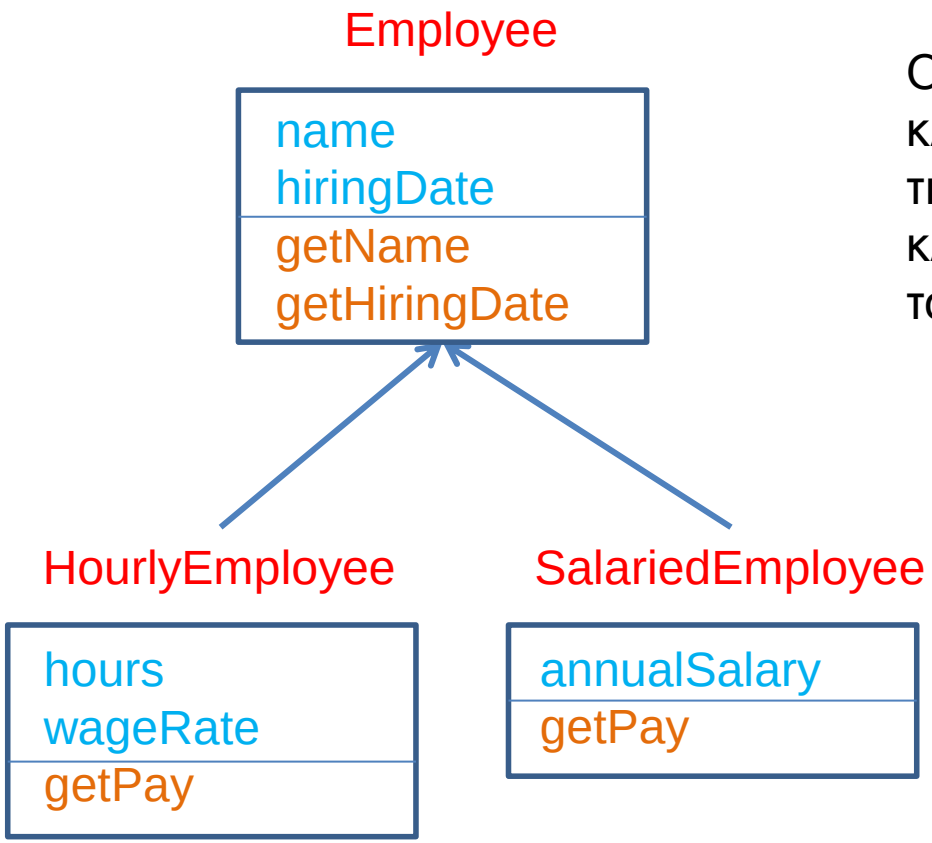
ΤΕΧΝΙΚΕΣ ΑΝΤΙΚΕΙΜΕΝΟΣΤΡΑΦΟΥΣ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ

Πολυμορφισμός – Αφηρημένες κλάσεις

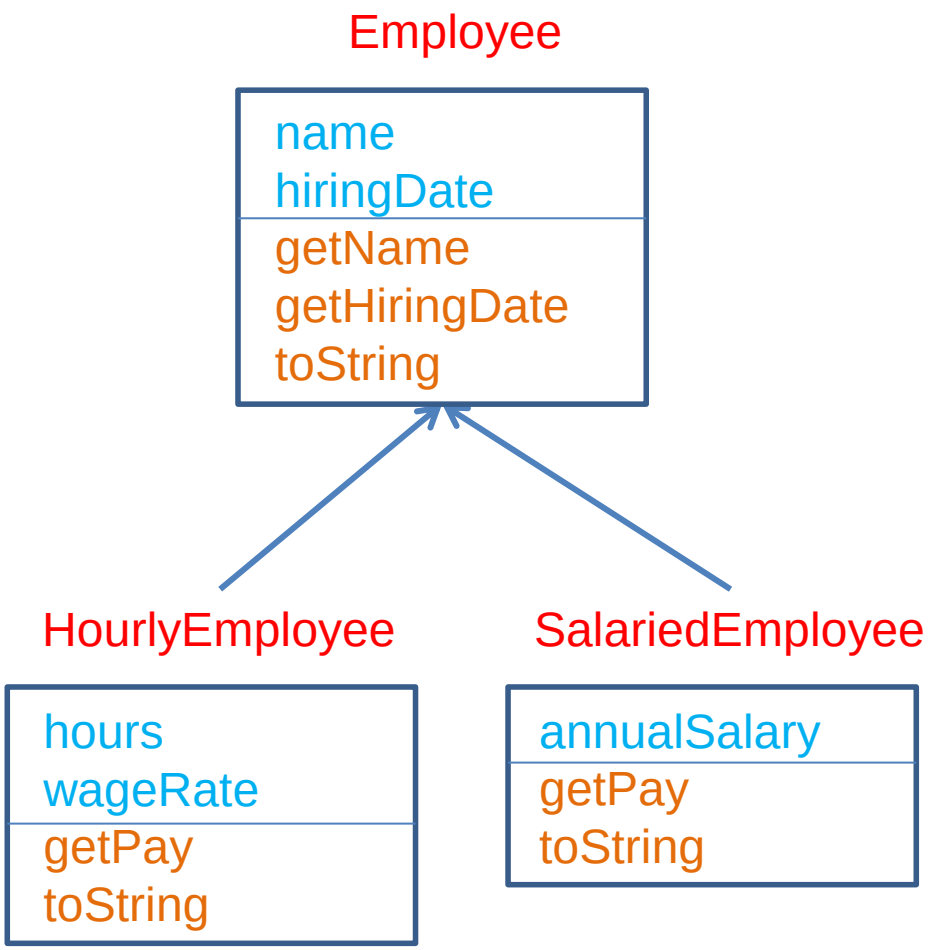
Interfaces (διεπαφές)

Κληρονομικότητα



Οι παράγωγες κλάσεις κληρονομούν τα πεδία και τις μεθόδους της βασικής κλάσης και έχουν και δικά τους πεδία και μεθόδους

