23 maja 2019

Programowanie obiektowe

Lista 10.

Poniższa lista zadań jest do zrobienia w Ruby. Każde zadanie to 4 punkty. Wykonaj dwa z tych zadań.

Zadanie 1. Algorytmy sortowania kolekcji wymagają kilku elementarnych operacji na kolekcji takich jak `swap(i, j)` zamieniające miejscami elementy, `length()` zwracająca długość kolekcji, czy `get(i)` zwracająca i-ty element kolekcji.

Zaimplementuj dwie klasy:

1. ***Kolekcja*** implementująca dowolną kolekcję wraz z wyżej wymienionymi metodami (ew. jeszcze dodatkowe jeśli będzie taka potrzeba);
2. ***Sortowanie*** implementującą dwa wybrane algorytmy (lub *strategie*) sortowania kolekcji. Klasa ta powinna implementować metody `sort1(kolekcja)`i `sort2(kolekcja)`, których argumentem jest obiekt klasy ***Kolekcja***. Która metoda jest szybsza?

Ważne w tym zadaniu jest to, aby metoda korzystała wyłącznie z metod udostępnianych przez kolekcję i nie odwoływała się w żaden sposób do implementacji ***Kolekcji***.

Czy potrafisz powiedzieć, jaki to *wzorzec projektowy*?

Zadanie 2. Zaprogramuj klasę ***Kolekcja*** implementującą kolekcję przechowującą elementy dowolnego typu w kolejności rosnącej. Lista ta powinna być zaprogramowana jako lista dwukierunkowa. Zaprogramuj również klasę ***Wyszukiwanie*** wraz z dwiema różnymi metodami wyszukiwania elementu w kolekcji. Dobrymi kandydatami są tu: algorytm wyszukiwania binarnego oraz jego wariant: wyszukiwanie interpolacyjne. Podobnie jak w poprzednim zadaniu, kolekcja powinna być argumentem wywołania metody, a metoda powinna być tak napisana, aby nie korzystała ze szczegółów implementacji kolekcji.

Zadanie 3. Zadanie polega na implementacji klasy (a właściwie kilku klas) służącej do zapamiętania w czytelnej postaci chwilowego stanu programu, tj. stanu obiektów istniejących w systemie. Chcemy, aby była możliwość zapisania stanu w pliku zarówno w formacie html, jak i LaTeX'a (zamiennie: w miarę czytelnie sformatowany plik tekstowy).

Zadanie można podzielić na dwie części:

1. analiza stanu programu
2. generowanie sformatowanego tekstu

Zaprogramuj klasę ***Snapshot*** z metodą `run` analizującą stan programu oraz klasy ***FormatHTML*** i ***FormatTXT*** implementujące metody formatujące. Zadanie można zaprogramować na dwa sposoby: albo klasy ***FormatHTML*** i ***FormatTXT*** są podklasami ***Snapshot***, albo obiekty klas ***FormatHTML*** czy ***FormatTXT*** są argumentami metody `run`. Przy oddawaniu programu oceń wady i zalety poszczególnych rozwiązań.

To już ostatnia lista zadań. Na stronie wykładu jest odnośnik do strony z propozycjami projektów programistycznych, ale można oczywiście zaproponować własne.

Marcin Młotkowski