Object-oriented programming (Programmazione ad oggetti) Parte 4

Liceo G.B. Brocchi – Bassano del Grappa (VI)
Liceo Scientifico – opzione scienze applicate
Giovanni Mazzocchin

Polimorfismo

- **Run-time polymorphism (aka dynamic dispatch, aka run-time dispatch):** la possibilità di definire una funzione in una classe base, ridefinirla in una classe derivata (*overriding*), e fare in modo che venga invocata la funzione ridefinita nella classe derivata quando viene invocata la funzione della classe base
- Il polimorfismo viene realizzato in C++ attraverso l'utilizzo dei metodi **`virtual`**

Polimorfismo

```
class A {
public:
  void f() {
    cout << "A::f" << endl;
  }
  virtual void g() {
    cout << "A::g" << endl;
  }
};
```

Polimorfismo

```
class B: public A {
public:
  void f() {
    cout << "B::f" << endl;
  }
  //overriding
  virtual void g() {
    cout << "B::g" << endl;
  }
};
```

Polimorfismo

```
int main(int argc, char* argv[]) {
  A a;
  B b;
  a.f();
  a.g();
  b.f();
  b.g();
}
```

Eseguire questo codice.
Succede qualcosa di inatteso?

Polimorfismo

```
int main(int argc, char* argv[]) {
  B b;
  A* p = &b;
  p -> f();
  p -> g();
}
```

Eseguire questo codice.
Succede qualcosa di inatteso?

Polimorfismo

```
int main(int argc, char* argv[]) {
  B b;
  A* p = &b;
  p -> f();
  p -> g();
}
```

- il puntatore (polimorfo) `p` ha tipo statico `A*` e tipo dinamico `B*`
- per i metodi `virtual` viene applicato il **dynamic dispatch**, ossia viene invocato il metodo del tipo dinamico
- per i metodi non `virtual`, viene invocato il metodo del tipo statico
- il polimorfismo funziona sui puntatori e sui riferimenti, non sugli oggetti

Classi astratte

```
class animal {
  public:
    virtual void speak() = 0;
};
```

- `speak` è un metodo virtuale puro (si noti `= 0`): significa che andrà overridato in una classe derivata
- una classe contenente almeno un metodo virtuale puro è detta classe astratta
- **non è possibile creare istanze di una classe astratta**
- una classe astratta ha quindi solo il significato di classe base di una gerarchia:
  - in questo esempio, *l'idea astratta di animale generico, che emette un verso non specificato*

Classi astratte

```
class animal {
  public:
    virtual void speak() = 0;
};

class sheep: public animal {
  public:
    void speak() {
      cout << "baa" << endl;
    }
};
```