Late Binding

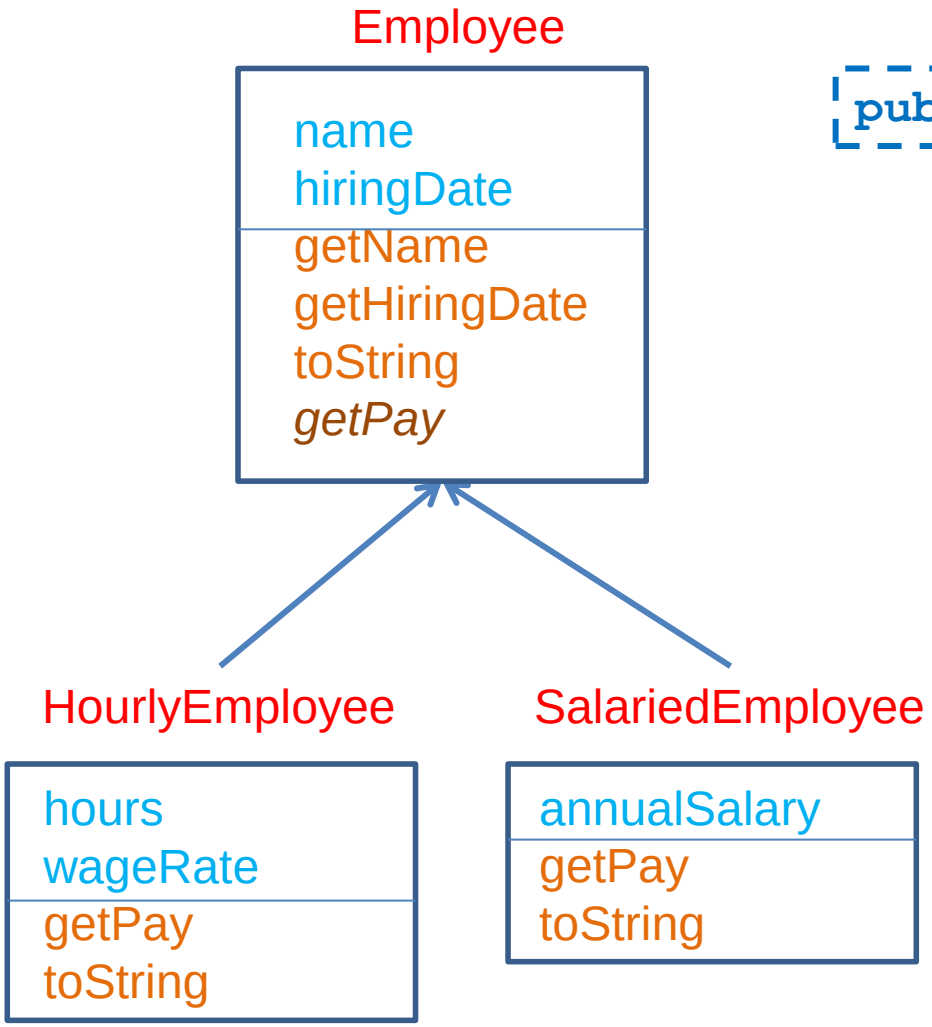



```
Employee e;
e = new HourlyEmployee();
System.out.println(e);
e = new SalariedEmployee();
System.out.println(e);
```

Late Binding:

Ο κώδικας που εκτελείται για την `toString()` εξαρτάται από την κλάση του αντικειμένου την ώρα της κλήσης (HourlyEmployee ή SalariedEmployee) και όχι την ώρα της δήλωσης (Employee)

Αφηρημένες κλάσεις




```
public abstract double getPay();
```

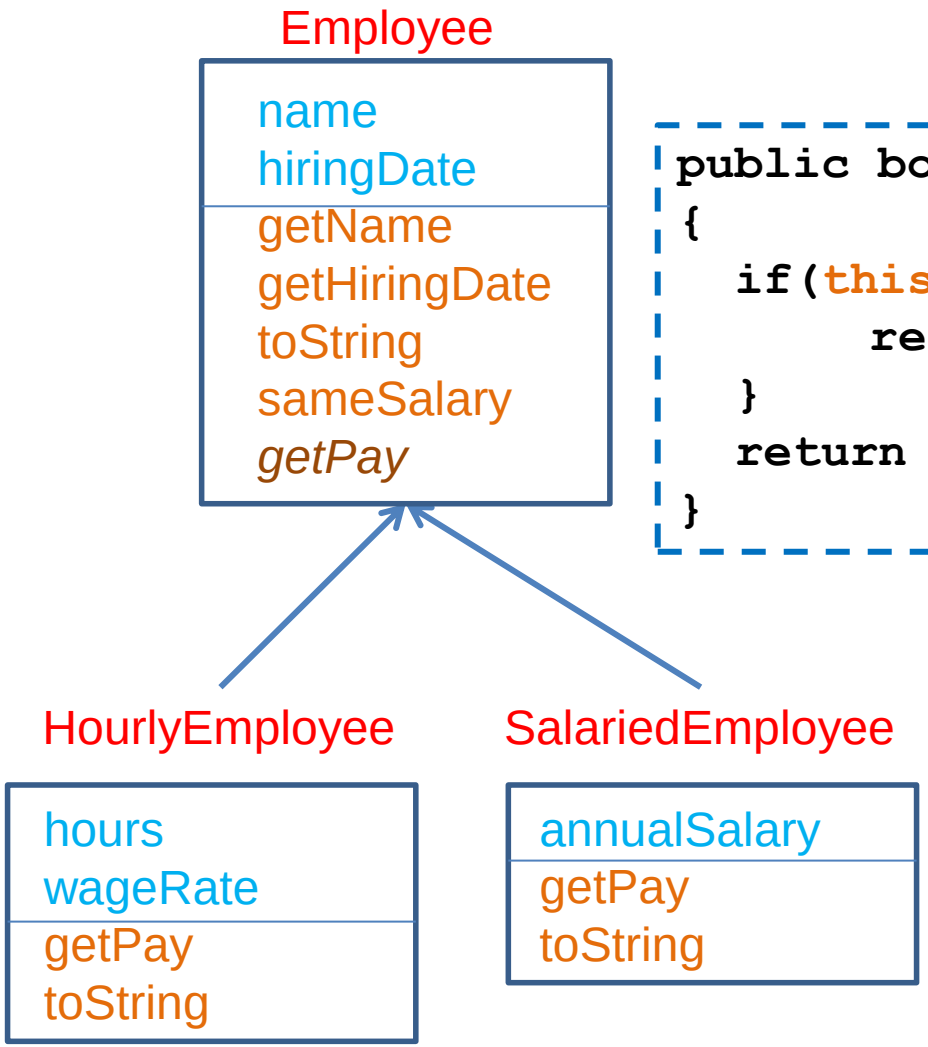
Μια αφηρημένη μέθοδος δηλώνεται σε μια γενική κλάση και ορίζεται σε μια πιο εξειδικευμένη κλάση

Οι κλάσεις με αφηρημένες μεθόδους είναι αφηρημένες κλάσεις.

