLine(“Dammi il coefficiente a”);
         double a = Convert.ToDouble(Console.ReadLine());
         Console.WriteLine(“Dammi il coefficiente b”);
         double b = Convert.ToDouble(Console.ReadLine());
         Console.WriteLine(“Dammi il coefficiente c”);
         double c = Convert.ToDouble(Console.ReadLine());
         if (a == 0)
         {
               Console.WriteLine(“non è una equazione di secondo grado”);
         }
```

```
        else
        {
             Equazione2 e = new Equazione2(a, b, c);
             Console.WriteLine(e.DammiEquazione());  // scrive l'equazione
             e.Risolvi();
             string tipo = e.DammiTipoDiEquazione();
             if (tipo == "impossibile")
             {
                      Console.WriteLine("l'equazione è impossibile");
             }
             else
             {
                 // scrivo le due soluzioni
                 Console.WriteLine("x1 = " + e.DammiSoluzione1());
                 Console.WriteLine("x2 = " + e.DammiSoluzione2());
             }
        }
    }
}
```

Esecuzione del programma:

dammi il coefficiente a

1

dammi il coefficiente b

0

dammi il coefficiente c

-4

1x^2 +0x -4 = 0

x1 =-2

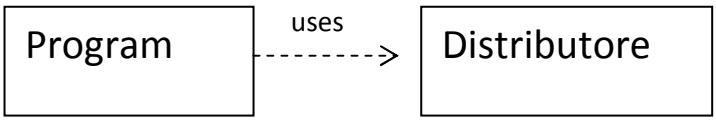
x2 = 2

ESERCIZIO

Programmare il distributore automatico di bibite.

Suggerimenti:

Il diagramma delle classi:



campi della classe Distributore:

- importo inserito
- prezzo del caffè
- prezzo del the
- prezzo della cioccolata
- resto da restituire

metodi della classe Distributore:

- Inserisci(valore_della_moneta) // ragioniamo in cent
- AumentaPrezzo(incremento) // aumenta il prezzo di tutte le bibite
- DammiImportoInserito()
- FammiIlCaffè() // se i soldi bastano restituisce un messaggio positivo
  // altrimenti ….
- FammiIlThe()
- FammiLaCioccolata()
- RestituisciIlResto()

Soluzione

```
public class Distributore
{
   // campi = sono la memoria interna al distributore
    private int importo;  // il totale inserito
    private int caffè; // prezzo del caffè
    private int resto;  // da restituire dopo aver consumato

   // costruttore
   public Distributore(int unPrezzo)
   {
        importo = 0;
        caffè = unPrezzo;
        resto = 0;
   }
```

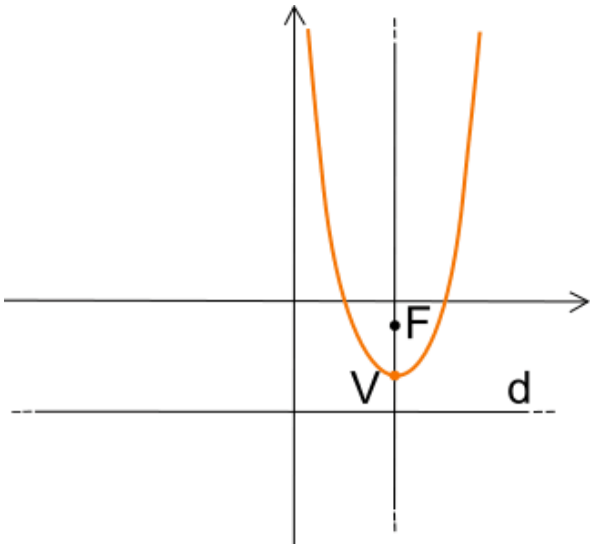
```
  // metodi
  public void AumentaPrezzo(int incremento)
  {
     caffè = caffè + incremento; // nuovo_valore = vecchio_valore + incremento
  }

  public void InserisciMoneta(int valore)
  {
     importo = importo + valore;
  }

  public int DammiImportoInserito()
  {
     return importo;
  }

  public int RestituisciResto()
  {
     // serve una variabile di appoggio per poter restituire il resto
     // e anche azzerarlo
     int appo = resto;
     resto = 0;
     return appo;
   }

   public string FammiIlCaffè()
  {
     if(importo > caffè)
    {
       resto = importo – caffè;
       importo = 0;
       return “preleva il caffè”;
    }
    else
    {
       return “mancano ” + (caffè – importo);
    }
  }

