3 Formularea temei

In rezolvarea cerintelor, variabilele utilizate sunt **doar literele mici ale alfabetului limbii engleze**.

3.1 Cerinta 1

Cerinta 1 va fi evaluata pe 4 teste si va ajuta sa obtineti un punctaj de 4p din nota 10.

Fie dat ca input un sir hexa, se cere sa se afiseze la *standard output* instructiunea *assembly* de executat.

De exemplu, pentru inputul `A78801C00A7890EC04`, se va afisa la standard output `x 1 let x -14 div`.

3.2 Cerinta 2

Cerinta 2 va fi evaluata pe 4 teste si va ajuta sa obtineti un punctaj de 2p din nota 10.

Fie dat ca input o instructiune in limbajul de asamblare al procesorului aritmetic considerat, se cere sa se afiseze la *standard output* evaluarea instructiunii. Pentru aceasta cerinta, in instructiune **nu exista variabile**, ea fiind formata doar din numere intregi si operatii.

De exemplu, poate fi data instructiunea `2 10 mul 5 div 7 6 sub add`. Rezultatul trebuie sa fie conform urmatorului algoritm:

- se adauga 2 pe stiva;
- se adauga 10 pe stiva;
- se identifica operatia mul, se aplica inmultirea dintre 2 si 10, se obtine 20, se elimina 2 si 10 de pe stiva si se pastreaza doar 20;
- se adauga 5 pe stiva;
- se identifica div - actioneaza ca `20 div 5`, iar rezultatul este 4; se elimina 20 si 5 de pe stiva, si se pastreaza doar 4;
- se adauga 7 pe stiva;
- se adauga 6 pe stiva;
- se identifica sub - se calculeaza diferenta dintre 7 si 6, se obtine 1, se elimina 7 si 6 de pe stiva, si se adauga pe stiva valoarea 1. **Atentie!** in acest moment, pe stiva avem 4 (la baza) si 1 in varf, intrucat *sub* este operatie binara si a lucrat doar cu argumentele 7 si 6, dar nu si cu 4 care era deja la baza stivei.
- se identifica add - se calculeaza suma dintre 1 si 4, se obtine 5, se elimina 1 si 4 de pe stiva, se adauga 5;
- am terminat de parcurs sirul, iar rezultatul obtinut este, acum, situat in varful stivei. Rezultatul acestui calcul este 5.

O sugestie de implementare a algoritmului gasiti la finalul acestui document. **Important!** Se cere evaluarea doar pe **unsigned**! **Se garanteaza ca toate operatiile vor fi pe unsigned.**

3.3 Cerinta 3

Cerinta 3 va fi evaluata pe 3 teste si va ajuta sa obtineti 1.5p din nota 10.

Fie dat ca input o instructiune in limbajul de asamblare al procesorului aritmetic considerat. Se cere sa se afiseze la *standard output* evaluarea instructiunii. Pentru aceasta cerinta, spre deosebire de cerinta 2, **se folosesc variabile** introduse prin **let**.

Un exemplu de input poate fi `x 1 let 2 x add y 3 let x y add mul`.

Evaluarea va fi facuta astfel:

- se adauga x si 1 pe stiva, este gasit let, si se intelege de acum ca x = 1 in toata expresia aritmetica; sunt eliminati x si 1 de pe stiva;
- se adauga 2 si 1 pe stiva (deoarece acel x este = 1 de acum);
- se intalneste add, se calculeaza suma 3, se elimina 2 si 1 de pe stiva si se pastreaza doar 3;
- se adauga y si 3 pe stiva, este gasit let, si se intelege de acum ca y = 3 in toata expresia aritmetica; sunt eliminati y si 3 de pe stiva;
- se adauga 1 si 3 pe stiva (x, respectiv y);
- se efectueaza adunarea, rezultatul va fi 4, se elimina 1 si 3 de pe stiva, se adauga 4;
- este identificat mul, iar pe stiva aveam deja 3 (de la a treia bulinuta din explicatia curenta) si 4, de la bulinuta anterioara, si se calculeaza rezultatul, 12, se elimina apoi 3 si 4 de pe stiva si se adauga 12;
- nu mai sunt elemente in sir, deci rezultatul final este 12.

Exact ca la cerinta a doua, se garanteaza ca toate operatiile vor fi aplicate pe **unsigned**.

3.4 Cerinta 4

Cerinta 4 va fi evaluata pe 5 teste si va ajuta sa obtineti 1.5p din nota 10.

Pentru aceasta cerinta, vom introduce operatii simple de lucru cu matrice. O matrice poate fi reprezentata in forma

```
nrLinii nrColoane nrLinii*nrColoane_elemente
```

Operatiile pe care le putem utiliza pe matrice sunt:

- add - adunam toate elementele din matrice cu valoarea operandului;
- sub - scadem din toate elementele din matrice valoarea operandului;
- mul - inmultim toate elementele din matrice cu valoarea operandului;

- div - impartim toate elementele din matrice la valoarea operandului;
- rot90d - rotim matricea la 90 de grade spre dreapta;

Operatiile pe matrice contin doar instructiunea `let` si una dintre operatiile mentionate anterior. **Nu sunt instructiuni complexe, precum cele de la cerintele anterioare!**

```
x 2 3 1 2 3 4 5 6 let M -2 add
```

In acest caz, matricea `x` este o matrice de `2 linii x 3 coloane`, care are urmatoarea forma:

```
1 2 3
4 5 6
```

Se aplica o adunare cu -2 pe toate elementele matricei:

```
-1 0 1
2  3 4
```

Ca output, la *standard output* se va afisa reprezentarea acestei matrice in forma in care e introdusa in operatie: numarul de linii, numarul de coloane, respectiv elementele matricei, de la stanga la dreapta si de sus in jos. In acest caz, outputul va fi `2 3 -1 0 1 2 3 4`.

Daca avem urmatoarea instructiune: `x 2 3 -1 0 1 2 3 4 rot90d`, atunci se aplica urmatoarea rotire la dreapta cu 90 de grade:

```
2 -1
3  0
4  1
```

In acest caz, outputul va fi

```
3 2 2 -1 3 0 4 1
```

Important! Toate operatiile din aceasta cerinta sunt aplicate pe signed!