Δεν μπορούμε να ορίσουμε αντικείμενα αφηρημένων κλάσεων.

Οι παράγωγες ενυπόστατες κλάσεις πρέπει να υλοποιούν τις αφηρημένες μεθόδους.

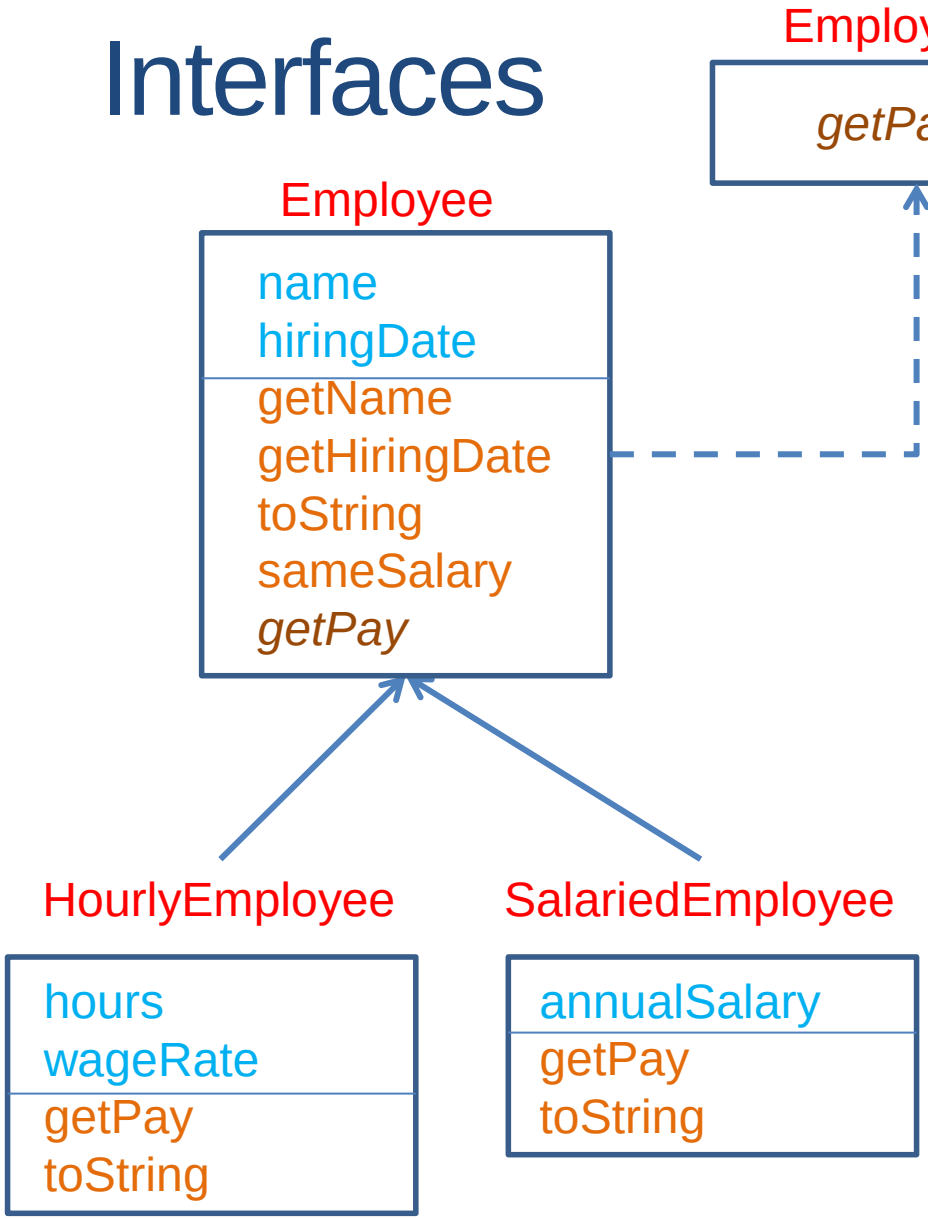
Αφηρημένες κλάσεις




```
public boolean sameSalary(Employee other)
{
   if(this.getPay() == other.getPay()){
        return true;
   }
   return false
}
```

Μια αφηρημένη μέθοδος μπορεί να χρησιμοποιηθεί μέσα σε άλλες μεθόδους της αφηρημένης κλάσης

Interfaces




```
public interface EmployeePay
{
   public double getPay();
}

public abstract Employee
        implements EmployeePay
```

Ένα interface ορίζει μια βασική λειτουργικότητα (μεθόδους).

Μία κλάση υλοποιεί το interface, δηλ. υλοποιεί τις μεθόδους του interface.

Μια κλάση μπορεί να υλοποιεί παραπάνω από ένα interfaces

Παράδειγμα: Το interface Comparable

- Το interface Comparable είναι ένα υπάρχον interface το οποίο ορίζει διεπαφή για αντικείμενα τα οποία μπορούν να συγκριθούν μεταξύ τους
- Ορίζει την μέθοδο
  - `public int compareTo(Object other);`
- Σημασιολογία:
  - Αν η μέθοδος επιστρέψει αρνητικό αριθμό τότε το αντικείμενο this είναι μικρότερο από το αντικείμενο other
  - Αν η μέθοδος επιστρέψει μηδέν τότε το αντικείμενο this είναι ίσο με το αντικείμενο other
  - Αν η μέθοδος επιστρέψει θετικό αριθμό τότε το αντικείμενο this είναι μεγαλύτερο από το αντικείμενο other

