Introduzione al C++

Corso di C++

INFN – LNS

13 Dicembre 2010

Corrado Santoro

Concetto di "Oggetto"

- Proposto del 1966 nell'ambito dell'AI per rappresentare l' "universo del discorso" (ambiente di riferimento per software di ragionamento)
- Un oggetto è caratterizzato da:
  - **Proprietà:** caratteristiche intrinseche dell'oggetto, stato dell'oggetto, etc.
  - **Operazioni:** azioni che posso compiere e che specificano il "comportamento" dell'oggetto.
- E' quindi un dato strutturato, dove:
  - Le proprietà sono variabili associate all'oggetto
  - Le operazioni sono "pezzi di codice" (funzioni) che agiscono sull'oggetto

Esempi di "oggetti"

- Figure geometriche:
  - **Punto, Quadrato, Cerchio, Triangolo**
- Ognuno di essi ha delle "caratteristiche intrinseche"
  - **Punto**: coordinate x, y
  - **Quadrato**: dimensione lato
  - **Cerchio**: raggio
  - **Triangolo**: dimensione lati e/o angoli
- Su ognuno di essi posso fare delle operazioni:
  - **Punto**: spostarlo di un "delta" o verso una nuova posizione assoluta
  - **Quadrato, Cerchio,Triangolo**: cambiare le dimensioni, calcolare il perimetro, calcolare l'aera

Immaginiamo un oggetto "punto"

- Proprietà dell'oggetto (**attributi** dell'oggetto)
  - **X**
  - **Y**
- Operazioni sull'oggetto (**metodi** dell'oggetto)
  - **setXY**: imposta le coordinate
  - **move**: sposta il punto di un delta_x e delta_y
  - **getX**: permette di ottenere la coordinata X
  - **getY**: permette di ottenere la coordinata Y

Implementazione dell'oggetto "punto"

- Occorre dichiarare una **classe**:
  - E' un **tipo** che permette di rappresentare tutti gli oggetti della stessa categoria (cioé tutti "punti")
- In cui specificare:
  - **Attributi**: nome e tipo;
  - **Metodi**: nome e prototipo (parametri, tipo dei parametri, valore di ritorno)
  - **Accessibilità**: chi può usare ed avere accesso ai metodi e attributi
- Occorre poi **implementare** il codice dei metodi.

La classe “Punto”: Dichiarazione

```
class Punto {
      int X;
      int Y;
public:
       void setXY(int newX, int newY);
       void move(int offsetX, int offsetY);
       int getX();
       int getY();
};

...
```

La classe "Punto": Implementazione

```
void Punto::setXY(int newX, int newY)
{
  X = newY;
  Y = newY;
}

void Punto::move(int offsetX, int offsetY)
{
  X += offsetX;
  Y += offsetY;
}

int Punto::getX() { return X; }
int Punto::getY() { return Y; }
```

La classe “Punto”: Utilizzo

```
int main (int argc, char* argv[])
{
  Punto p;
  p.setXY(10, 10);
  printf ("Il punto si trova in (%d,%d)\n",
            p.getX(), p.getY());
  p.move(-30, 50);
  printf ("Ora il punto si trova in (%d,%d)\n",
            p.getX(), p.getY());
  getchar();
}
```

Esercizio

- Definire una classe "Cerchio" con le seguenti caratteristiche:
  - Attributi:
    - Posizione x
    - Posizione y
    - raggio
  - Metodi:
    - Impostazione posizione
    - Impostazione raggio
    - Calcolo circonferenza
    - Calcolo area

I/O in C++

- L'I/O in C++ è migliorato rispetto al C
- Sono presenti delle classi "standard" (stream):
  - cout: (console out)
  - cin: (console in)
- E' possibile utilizzarli:
  - Inviando dati al cout (per stamparli a video)
  - Ricevendo dati dal cin (per leggere da tastiera)

“Hello World” in C++

```
#include <cstdlib>
#include <iostream>

int main (int argc, char* argv[])
{
  std::cout << “Hello, world!” << std::endl;
  system(“PAUSE”);
}
```

```
#include <cstdlib>
#include <iostream>

using namespace std;

int main (int argc, char* argv[])
{
  cout << “Hello, world!” << endl;
  system(“PAUSE”);
}
```

Costruttore e Distruttore

- Ogni oggetto ha attributi che rappresentano il suo "stato" (coordinate, raggio, ...)
- Quando l'oggetto viene creato, occorre inizializzare questi attributi
- Abbiamo usato dei metodi specifici, tuttavia

- ... esiste un metodo "speciale", denominato Costruttore dell'oggetto
- Questo metodo ha lo ***stesso nome*** della classe

La classe "Punto" con il Costruttore

```
class Punto {
      int X;
      int Y;
public:
       Punto(int initial_x, int initial_y);
       void setXY(int newX, int newY);
       void move(int offsetX, int offsetY);
       int getX();
       int getY();
};

Punto::Punto(int initial_x, int initial_y)
{
  X = initial_x;
  Y = initial_y;
}
…
  Punto p(10,10);
```

Distruttore

- Quando una variabile oggetto non è più usata (es. termina la funzione che l'ha dichiarata e utilizzata) essa viene rimossa
- E' possibile indicare un metodo che viene eseguito quando l'oggetto/variabile viene **cancellato**
- Questo metodo speciale è chiamato **distruttore** e si indica con **`~NomeClasse`**

```
class Punto {
      int X;
      int Y;
public:
       Punto(int initial_x, int initial_y);
       ~Punto(int initial_x, int initial_y);
....
};
```

Esercizio

- Modificare la classe “Cerchio”, introducendo l'opportuno costruttore