Nr. 3

Cerință

Se citesc de la tastatură *n* numere întregi, fiecare reprezentând timpul de deservire pentru un client. Clienții de index par vor fi introduşi într-o coadă, iar clienții de index impar într-o a doua coadă, utilizând codul ataşat. Să se implementeze următoarele funcționalităţi:

(1) o funcție care să permită afișarea datelor din fiecare nod dintr-o coadă;

(2) să se completeze funcția *main* cu codul necesar introducerii elementelor de index par în coada al cărei început este indicat de pointerul ***par**, iar cele de index impar în coada al cărei început este indicat de pointerul ***impar**.

(**Bonus**) Să se declare şi apeleze o funcție care să permită calcularea timpului total necesar de servire al clienților dintr-o coadă al cărei pointer de început este transmis prin parametru.

Date de intrare

Se vor citi de la tastatură (fluxul *stdin*) următoarele date:

- O valoare întreagă *n* reprezentând numărul total de clienți, urmată de tasta Enter;
- Pentru fiecare client în parte, un număr întreg reprezentând valoarea timpului de prelucrare, care va fi stocată în câmpul *timpPrelucrare* al fiecărui client, urmat de tasta Enter.

Date de ieșire

Programul va afișa pe ecran la ieşire:

- Valoarea din membrul *timpPrelucrare* al fiecărui nod al cozii care reține elementele de index **par**, fiecare pe o linie nouă, în ordinea în care au fost introduse în coadă;
- Valoarea din membrul *timpPrelucrare* al fiecărui nod al cozii care reține elementele de index **impar**, fiecare pe o linie nouă, în ordinea în care au fost introduse în coadă;
- Valoarea timpului total necesar de deservire al clienților cozii de elemente de index **par**;
- Valoarea timpului total necesar de deservire al clienților cozii de elemente de index **impar**.

ATENȚIE la respectarea cerinţei problemei: afişarea rezultatelor trebuie făcută EXACT în modul in care a fost indicat! Cu alte cuvinte, pe stream-ul standard de ieşire nu se va afişa nimic în plus fată de cerinţa problemei; ca urmare a evaluării automate, orice caracter suplimentar afişat, sau o afişare diferită de cea indicată, duc la un rezultat eronat.

Precizări

- Biblioteca ataşată permite declararea şi iniţializarea unei cozi de clienți ai căror timpi de prelucrare sunt citiți de la tastatură. Nodurile relaționează între ele prin intermediul pointerilor *struct COADA *urmator*, care stochează adresa de memorie a nodului următor din coadă;
- Funcția *struct COADA *enqueue(struct COADA *head, int tPrelucrare)* este o funcție generală, prin intermediul căreia se creează şi se alocă memorie pentru un nod nou. Conținutul acestuia este inițializat cu valoarea transmisă ca parametru, iar locația nodului următor este inițializată cu *NULL*, deoarece în prima instanță acest nod nu este inserat în listă. Dacă lista este nulă, atunci noul nod devine primul nod din coadă iar

adresa lui este returnată printr-un pointer de tip *struct LISTA* altfel, noul nod este adăugat la sfârşitul listei de date prin parcurgerea listei cu ajutorul structurii repetitive *while*, cu condiţia de oprire activată de nodul a cărui adresă este *NULL* prin expresia: *nodCurent!=NULL*.

Restricţii

Indexarea elementelor citite de la tastatură începe de la 0. Astfel, primul element citit de la tastatură va fi introdus în lista al cărei început este indicat de pointerul *par.

Exemplu

Intrare	Ieşire
5 2 7 5 9 3	2 5 3 7 9 10 16

Timp de lucru: 60 de minute