Εφαρμογή

- Μπορούμε να ορίσουμε μια μέθοδο sort η οποία να μπορεί να εφαρμοστεί σε πίνακες με οποιαδήποτε μορφής αντικείμενα

```
public static void sort(Comparable[] array){
  for (int i = 0; i < array.length; i ++){
    Comparable minElement = array[i];
    for (int j = i+1; j < array.length; j ++){
      if (minElement.compareTo(array[j]) > 0){
       minElement = array[j];
       array[j] = array[i];
       array[i] = minElement;
      }
    }
  }
}
```

```
import java.util.Scanner;

class Person implements Comparable
{
  private String name;
  private int number;

  public Person(){
        System.out.println("enter name and number:");
        Scanner input = new Scanner(System.in);
        name = input.next(); number = input.nextInt();
  }

  public String toString(){
        return name + " " + number;
  }

  public int compareTo(Object other){
        Person otherPerson = (Person) other;
        if (number < otherPerson.number){
                return -1;
        }else if (number == otherPerson.number){
                return 0;
        } else { return 1;}
  }
}
```

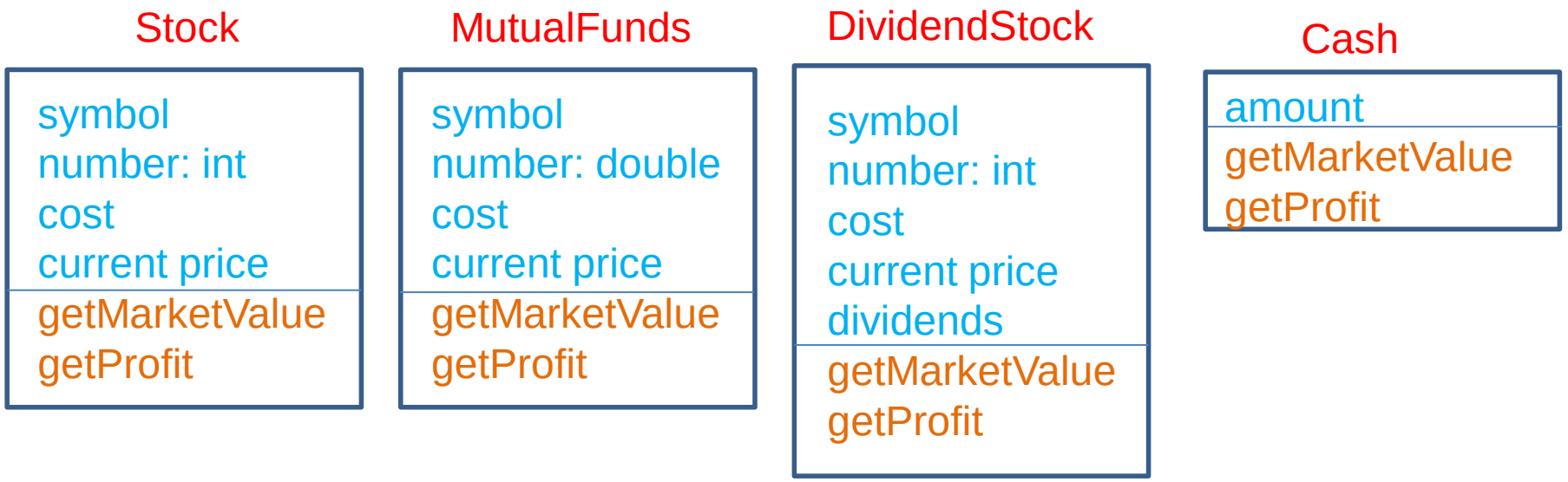
```
public class ComparableExample
{
  public static void main(String[] args){
    Person[] array = new Person[5];
    for (int i = 0; i < array.length; i ++){
        array[i] = new Person();
    }
    sort(array);
    System.out.println();
    for (int i = 0; i < array.length; i ++){
        System.out.println(array[i]);
    }
  }

  public static void sort(Comparable[] array){
    for (int i = 0; i < array.length; i ++){
        Comparable minElement = array[i];
        for (int j = i+1; j < array.length; j ++){
          if (minElement.compareTo(array[j]) > 0){
                minElement = array[j];
                array[j] = array[i];
                array[i] = minElement;
          }
        }
    }
  }
}
```

