Evaluarea expresiilor matematice utilizând forma poloneză postfixată

Universitatea "Transilvania" din Braşov

Forma poloneză postfixată

Reprezintă un mod de rearanjare a unei expresii aritmetice, inventat de un matematician polonez.

Există două tipuri de forme poloneze:

a. Forma poloneză (scriere prefixată): presupune scrierea operatorilor înaintea operanzilor

b. Forma poloneză inversă (scriere postfixată): presupune scrierea operatorilor după operanzi

Exemplu

Expresia: $3 * 4 - 3 * (24 - 12) - 7$

a. Forma poloneză (scriere prefixată): $- - *3\ 4 * 3 - 24\ 12\ 7$

b. Forma poloneză inversă (scriere postfixată): $3\ 4 * 3\ 24\ 12 - * - 7-$

Observații:

- În forma poloneză inversă nu mai sunt necesare parantezele.
- Din forma poloneză inversă este foarte ușor de evaluat expresia aritmetică.

Algoritm de construcție a formei poloneze postfixate

- utilizează o stivă în care sunt păstrați operatorii.
- fiecărui operator i se atribuie o precedență:

precedență	operator
(	0
+, -	1
*, /	2
^	3

unde ^ reprezintă ridicarea la putere.

Algoritm de construcție a formei poloneze postfixate

- Notăm cu OP stiva în care se introduc operatorii. Inițial $S = \emptyset$
- Notăm cu FP vectorul în care memorăm forma poloneză
- Parcurgem șirul de caractere care conține expresia aritmetică și pentru fiecare element E - operator sau număr efectuăm pașii următori.

Algoritm de construcție a formei poloneze postfixate

Algoritm: FormaPoloneza(string *sir*)

```
pentru fiecare E ∈ sir executa
    daca E este numar atunci
        FP.push_back(E)
    altfel
        daca E = '(' atunci
            OP.push('(')
        altfel
            daca E = ')' atunci
                cat_timp OP ≠ ∅ si OP.top() ≠ '(' executa
                    FP.push_back(OP.top())
                    OP.pop()
                OP.pop()
            altfel
                cat_timp OP ≠ ∅ si prec(E) ≤ prec(OP.top()) executa
                    FP.push_back(OP.top())
                    OP.pop()
                OP.push(E)
//Continuare pe slide-ul urmator
```

Algoritm de construcție a formei poloneze postfixate

cat_timp $S \neq \emptyset$ **executa**
    $FP.push\_back(OP.top())$
    $OP.pop()$
return FP

Observații:

- Acest algoritm nu tratează erorile care pot apărea în expresia aritmetică (de exeplu de parantezare)
- S-au considerat doar operatori numere naturale de o singură cifră.

Evaluarea expresiei aritmetice din forma poloneză postfixată

Algoritm: Evaluare(vectorul *FP*)

Intrare: vectorul *FP* care conține forma poloneză

```
Num = ∅
pentru fiecare E ∈ FP executa
    daca E este numar atunci
        Num.push(E)
    altfel
        //E este operator
        y = Num.top()
        x = Num.top()
        rez = x E y//efectuez operatia
        Num.push(rez)
return Num.top()
```

Observații:

- *FP* conține caractere, *Num* este o stivă de **numere reale**!
- când se introduce un număr din *FP* în *Num*, trebuie mai întâi transfromat caracterul *E* în numărul corespunzător
- efectuarea operației între *x* și *y* se poate face printr-o funție de evaluare / folosind o instrucțiune de tip **switch**
- nu au fost tratate erorile care pot apărea dacă expresia aritmetică este greșită (prea mulți operatori)

Indicații de programare

1. Pentru stive:
   - stiva de operatori:

```
std::stack<char> OP;
```

   - stiva de numere:

```
std::stack<double> Num;
```

   - verificarea dacă o stivă e vidă: - cu metoda **empty()**

```
if(!OP.empy()) //verifica daca OP nu e vida
{
        ...
}
```

Transfromarea unui caracter *E* cifră într-un int:

```
int val = (int)(E - '0');
```

- se lucrează cu codurile ASCII