Paradygmaty programowania - ćwiczenia
Lista 9

Na wykładzie 9 pokazano, jak wygląda kompletny program w Scali. W najprostszym przypadku jest to autonomiczny obiekt singletonowy (który można wykorzystywać jako moduł) z metodą main: Array[String] => Unit.
Zwykle kompilujemy i uruchamiamy programy za pomocą wygodnych narzędzi, np. sbt bądź jeszcze wygodniejszych IDE, np. IntelliJ. Zawsze warto jednak wiedzieć, co się dzieje „pod podszewką” i umieć to zrobić z wiersza poleceń (z okna terminala).
Załóżmy, że napisaliśmy poniższy program, i że w oknie terminala przeszliśmy do folderu, zawierającego plik MyModule.scala.

```
// Komentarz
/* Inny komentarz */
/** Komentarz dokumentacyjny */
object MyModule:
  def main(args: Array[String]): Unit =
      println("My program runs!")
```

Kompilacja i uruchomienie programu wygląda następująco:

```
> scalac MyModule.scala

> scala MyModule
My program runs!
```

W odróżnieniu od wymagań języka Java, plik źródłowy w Scali (z rozszerzeniem .scala) może zawierać dowolną liczbę definicji klas i obiektów z dowolnymi nazwami. Po skompilowaniu każda klasa i obiekt zostaną umieszczone w oddzielnym pliku z odpowiedną nazwą i rozszerzeniem .class.
Można też przekazać do aplikacji scala kod źródłowy, co jest wygodne dla skryptów (zauważ, że nie są tworzone pliki .class).

```
> scala MyModule.scala
My program runs!
```

Można też użyć środowiska interakcyjnego i zrobić to w cyklu REPL (też nie są tworzone pliki .class).

```
scala> :load MyModule.scala
// defined object MyModule

scala> MyModule.main(Array())
My program runs!
```

Paradygmaty programowania - ćwiczenia
Lista 9

Wszystkie programy mają być napisane w języku Scala 3.

1. Napisz klasę Time, w której konstruktor główny pobiera godzinę jako argument. Konstruktor ma zamieniać wartości ujemne na 0. Klasa ma mieć jedno pole modyfikowalne. Kolejne zmiany wartości godziny mają również zamieniać wartości ujemne na 0.

2.
   a) Zdefiniuj klasę Time z dwiema modyfikowalnymi właściwościami hour i minute, oraz metodą before(other: Time): Boolean sprawdzającą, czy zapamiętany moment czasowy poprzedza moment other. Egzemplarz klasy powinien być konstruowany za pomocą new Time(godz, min), np. new Time(21, 15). Sprawdzaj poprawność argumentów, w razie potrzeby zgłaszaj wyjątek IllegalArgumentException. Poprawność zmian czasu też ma być sprawdzana.
   b) Zmień klasę Time z podpunktu a) w taki sposób, żeby czas był pamiętany jako liczba minut, które upłynęły od północy. Klasa ma mieć tylko **jedno** pole. Publiczny interfejs klasy i jej funkcjonalność mają pozostać niezmienione, tzn. obie implementacje klasy mają być pod tym względem nierozróżnialne.

3. Zdefiniuj klasę Pojazd z polami tylko do odczytu dla producenta, modelu, roku produkcji i z polem modyfikowalnym dla numeru rejestracyjnego. Klasa powinna umożliwiać tworzenie nowych obiektów na cztery sposoby. Zawsze należy podawać nazwę producenta i modelu. Rok produkcji i numer rejestracyjny są opcjonalne. Domyślną wartością dla roku produkcji jest -1, a dla numeru rejestracyjnego napis pusty. Który konstruktor powinien być główny i dlaczego?

4. Przepisz program z wykładu 8, str. 39 (wyjątki) w języku Scala i uruchom go. Wyjaśnij (ustnie) komunikaty (tylko własne metody ze szczytu stosu).