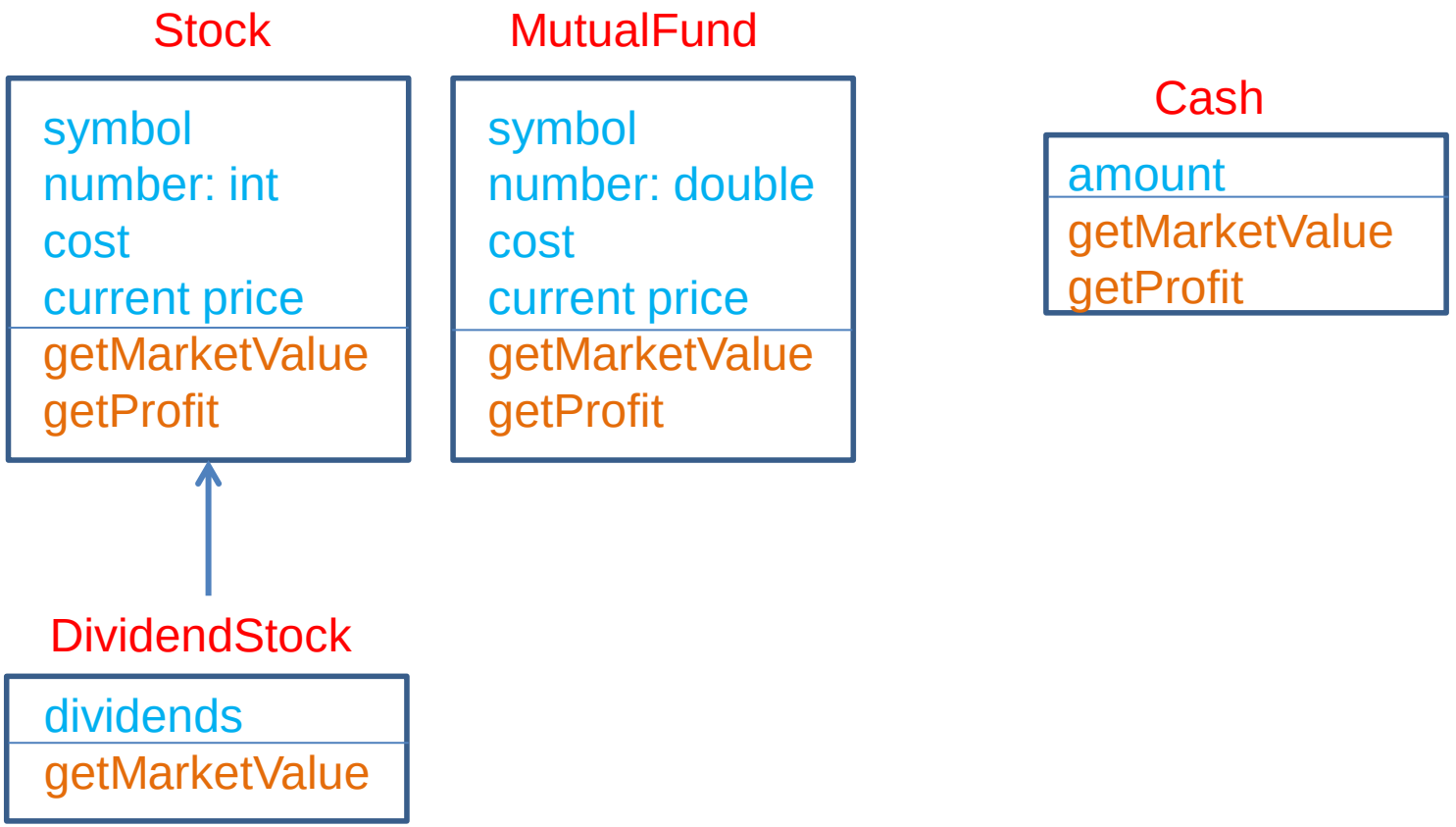
Ένα μεγάλο παράδειγμα

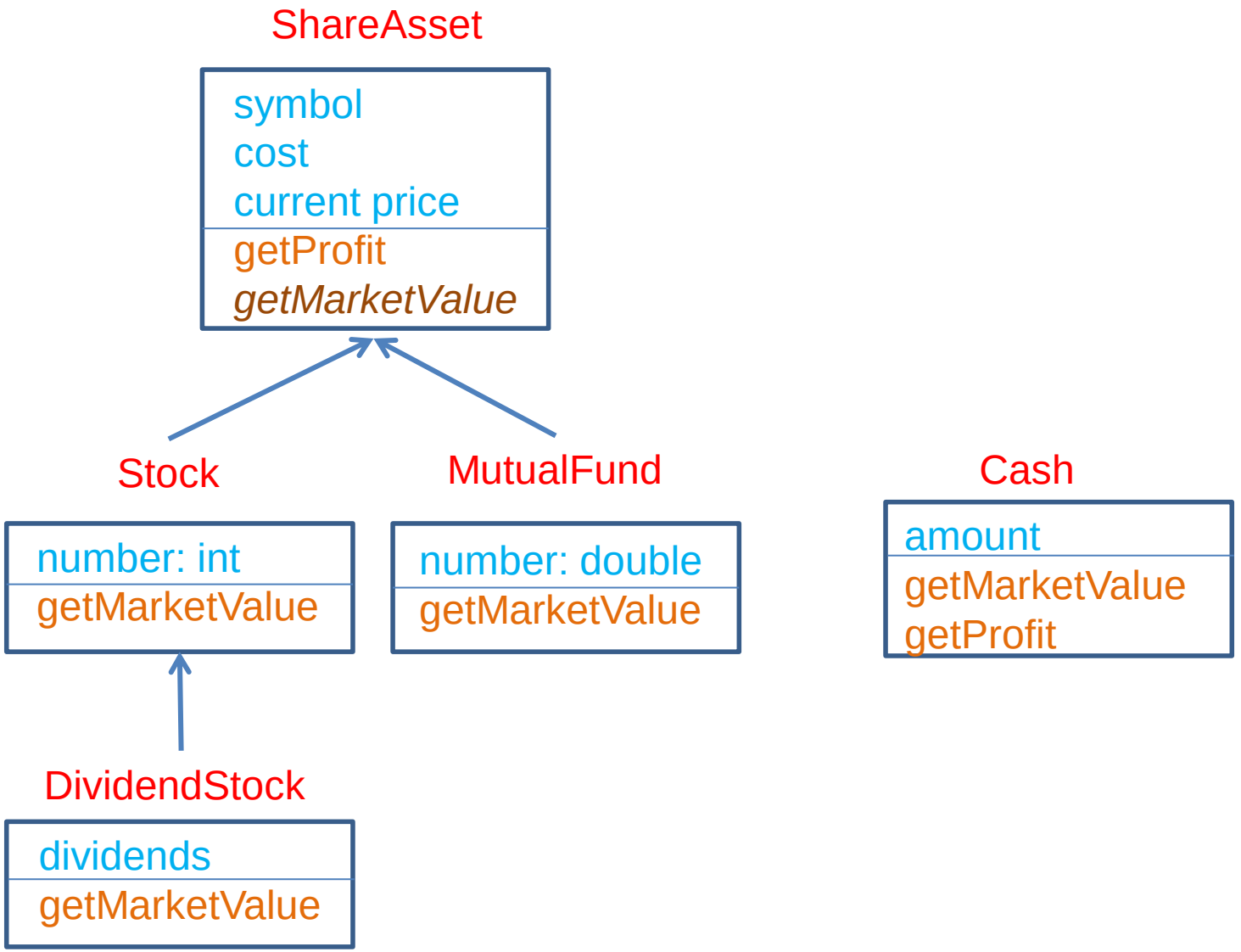
- Θέλουμε να φτιάξουμε ένα πρόγραμμα που διαχειρίζεται το πορτοφόλιο (portofolio) ενός χρηματιστή. Το portofolio έχει μετοχές (stocks), μετοχές που δίνουν μέρισμα (divident stocks), αμοιβαία κεφάλαια (mutual funds), και χρήματα (cash). Για κάθε μια από αυτές τις αξίες (assets) θέλουμε να υπολογίζουμε την τωρινή της αποτίμηση (market value) και το κέρδος (profit) που μας δίνει. Μετά θέλουμε να υπολογίσουμε τη συνολική αξία του πορτοφόλιου και το συνολικό κέρδος

