Programowanie funkcyjne i współbieżne
Lista 5

1. Zdefiniuj swoją klasę dla modyfikowalnej pary polimorficznej MyPair[A, B]. Ma ona mieć dwa pola modyfikowalne fst i snd z odpowiednimi akcesorami i mutatorami, oraz metodę toString, zwracającą napis w formacie (*fst, snd*).

2. Klasa BankAccount jest zdefiniowana następująco:

```
class BankAccount(initialBalance : Double) {
  private[this] var balance = initialBalance
  def checkBalance = balance
  def deposit(amount : Double) = { balance += amount; balance}
  def withdraw(amount : Double) = { balance -= amount; balance}
  override def toString = "%.2f".format(balance)
}
```

W podpunktach a) i b) należy utworzyć odpowiednie konto i przeprowadzić na nim testy.

a) Zdefiniuj klasę CheckingAccount, rozszerzającą klasę BankAccount, w której pobierana jest opłata w wysokości 1$ za każdą wpłatę i wypłatę z konta. Zmodyfikuj odpowiednio metody deposit i withdraw.

b) Zdefiniuj klasę SavingsAccount, rozszerzającą klasę BankAccount, w której co miesiąc do konta dodawane jest oprocentowanie 1%, a w przypadku debetu pobierana jest opłata w wysokości 1% . Nie chodzi tu o korzystanie z kalendarza. Dodaj metodę earnMonthlyInterest: Unit; jej użycie oznacza, że upłynął miesiąc.
Trzy transakcje miesięcznie są bezpłatne, za pozostałe pobierana jest opłata w wysokości 1$.
Zmodyfikuj odpowiednio metody deposit i withdraw.

3.
a) Zdefiniuj abstrakcyjną klasę Zwierz, z dokładnie jednym niemodyfikowalnym polem imie. Jej konstruktor(y) ma(ją) umożliwiać tworzenie zwierzęcia z domyślnym imieniem (np. „bez imienia") lub z imieniem, podanym jako argument.
Klasa ma mieć cztery publiczne bezargumentowe metody: imie, rodzaj, dajGlos i toString, które odpowiednio zwracają: imię, rodzaj i charakterystyczny głos zwierzęcia.
Metoda toString ma zwracać informację o zwierzęciu w poniższym formacie:
<rodzaj zwierzęcia> <imię zwierzęcia> daje głos <charakterystyczny głos zwierzęcia>!
Na przykład (dla psa):
Pies Kruczek daje głos Hau, hau!
Zdecyduj, które metody muszą być abstrakcyjne.

b) Zdefiniuj klasy publiczne dla kilku rodzajów zwierząt (co najmniej dwóch), dziedziczące z klasy Zwierz. Podobnie jak klasa Zwierz mają one umożliwiać tworzenie zwierzęcia na dwa sposoby: bez podanego imienia i z imieniem. **Nie wolno dodawać żadnych nowych pól ani metod!** Które metody trzeba zdefiniować, które zastąpić, a które odziedziczyć bez zmian?

c) Napisz program testujący TestZwierza (jako obiekt singletonowy), w którym należy utworzyć kolekcję kilku zwierząt (użyj kolekcji Vector) i wypisać informacje o tych zwierzętach (wykorzystaj wyrażenie for). Zaobserwuj efekt działania polimorfizmu inkluzyjnego i wiązania dynamicznego.
W arkuszu elektronicznym (worksheet) program uruchamia się tak: TestZwierza.main(Array())

UWAGA. Klasy mają posiadać wyłącznie pola i metody opisane w specyfikacji. Proszę ściśle przestrzegać powyższej specyfikacji. Jest ona celowo bardzo restrykcyjna i dość sztuczna z powodów dydaktycznych. Chodzi o wymuszenie użycia pewnych konstrukcji języka Scala.