}
```

La finestra di controllo immediato consente di provare in modo interattivo ad interagire con la classe Distributore; si può creare un oggetto di tipo Distributore e poi provare a chiamarne i metodi:

Finestra di controllo immediato

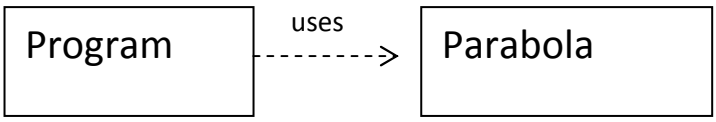
```
Distributore d = new Distributore(35);
L'espressione è stata valutata e non contiene alcun valore
d.FammiIlCaffè()
"Mancano 35"
d.InserisciMoneta(50)
L'espressione è stata valutata e non contiene alcun valore
d.FammiIlCaffè()
"Beviti il caffè..."
d.DimmiResto()
15
d.DimmiImportoInserito()
0
d.RestituisciResto()
15
d.FammiIlCaffè()
"Mancano 35"
```

Esercizio Parabola




```
Y = ax^2 +bx + c

// campi
i coefficienti a, b, c
// metodi
Calcola y data la x
Dimmi la concavità della parabola
Calcola la x del vertice (=-b/2a)
Calcola la y del vertice
Dimmi l'equazione della parabola
```

Il diagramma delle classi:




```
public class Parabola
{
  // campi
  double a;
  double b;
  double c;

  // costruttore    --- ipotesi ca ≠ 0
  public Parabola(double ca, double cb, double cc)
  {
      a = ca; b = cb; c = cc;
  }

  // metodi
  public double CalcolaY(double x)
  {
     return a*x*x+b*x+c;
  }

public string DimmiLaConcavità()
{   // per ipotesi a ≠ 0
    if (a > 0) { return "verso l'alto"; } else { return "verso il basso"; }
}
```

// creazione di un oggetto di tipo Parabola
// di coefficienti 1, 2, 4

Parabola p1 = new Parabola(1, 2, 4);

```
public double CalcolaXVertice()
{
    return –b/(2*a);
}

public double CalcolaYVertice()
{
    // uso una variabile di appoggio
    // e utilizzo metodi già presenti nella classe
    double v = CalcolaXVertice();
    return CalcolaY(v);
}

public string DimmiEquazione()
{
     // ritorna ad eesmpio  “y = 3x^2-4x+5”
     // si ricorda che i numeri positivi vengono scritti senza segno!
     string segnob;
     string segnoc;
     if (b >= 0) { segnob = “+” } else { segnob = ““};
     if (c >= 0) { segnoc = “+” } else { segnoc = ““};
     return “y = “ + a + “x\u00B2” + segnob + b + “x” + segnoc + c;
     // \u00B2 è la codifica UNICODE dell’esponente 2
}
}
```

----------VERIFICA LE TUE COMPETENZE----------

1) Viene dato il seguente codice della classe Palla.

⇨ Riempire i buchi con le istruzioni mancanti

```
public class Palla
{
    // campi
    private double raggio;
    // costruttore
    public Palla(double unRaggio)
    { _____________________________  }   // (1) assegna l'input
    // metodi
    public void Gonfia(double fattoreDiScala)
    {  raggio = raggio * fattoreDiScala; }
    public double DimmiRaggio()
    {  _____________________________}   // (2) ritorna il raggio
    public double CalcolaDiametro()
    {  _____________________________}  // (3) ritorna il risulto del calcolo
}
```

⇨ Scrivere l'output prodotto su Console

```
public class Program
{
   public static void Main()
   {
       Palla p1 = new Palla(2);
       p1.Gonfia(3);
       Console.WriteLine(p1.DimmiRaggio());    (4) ___________
       p1.Gonfia(0.5);
       Console.WriteLine(p1.DimmiRaggio());    (5) ___________

