Zadane zaliczeniowe nr 1

Jeszcze Jeden Odpluskwiacz

Info.PO.22/23
ver 1.0

W tym zadaniu należy napisać zestaw klas, które reprezentują instrukcje prostego języka programowania *Macchiato*. Programy w tym języku są pojedynczym blokiem (p. opis instrukcji dalej).

Klasy będące rozwiązaniem powinny oczywiście umożliwiać wykonywanie programów. Efektem wykonania programu ma być wykonanie poszczególnych instrukcji (zgodnie z ich opisem dalej) i wypisanie na standardowe wyjście wartości wszystkich zmiennych z głównego bloku programu. W tym zadaniu **nie** jest wymagane wczytywanie programu z plików tekstowych. Przykładowe programy w Macchiato mają być budowane w kodzie programu w Javie z zaimplementowanych w Państwa rozwiązaniu klas.

Oprócz samej możliwości wykonania programu interesuje nas też śledzenie tego wykonania (czyli realizacja prostego odpluskwiacza). W związku z tym Państwa rozwiązanie powinno dawać dwie możliwości wykonywania programu:

- wykonywanie bez odpluskwiania - tu program jest wykonywany od początku do końca (chyba, że nastąpi błąd wykonania, ale nawet wtedy wypisujemy tylko komunikat i kończymy wykonywanie, bez uruchamiania odpluskwiacza),
- wykonywanie z odpluskwianiem - tu program od razu się zatrzymuje (przed wykonaniem pierwszej instrukcji) i czeka na polecenia podawane ze standardowego wejścia.

Wszystkie polecenia odpluskwiacza są jednoznakowe. Po nazwie niektórych poleceń (p. dalej) może wystąpić po (jednej lub więcej) spacji pojedyncza liczba całkowita.

Zestaw poleceń odpluskwiacza:

- **c**(ontinue)
  - bezparametrowa, odpluskwiany program zaczyna się dalej wykonywać (do zakończenia). Jeśli program już się zakończył, to polecenie wypisuje tylko stosowny komunikat (i nic więcej nie robi).
- **s**(tep) <liczba>
  - odpluskwiany program wykonuje dokładnie <liczba> kroków. Przez krok rozumiemy wykonanie pojedynczej instrukcji (liczymy także wykonania wszystkich instrukcji składowych, dowolnie zagnieżdżonych). Po wykonaniu zadanej liczby kroków wypisywana jest instrukcja (być może złożona), która ma być wykonana jako następna. Jeśli wykonywanie dojdzie do końca programu przed wykonaniem zadanej liczby instrukcji, to tylko wypisujemy stosowny komunikat, program kończy się normalnie.
- **d**(isplay) <liczba>
  - wyświetla bieżące wartościowanie. Parametr podaje z o ile poziomów wyższego bloku wartościowanie zmiennych ma być wyświetlone. Polecenie *d 0* oznacza polecenie wyświetlenia bieżącego wartościowania. Jeśli podana liczba jest większa niż poziom zagnieżdżenia bieżącej instrukcji programu, to wypisywany jest tylko stosowny komunikat.
- **e**(xit)
  - przerywa wykonanie programu i kończy działanie debuggera. Nie wypisujemy końcowego wartościowania zmiennych.

Język Macchiato dopuszcza jedynie zmienne o nazwach jednoliterowych (litery alfabetu angielskiego od `a` do `z`). Wszystkie zmienne są typu int (tego samego co w Javie).

Programy mogą zawierać następujące instrukcje:

- Blok
    - blok zawiera ciąg deklaracji zmiennych oraz sekwencję instrukcji. Każda z tych części może być pusta. Deklaracje umieszczone w bloku są widoczne od swojego końca do końca swojego bloku (i nigdzie więcej). Lokalne deklaracja mogą przesłaniać deklaracje zewnętrzne.
- Pętla for <zmienna> <wyrażenie> <instrukcje>
    - wykonuje instrukcje <wartość razy>, przy każdej iteracji <instrukcje> są wykonywane w bloku ze <zmienną> przyjmującą kolejną wartość z zakresu 0..<wartość wyrażenia>-1. Wartość wyrażenia jest wyliczana tylko raz, tuż przed rozpoczęciem wykonywania pętli (więc nawet jeśli wykonywane instrukcje zmienią tę wartość, to i tak nie zmieni się liczba obrotów tej pętli). Podana zmienna jest inicjowana kolejnymi wartościami przy każdym obrocie pętli, więc ewentualne przypisania na nią w instrukcjach składowych pętli nie zmienią przebiegu sterowania (liczby obrotów i wartości zmiennej na początku kolejnych obrotów) tej pętli. Jeśli wyliczona wartość wyrażenia jest mniejsza lub równa zeru, to pętla nie wykonuje ani jednego obrotu. Błąd przy wyliczaniu wyrażenia przerywa dalsze wykonywanie pętli (instrukcje w pętli nie zostaną ani razu wykonane).
- Instrukcja warunkowa if <wyr1> =|<>|<|>|<=|>= <wyr2> then <instrukcje> else <instrukcje>
    - znaczenie standardowe,
    - najpierw wyliczamy pierwsze, potem drugie wyrażenie,
    - błąd przy wyliczaniu wyrażenia przerywa dalsze wykonywanie tej instrukcji,
    - część else <instrukcje> można pominąć.
- Przypisanie wartości na zmienną <nazwa> <wyr>
    - nadaje zmiennej wartość równą wyliczonej wartości wyrażenia,
    - błąd przy wyliczaniu wyrażenia przerywa dalsze wykonywanie tej instrukcji (tzn. w takiej sytuacji wartość zmiennej nie zostaje zmieniona),
    - kończy się błędem, jeśli w tym miejscu programu nie ma widocznej żadnej zadeklarowanej zmiennej o podanej nazwie.
- Wypisanie wartości wyrażenia print <wyr>
    - wyliczana jest wartość wyrażenia, a następnie wypisywana w kolejnym wierszu standardowego wyjścia.
    - jeśli wyliczanie wyrażenie nie powiedzie się, ta instrukcja niczego nie wypisuje.

