Wzorce Projektowe, lab. 5

Temat: Wykorzystanie wzorca **Composite** oraz **Decorator** do stworzenia konsolowego kalkulatora wyrażeń arytmetycznych zgodnych z notacją polską.

1. Stworzyć interfejs **Wyrazenie**, w którym będzie metoda **wartość()** zwracająca liczbę rzeczywistą. To będzie nasz komponent. Następnie zaimplementować powyższy interfejs w klasie **Liczba**, która będzie naszym liściem ze slajdu nr 2 z pliku composite.pdf. Powinna ona zawierać w sobie prywatne pole **wartość** odpowiadające jej faktycznej wartości liczbowej. W konstruktorze powinna przyjmować ona wartość typu **double** i zwracać ją metodą **wartość()**.
2. Dodać klasy **Suma**, **Roznica**, **Iloczyn** i **Iloraz**, które będą zawierały w sobie prywatne pola **a** i **b** typu **Wyrazenie** oraz będą wykonywały na nich odpowiednie operacje poprzez metodę **wartość()**, np. **Suma** powinna zwracać **a.wartosc() + b.wartosc()** (oczywiście jako double). Podobny przypadek pojawia się na slajdach 11 i 12 z pliku composite.pdf. Zakładamy, że wszystkie operacje arytmetyczne w naszym programie na ten moment będą dwuargumentowe. W przypadku dzielenia należy też obsłużyć sytuację, w której drugie wyrażenie będzie zerem, poprzez rzucenie wyjątku.
3. Przetestować działanie programu poprzez stworzenie w funkcji **main()** przykładowego wyrażenia:

```
Wyrazenie w1 = new Roznica(new Suma(new Liczba(10), new Iloraz(new Iloczyn(new Liczba(2),
new Liczba(15)), new Liczba(3))), new Liczba(4));
System.out.println(w1.wartosc());
```

   Poprawny wynik powinien wynosić 16.0.
4. Dodać klasę **Podwojenie** implementującą interfejs **Wyrazenie**. Zgodnie z wzorcem projektowym **Decorator** powinna ona przechowywać w sobie obiekt jakiejś konkretnej klasy implementującej **Wyrazenie**, który będzie obudowywać/”dekorować”. Należy przekazać go jej już w konstruktorze (slajd 5 z decorator.pdf). Ta klasa powinna poprzez metodę **wartość()** podwajać wartość zwracaną przez zawierany w sobie komponent.
5. Prawdziwy programista nie może zostawić swojej aplikacji w takim stanie! Trzeba dopisać parser, który wczyta z konsoli **String** zawierający działanie zgodne z notacją polską (najpierw operator działania arytmetycznego, a potem argumenty), następnie przetworzy je i zwróci jego wartość. Każdemu z zaimplementowanych wcześniej działań należałoby wtedy przyporządkować określony znak: +, -, *, / oraz dowolnie wybrany dla **Podwojenia**, np. „^”.
6. Sposób zaimplementowania parsera jest dowolny, jednak warto wykorzystać funkcję **split()** do podzielenia wejściowego **Stringa** na tokeny oraz odwrócić kolejność zapisanych wyrażeń. Dzięki temu stosując stos możemy wrzucać na niego kolejno wczytywane **Liczby** (**Wyrazenia**), a gdy pojawi się operator działania, zdjąć dwie z góry (lub jedno dla **Podwojenia**) i wrzucić nowe wyrażenie, np. **Sumę** dwóch poprzednich. Na końcu na stosie pozostanie tylko jedno rozbudowane **Wyrazenie** w postaci podobnej do tej z pkt. 3. Mając to wystarczy wywołać dla niego metodę **wartość()** i wypisać ją użytkownikowi na konsoli.

UWAGI:

- Więcej o notacji polskiej można poczytać chociażby na Wikipedii polskiej lub angielskiej.
- Przy podawaniu wyrażeń w konsoli wszelkie liczby i operatory powinny być od siebie oddzielone spacjami.
- Dla wyrażenia „`^ + 10 / * 2 15 3`” parser powinien zwrócić wynik `40.0`.