

KURS JĘZYKA C++

KALKULATOR ONP

Instytut Informatyki Uniwersytetu Wrocławskiego

Paweł Rzechonek

Zadanie.

Napisz program interaktywnego kalkulatora. Kalkulator ten powinien interpretować i obliczać wyrażenia zapisane w postaci ONP. Program ma odczytywać polecenia ze standardowego wejścia `cin`, wykonywać obliczenia i wypisywać wyniki na standardowe wyjście `cout`. Wszelkie komentarze i uwagi program ma wysyłać na standardowe wyjście dla błędów `clog`. Dodatkową funkcjonalnością tego kalkulatora ma być możliwość zapamiętywania wyników obliczeń.

Zaprojektuj hierarchię klas, która umożliwi elegancką klasyfikację poszczególnych symboli (abstrakcyjna klasa `symbol`) w wyrażeniu ONP. Wyrażenie to ciąg operandów (klasa `operand`) i operatorów/funkcji (klasa `funkcja`). Operandy to liczby (klasa `liczba` pamiętająca wartość typu `double`) albo zmienne (klasa `zmienna` z nazwą zmiennej); w klasie `zmienna` umieść *kolekcję asocjacyjną* ze zmiennymi jako niepubliczne pole statyczne (na przykład w obiekcie `map<string,double>` albo `hash_map<string,double>`). Funkcje to przede wszystkim dwuargumentowe operatory dodawania, odejmowania, mnożenia i dzielenia; należy też zaimplementować funkcje dwuargumentowe `min`, `max`, `log` i `pow`, jednoargumentowe `abs`, `sgn`, `floor`, `ceil`, `frac`, `sin`, `cos`, `atan`, `acot`, `ln` i `exp` oraz funkcje bezargumentowe (pełniące rolę stałych) `e` i `pi`.

Symbole występujące w wyrażeniu ONP należy najpierw sparsować, potem utworzyć dla nich obiekty, a na koniec umieścić je w *kolekcji sekwencyjnej* (na przykład w obiekcie `vector<token>` albo `list<token>`). Pamiętaj o klasie `token` opakowującej symbole różnych typów, które mogą się pojawić w wyrażeniu (nie można umieszczać klas pochonych w kolekcji).

Program kalkulatora ma pracować z użytkownikiem interaktywnie i powinien rozpoznawać trzy rodzaje poleceń:

- `print wyrażenieONP`

Obliczenie wartości wyrażenia `wyrażenieONP` i wypisanie jej na standardowym wyjściu. Wyrażenie będzie zapisane w postaci postfiksowej (*Odwrotna Notacja Polska*). Czytając kolejne tokeny wyrażenia program powinien je zamieniać na konkretne symbole i umieszczać w *kolejce* (klasa `queue<>`). Przy obliczaniu wartości wyrażenia należy się posłużyć *stosem* (klasa `stack<>`).

- `set zm : wyrażenieONP`

Utworzenie nowej zmiennej `zm` i przypisanie jej wartości obliczonego wyrażenia `wyrażenieONP`. Wartość obliczonego wyrażenia należy wypisać na standardowym wyjściu. Jeśli zmienna `zm` była zdefiniowana już wcześniej, to należy tylko zmodyfikować zapisaną w niej wartość.

- `clear`

Usunięcie wszystkich zmiennych zapamiętanych do tej pory w zbiorze zmiennych. Do kolekcji mogą trafiać tylko zmienne o nazwach będących poprawnymi identyfikatorami i różnych od nazw standardowych dla tego programu funkcji.

Jeśli w wyrażeniu ONP zostanie wykryty błąd (nieznana komenda, źle sformułowane wyrażenie, błędna nazwa, błędny literal stałopozycyjny, czy nierozpoznany operator, funkcja lub zmienna) to należy wypisać stosowny komunikat o błędzie, ale nie przerywać działania programu.

Wskazówka.

Wykorzystaj kolekcje zdefiniowane w STL. Po kolekcjach tych poruszaj się za pomocą iteratorów.