W blokach występują deklaracje:

- Deklaracja zmiennej <nazwa> <wyr>
    - wprowadza zmienną do bieżącego zakresu widoczności (związanego z zawierającym tę deklarację blokiem) i inicjuje ją wyliczoną wartością wyrażenia,
    - w jednym bloku nie można zadeklarować dwu (lub więcej) zmiennych o tej samej nazwie.
    - nazwy zmiennych z różnych bloków mogą być takie same, w szczególności może dochodzić do przesłaniania zmiennych (instrukcje i wyrażenia zawsze widzą tę zmienną, która była zadeklarowana w najciaśniej otaczającym bloku).
    - deklarować zmienne można jedynie na początku bloku (tzn. w ciągu deklaracji zmiennych).
    - błąd przy wyliczaniu wyrażenia przerywa dalsze przetwarzanie deklaracji (tzn. w takiej sytuacji zmienna nie zostaje zadeklarowana).

Jeśli w trakcie wykonywania instrukcji nastąpi błąd, wykonywanie programu jest przerywane oraz wypisywany jest komunikat zawierający wartości wszystkich zmiennych widocznych w bloku, w którym nastąpił błąd (te zmienne mogą być zadeklarowane w tym lub w otaczających blokach) oraz wypisywana jest instrukcja, która bezpośrednio spowodowała błąd.

W języku Macchiato wszystkie wyrażenia mają wartości całkowite. Wyrażenie może mieć jedną z następujących postaci:

- Literał całkowity

    - wartością takiego wyrażenia jest wartość literału. Składnia i zakres literałów jest taka sama jak w Javie dla typu int.
- Zmienna
    - wartością takiego wyrażenia jest wartość zmiennej widocznej w danym miejscu programu. Jeśli w tym miejscu nie ma żadnej widocznej zmiennej o podanej nazwie, wyliczenie wartości zmiennej kończy się błędem.
- Dodawanie <wyr1> <wyr2>
    - suma dwóch wyrażeń, najpierw wyliczamy pierwsze, potem drugie wyrażenie, następnie wykonujemy dodawania uzyskanych wartości. Przy przekroczenia zakresu należy podać taki wynik jaki dla takich samych wartości daje Java.
- Odejmowanie <wyr1> <wyr2>
    - różnica dwóch wyrażeń, najpierw wyliczamy pierwsze, potem drugie wyrażenie, następnie od wartości pierwszego odejmujemy wartość drugiego. Przy przekroczenia zakresu należy podać taki wynik jaki dla takich samych wartości daje Java.
- Mnożenie <wyr1> <wyr2>
    - iloczyn dwóch wyrażeń, najpierw wyliczamy pierwsze, potem drugie wyrażenie, następnie wykonujemy mnożenie uzyskanych wartości. Przy przekroczenia zakresu należy podać taki wynik jaki dla takich samych wartości daje Java.
- Dzielenie <wyr1> <wyr2>
    - dzielenie całkowite dwóch wyrażeń, najpierw wyliczamy pierwsze, potem drugie wyrażenie, następnie wartość pierwszego dzielimy przez wartość drugiego. Dla liczb ujemnych lub różnych znaków należy podać taki wynik jaki dla takich samych wartości daje Java. Wyliczanie kończy się błędem, gdy wartością drugiego wyrażenia jest zero.
- Modulo <wyr1> <wyr2>
    - reszta z dzielenia całkowitego dwóch wyrażeń, najpierw wyliczamy pierwsze, potem drugie wyrażenie, następnie wartość pierwszego dzielimy przez wartość drugiego, wynikiem jest reszta. Dla liczb ujemnych lub różnych znaków należy podać taki wynik jaki dla takich samych wartości daje Java. Wyliczanie kończy się błędem, gdy wartością drugiego wyrażenia jest zero.

Przykładowy program w języku Macchiato zapisany w metaskładni (przypominamy, że Państwa rozwiązanie **nie** ma wczytywać tekstu programów w Macchiato, zatem konkretna składnia nie ma w tym zadaniu znaczenia):

```
begin block
    var n 30
    for k n-1
        begin block
            var p 1
            k := k+2
            for i k-2
                i := i+2
                if k % i = 0
                    p := 0
            if p = 1
                print k
        end block
end block
```

Jako rozwiązanie należy oddać zestaw pakietów i klas umożliwiających wykonywania i śledzenie programów w Macchiato wraz z podanym powyżej przykładowym programem w tym języku utworzonym w funkcji main.

Życzymy powodzenia!