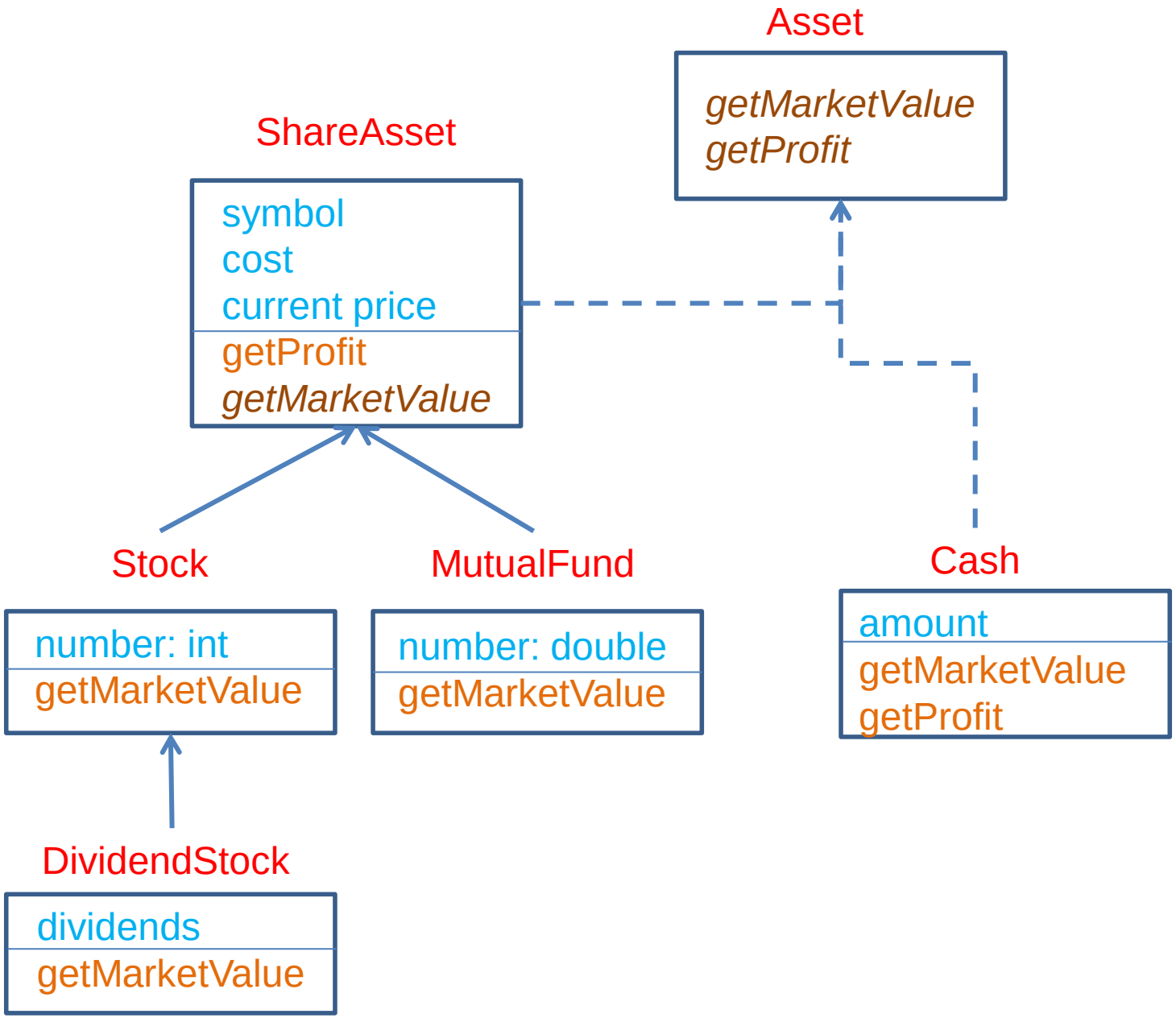
Λεπτομέρειες

- Cash: Δεν μεταβάλλεται η αξία του, δεν έχει κέρδος
- Stocks: Η αξία του είναι ίση με τον αριθμό των μετοχών επί την αξία της μετοχής. Το κέρδος είναι η διαφορά της τωρινής αποτίμησης με το κόστος αγοράς
- Mutual Funds: Παρόμοια με τα Stocks αλλά ο αριθμός των μετοχών που μπορούμε να έχουμε είναι πραγματικός αριθμός αντί για ακέραιος
- Dividend Stocks: Όμοια με τα Stocks αλλά στο κέρδος προσθέτουμε και τα μερίσματα


Stock
symbol
number: int
cost
current price
getMarketValue
getProfit
MutualFunds
symbol
number: double
cost
current price
getMarketValue
getProfit
DividendStock
symbol
number: int
cost
current price
dividends
getMarketValue
getProfit
Cash
amount
getMarketValue
getProfit


Stock
symbol
number: int
cost
current price
getMarketValue
getProfit
MutualFund
symbol
number: double
cost
current price
getMarketValue
getProfit
Cash
amount
getMarketValue
getProfit
DividendStock
dividends
getMarketValue


ShareAsset
symbol
cost
current price
getProfit
getMarketValue
Stock
MutualFund
Cash
number: int
getMarketValue
number: double
getMarketValue
amount
getMarketValue
getProfit
DividendStock
dividends
getMarketValue

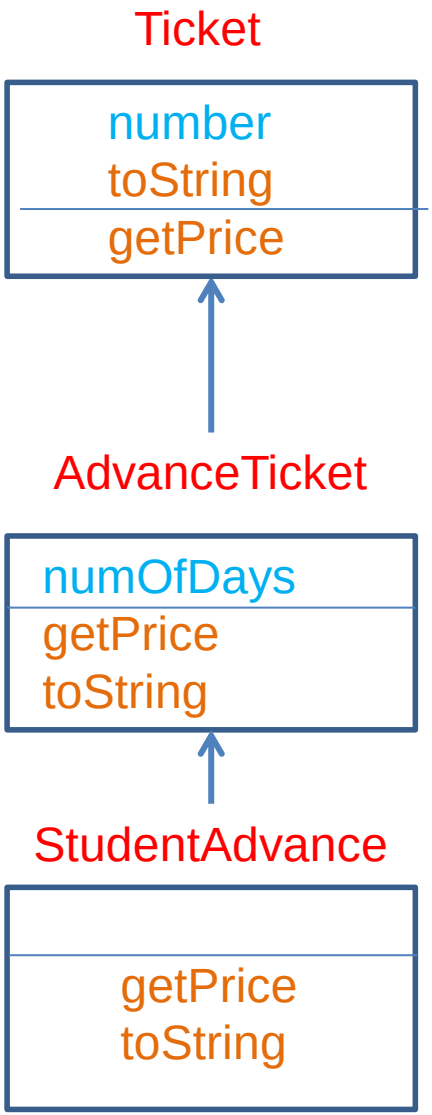

Asset
getMarketValue
getProfit
ShareAsset
symbol
cost
current price
getProfit
getMarketValue
Stock
number: int
getMarketValue
MutualFund
number: double
getMarketValue
Cash
amount
getMarketValue
getProfit
DividendStock
dividends
getMarketValue

