Complementi di Programmazione

Python: Tipizzazione

CdL Informatica - Università degli studi di Modena e Reggio Emilia
AA 2023/2024

Filippo Muzzini

Tipi in Python

Python è un linguaggio completamente orientato agli oggetti.

Ogni variabile (che ha un tipo) è un oggetto.

Anche i tipi che si potrebbero definire primitive (es. int, float, ecc…) sono classi con i loro metodi e i loro attributi.

Tipi in Python

Python è un linguaggio completamente orientato agli oggetti.

Ogni variabile (che ha un tipo) è un oggetto.

Anche i tipi che si potrebbero definire primitive (es. int, float, ecc…) sono classi con i loro metodi e i loro attributi.

Classi e oggetti

Nei linguaggi ad oggetti, ogni oggetto ha a disposizione:

- I metodi e gli attributi della sua classe
- I metodi e gli attributi ereditati

Classi e oggetti … e polimorfismo

Nei linguaggi ad oggetti, ogni oggetto ha a disposizione:

- I metodi e gli attributi della sua classe
- I metodi e gli attributi ereditati

Tramite l’ereditarietà si possono realizzare comportamenti diversi per uno stesso metodo

Classi e oggetti … e polimorfismo

Tramite l'ereditarietà si possono realizzare comportamenti diversi per uno stesso metodo.

class Moto extends Veicolo

getRoute(): return 2

class Auto extends Veicolo

getRoute(): return 4

Veicolo v = getVeicolo() //ritorno o Auto o Moto

v.getRoute() // risultato?

Classi e oggetti … e polimorfismo

v.getRoute() // risultato?

E’ necessario capire quale metodo effettivo chiamare (da un punto di vista del compilatore/interprete).

- Controllare che la classe abbia tale metodo (e che sia una classe/sottoclasse di Veicolo)
- Se non è così passare alle superclassi alla ricerca del metodo

Classi e oggetti … e polimorfismo

v.getRoute() // risultato?

E’ necessario capire quale metodo effettivo chiamare (da un punto di vista del compilatore/interprete).

- A tempo di compilazione -> più efficiente ma meno flessibile
- A tempo di esecuzione -> meno efficiente ma più flessibile

Duck Typing

v.getRoute() // risultato?

Alternativa: non controllare il tipo!

Controllo solo che l'oggetto abbia tale metodo (non controllo la classe di appartenenza).

Meccanismo utilizzato in Python

Duck Typing

"When I see a bird that walks like a duck and swims like a duck and quacks like a duck, I call that bird a duck"

Duck Typing

```
class Duck:
    def quack(self):
        print("Quaaaaaack!")
class Person:
    def quack(self):
        print("The person imitates a duck.")
```

```
def in_the_farm(a):
    a.quack()
```

Duck Typing

function calcola(a,b,c) => return (a+b)*c

e1 = calcola(1,2,3)

e2 = calcola([1,2,3],[4,5,6],2)

e3 = calcola(‘mele ’, ‘e arance’, 3)

Calcola deve ricevere dei parametri che supportino i metodi + e *

Duck Typing

Risultati

e1 → 9

e2 → [1, 2, 3, 4, 5, 6, 1, 2, 3, 4, 5, 6]

e3 → “mele e arance mele e arance mele e arance”