       Palla p2 = new Palla(10);
       double d = p2.CalcolaDiametro();
       Console.WriteLine("diametro = " + d);   (6) _______________

       if (p1.DimmiRaggio() > p2.DimmiRaggio())
       {  Console.WriteLine("maggiore"); }
       else if ((p1.DimmiRaggio() == p2.DimmiRaggio())
             {  Console.WriteLine("uguale"); }
             else
             {  Console.WriteLine("minore"); }   }
                         // che cosa scrive?  (7) _________________

 }  }
```

2) (4 punti) Aggiungere alla classe Palla il campo colore di tipo string con il colore della palla:

```
// campi
private double raggio;
```

Modificare opportunamente il costruttore per tener conto del nuovo campo da inizializzare:

Scrivere il metodo che restituisce il colore della palla:

Scrivere il metodo che cambia il colore della palla con un colore fornito in input:

3) Scrivere le istruzioni per (1.5 punti)

- creare una nuova palla di raggio 8 e colore giallo
- e poi cambiare il suo colore in verde
- e poi scrivere su console il raggio e il colore della palla

4) (1 punto) segnare con una X le assegnazioni errate:

```
int n = “ciao”
int x = 20.5
double y = 20
string s = “dica” + 33
```

---------- SOLUZIONE ----------

5) Viene dato il seguente codice della classe Palla.

⇨ Riempire i buchi con le istruzioni mancanti

```
public class Palla
{
    // campi
    private double raggio;
    // costruttore
    public Palla(double unRaggio)
    { ___raggio = unRaggio;_____  }   // (1) assegna l'input
    // metodi
    public void Gonfia(double fattoreDiScala)
    {  raggio = raggio * fattoreDiScala; }
    public double DimmiRaggio()
    {  _____return raggio;_______  }   // (2) ritorna il raggio
    public double CalcolaDiametro()
    {  ____return raggio*2;______  }  // (3) ritorna il risulto del calcolo
}
```

⇨ Scrivere l'output prodotto su Console

```
public class Program
{
   public static void Main()
   {
       Palla p1 = new Palla(2);
       p1.Gonfia(3);
       Console.WriteLine(p1.DimmiRaggio());    (4) ___6_______
       p1.Gonfia(0.5);
       Console.WriteLine(p1.DimmiRaggio());    (5) _____3______

       Palla p2 = new Palla(10);
       double d = p2.CalcolaDiametro();
       Console.WriteLine("diametro = " + d);   (6) ___diametro = 20___

       if (p1.DimmiRaggio() > p2.DimmiRaggio())
       {  Console.WriteLine("maggiore"); }
       else if ((p1.DimmiRaggio() == p2.DimmiRaggio())
             {  Console.WriteLine("uguale"); }
             else
             {  Console.WriteLine("minore"); }   }
                        // che cosa scrive?  (7) ____minore_____

 }  }
```

6) (4 punti) Aggiungere alla classe Palla il campo colore di tipo string con il colore della palla:

```
// campi
private double raggio;
private string colore;
```

Modificare opportunamente il costruttore per tener conto del nuovo campo da inizializzare:

```
public Palla(double unRaggio, string unColore)
{ raggio = unRaggio;  colore = unColore;  }
```

Scrivere il metodo che restituisce il colore della palla:

```
public string DimmiColore()
{  return colore; }
```

Scrivere il metodo che cambia il colore della palla con un colore fornito in input:

```
public void CambiaColore(string nuovo Colore)
{  colore = nuovo Colore; }
```

7) Scrivere le istruzioni per (1.5 punti)

- creare una nuova palla di raggio 8 e colore giallo
- e poi cambiare il suo colore in verde
- e poi scrivere su console il raggio e il colore della palla

```
Palla p2 = new Palla(8, “giallo”);
p2.CambiaColore(“verde”);
Console.WriteLine( p2.DimmiRaggio());
Console.WriteLine( p2.DimmiColore());
```

8) (1 punto) segnare con una X le assegnazioni errate:

```
int n = “ciao”   X  non si può assegnare una stringa ad una variabile int
int x = 20.5     X  non si può assegnare un numero decimale ad una var int
double y = 20    è valido assegnare un intero ad una variabile double
string s = “dica” + 33   si ottiene una concatenazione di stringhe
          perché il numero 33 viene automaticamente convertito in stringa
```