Άλλο ένα παράδειγμα

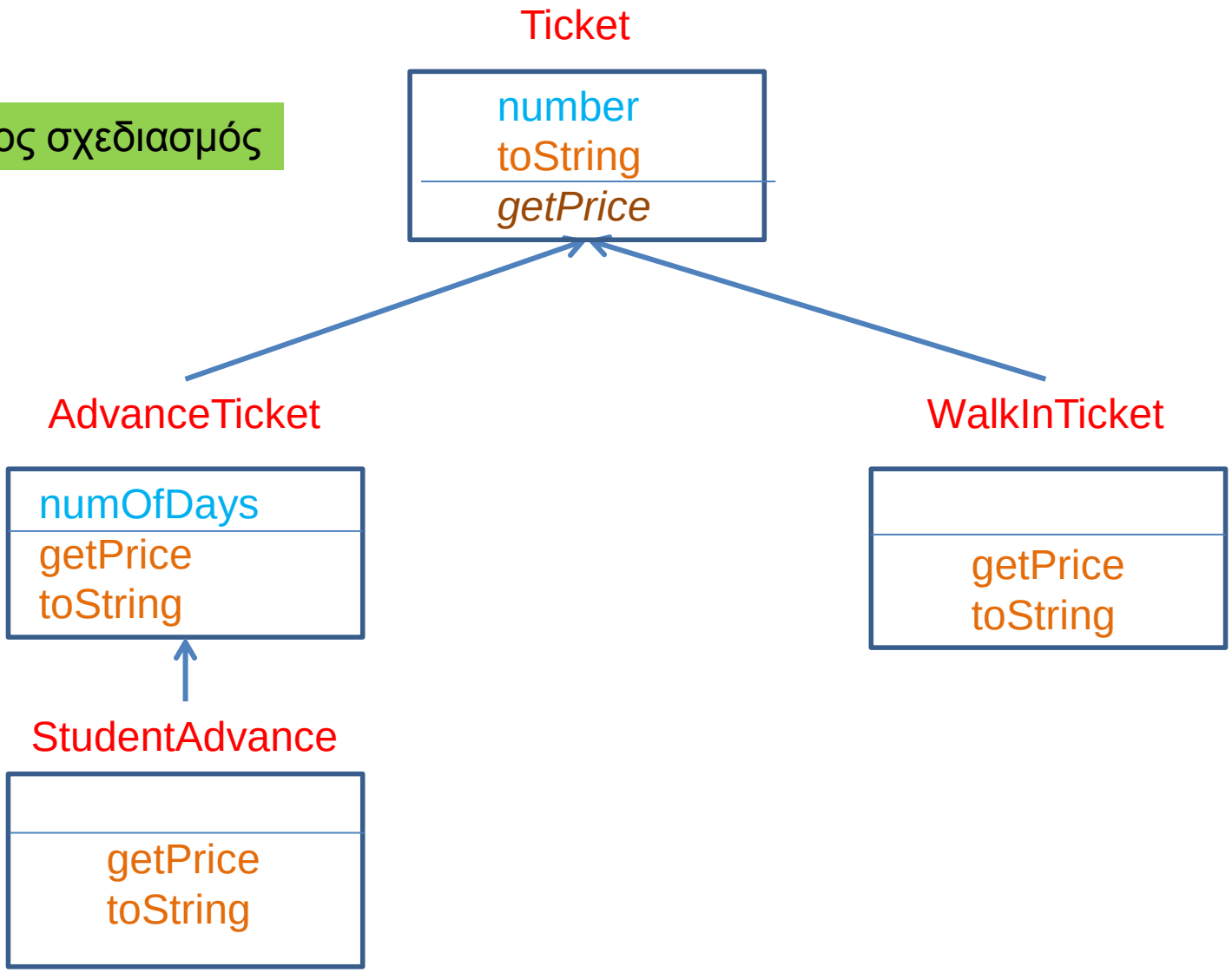
- Έχουμε ένα σύστημα διαχείρισης εισιτηρίων μιας συναυλίας. Το κάθε εισιτήριο έχει ένα νούμερο και τιμή. Η τιμή του εισιτηρίου εξαρτάται αν θα αγοραστεί στην είσοδο (50 ευρώ), ή θα αγοραστεί μέχρι και 10 μέρες πριν την συναυλία (40 ευρώ), ή πάνω από 10 μέρες πριν την συναυλία (30 ευρώ). Τα εισιτήρια εκ των προτέρων έχουν φοιτητική έκπτωση 50%.

- Θέλουμε να τυπώσουμε τα εισιτήρια και να υπολογίσουμε τα συνολικά έσοδα της συναυλίας.

