Uniwersytet Jagielloński Instytut Informatyki i Matematyki Komputerowej KIS Katedra Informatyki Stosowan	Metody programowania 2016/2017 Konwersje: ONP ⟺ INF	P03

Opis

Napisz program w Javie, który będzie realizował następujące operacje:

1. Konwersja wyrażeń arytmetycznych i instrukcji przypisania z tradycyjnej notacji infiksowej do ONP.
2. Konwersja wyrażeń arytmetycznych i instrukcji przypisania z ONP do notacji infiksowej z **minimalną liczbą użytych nawiasów**.

Instrukcja przypisania ma postać: operand = wyrażenie arytmetyczne.

Wyrażenia arytmetyczne mogą zawierać jedynie:

a. nawiasy: (,) - tylko w notacji infiksowej
b. operandy: małe litery alfabetu angielskiego
c. operatory:

Operator	Priorytet	Łączność	Rodzaj operatora
=	0	prawostronna	przypisania
< >	1	lewostronna	relacyjny
+ -	2	lewostronna	addytywny
* / %	3	lewostronna	multiplikatywny
^	4	prawostronna	potęgowania
~	5	prawostronna	unarny

Wejście

Dane do programu wczytywane są ze standardowego wejścia (klawiatury) zgodnie z poniższą specyfikacją. Pierwsza linia wejścia zawiera liczbę całkowitą *z* – liczbę linii zawierających wyrażenia arytmetyczne, których opisy występują kolejno po sobie.

Każda linia zawiera co najmniej 6 znaków i nie przekracza 256 znaków, może mieć jedną z dwóch postaci:

INF: wyrażenie arytmetyczne lub instrukcja przypisania, zapisane w notacji infiksowej

ONP: wyrażenie arytmetyczne lub instrukcja przypisania zapisane w notacji ONP

Przy czym wyrażenia mogą zawierać dowolne znaki. Program najpierw usuwa znaki niewystępujące w wyrażeniach, w tym spacje oraz sprawdza poprawność wyrażeń.

Można założyć, że po usunięciu błędnych symboli wyrażenia wejściowe w postaci **`INF`** są poprawne jeśli są poprawne w C. Natomiast wyrażenia w postaci **`ONP`** są poprawne jeśli są wykonalne.

Wyjście

- Wyrażenie poprzedzone na wejściu napisem "**`INF:`** " musi być na wyjściu poprzedzone napisem "**`ONP:`** " i analogicznie wyrażenie poprzedzone na wejściu napisem "**`ONP:`** "

musi być na wyjściu poprzedzone napisem **"INF: "**. W przypadku błędnego wyrażenia, na wyjściu, zamiast skonwertowanego wyrażenia pojawi napis **error**.

- W przypadku konwersji do notacji infiksowej, wyjściowe wyrażenie musi zawierać minimalną liczbę nawiasów gwarantującą taką kolejność operacji, jak w wejściowym wyrażeniu, np. **ONP: abc**** zostanie przekształcone do **INF: a*(b*c)** a nie do **INF: a*b*c**. Nawiasy obejmujące **b*c** w wyrażeniu wyjściowym wymuszają taką kolejność operacji mnożenia jaka jest w zapisie **ONP** w wyrażeniu wejściowym.
- W przypadku wyrażeń w postaci infiksowej, np. **INF:(a,+ b)/..[c3** , program pozostawia jedynie: **(a+b)/c**, pozostałe znaki, w tym spacje – odrzuca, dodatkowo sprawdza poprawność wyrażenia, po czym dokonuje konwersji, wypisując na wyjściu: **ONP: ab+c/.**
- W przypadku wyrażeń w notacji **ONP**, np. **ONP: (a,b,.).c;-,*** program pozostawia jedynie: **abc-*,** dodatkowo sprawdza, czy wyrażenie jest poprawne, po czym dokonuje konwersji, wypisując na wyjściu: **INF: a*(b-c)** .

Wymagania implementacyjne

Ogólnie jak w poprzednich programach, w szczególności jedynym możliwym importem jest import skanera wczytywania z klawiatury. Tym samym klasę stosu należy zaimplementować samodzielnie.

Przykład danych

wejście:	wyjście:
18	
INF: a)+(b	ONP: error
ONP: ab+a~a-+	INF: a+b+(~a-a)
INF: a+b+(~a-a)	ONP: ab+a~a-+
INF: x=~~a+b*c	ONP: xa~~bc*+=
INF: t = ~ a < x < ~b	ONP: ta~x<b~<=
INF: ~a-~~b<c+d&!p|!!q	ONP: error
INF: a^b*c-d<xp|q+x	ONP: error
INF: x=~a*b/c-d+e%~f	ONP: xa~b*c/d-ef~%+=
ONP: xabcdefg++++++=	INF: x=a+(b+(c+(d+(e+(f+g)))))
ONP: ab+c+d+e+f+g+	INF: a+b+c+d+e+f+g
ONP: abc++def++g++	INF: a+(b+c)+(d+(e+f)+g)
ONP: abc++def++g+++	INF: error
INF: x=a=b=c	ONP: xabc===
ONP: xabc===	INF: x=(a=(b=c))
INF: x=a^b^c	ONP: xabc^^=
INF: x=a=b=c^d^e	ONP: xabcde^^===
ONP: xabcde^^===	INF: x=(a=(b=c^(d^e)))
INF: x=(a=(b=c^(d^e)))	ONP: xabcde^^===