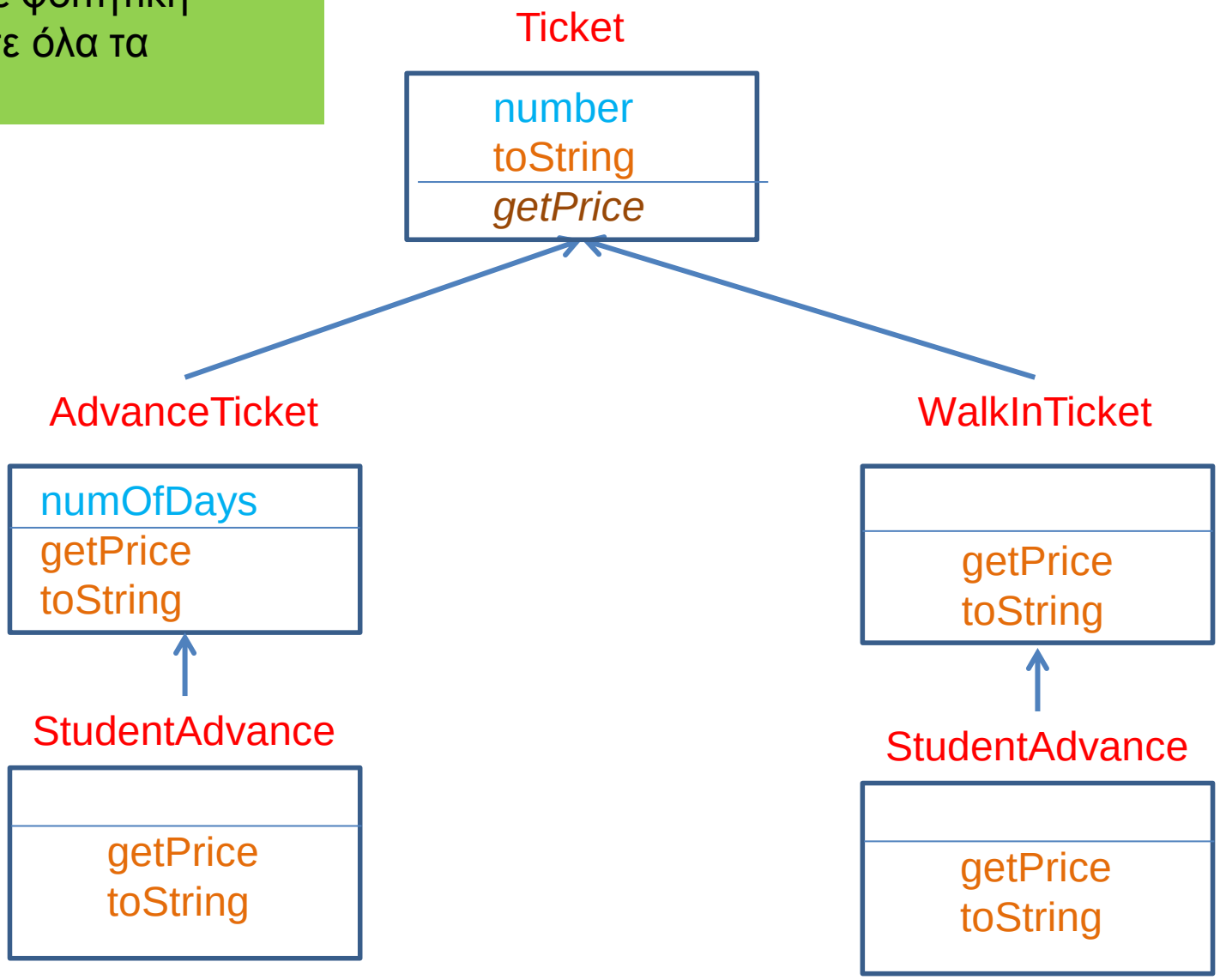
Ένας σχεδιασμός



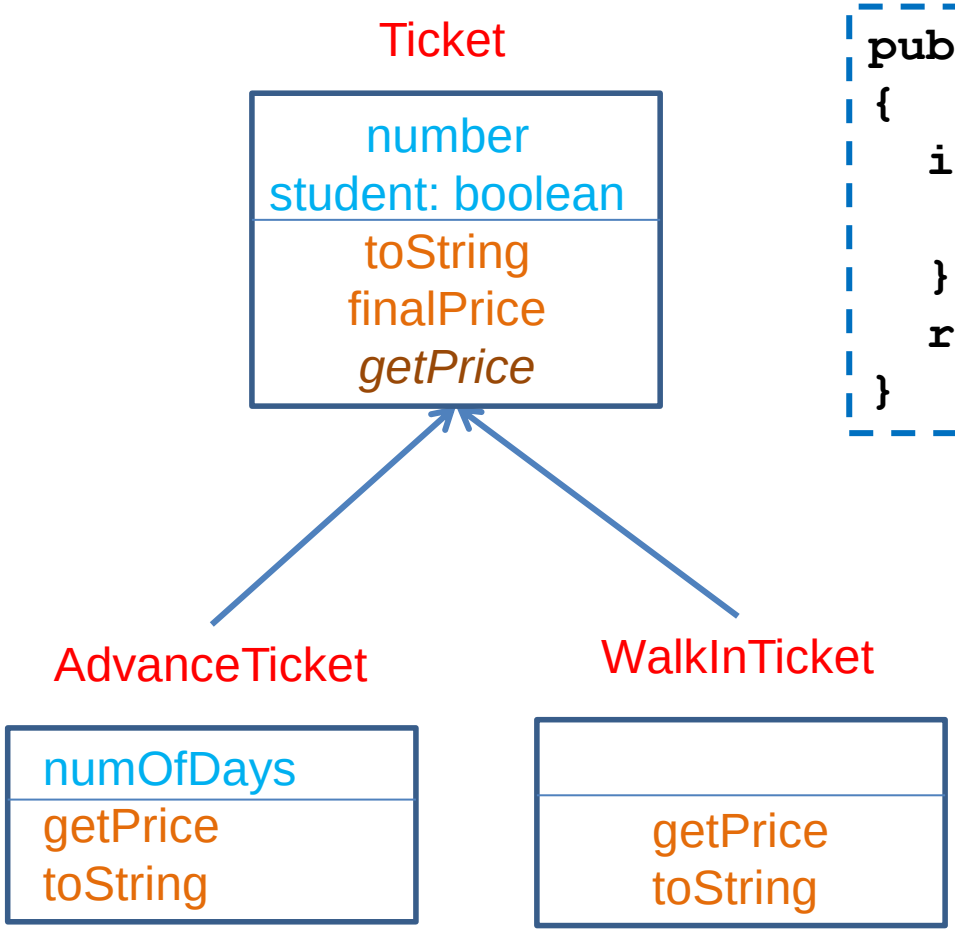
Ένας άλλος σχεδιασμός



Αν θέλουμε φοιτητική έκπτωση σε όλα τα εισιτήρια?



Αν θέλουμε φοιτητική έκπτωση σε όλα τα εισιτήρια?




```
public double finalPrice()
{
   if (student){
        return getPrice()*0.5;
   }
   return getPrice();
}
```