EXAMEN STRUCTURI DE DATE

Subiectul 8

1 5p

De la tastatură se citesc numere naturale din mulţimea **M={1,2,..., 100}** şi se introduc într-o tabelă de dispersie cu 10 intrări. Se cere:

a) Dacă avem de ales între funcţiile de dispersie (*hashing*) H_1**:M** $\rightarrow${**0,1,..., 9**}, $H_1(x)$**= 13x mod 10** şi H_2**:M**$\rightarrow${**0,1,...,9**}, $H_2(x)$**= x² mod 10**, care dintre ele este mai bună şi de ce?

0/ 5p b) Dacă numerele citite sunt 1, 2, 3, 5, 7, 11,13,19 şi funcţia de dispersie este H_2 desenaţi tabela de dispersie. Ce cantitate de memorie este folosită pentru stocarea ei?

0/

Pentru o permutare $p \in S_n$ fie F_p figura determinată de următoarele puncte din plan:

$$(1, 0), (1, p(1)), (2, p(2)),..., (n, p(n)), (n, 0).$$

2 Se cere:

4p a) Să se scrie o funcţie

0/

double PermArea(Perm p);

care calculează şi returnează aria figurii F_p corespunzătoare permutării p;

6p b) Pentru ce permutare $p \in S_n$ aria figurii F_p este maximă ? Justificaţi răspunsul.

1/

Tabela de dispersie (**HashTable**):

3 0/ 4p
- exemplu
- structura de date C++ (cu Templates) pentru memorare,
- declaraţia şi implementarea funcţiilor de inserare şi căutare valoare.

vezi sub 5 pb. 4.

1/

(Se vor puncta numai exemplele ce folosesc template-uri)

Sortarea cu găleţi (**Bucketsort**):

4 0/ 4p
- descriere, exemplu,
- funcţie **C** de sortare,
- ordin de operaţii.

0/

Să se implementeze un mecanism propriu de alocare (*Alocator*) blocuri de memorie compus din functii complementare de **alocare** si **dealocare** a memoriei allocate dintr-o zona de memorie proprie care începe la adresa **void* MEM**, gestionate astfel:

5 0/ 4p a)

1/

void * MEM_Alloc(**void*** start, **void*** end, **long** num_values);

verifica memoria de la start la end si cauta o zona continua de **num_values** octeti nealocati pe care ii aloca si returneaza adresa de inceput a zonei alocate;

0/ 4p b)

1/

void * MEM_Dealloc(**void*** start_alloc, **void*** end);

care verifica zona care începe la start_alloc daca este alocata si in caz afirmativ ii dealoca si returneaza adresa zonei dealocate sau NULL in caz contrar.

(Indicaţie: se va implementa o clasă de alocare folosind vector unidimensional sortat)

Să se scrie o funcţie

6 0/ 4p

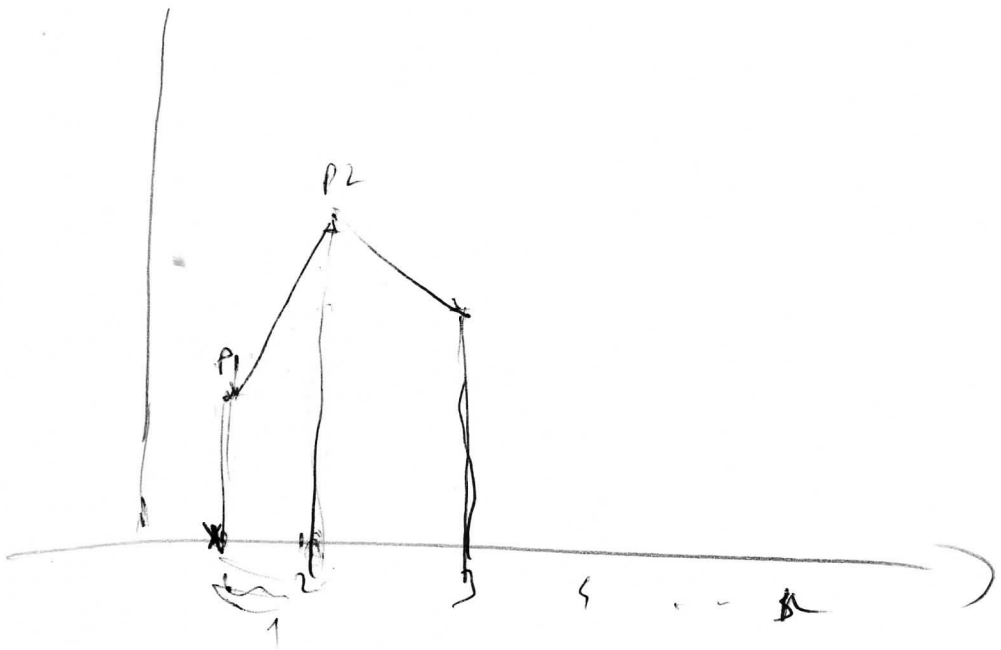
1/

void BTreeHeight(BTree* bt);

care calculeaza inaltimea unui arbore de cautare binara.

parcurgere in latime

TOTAL: 9p+1p oficiu=10p



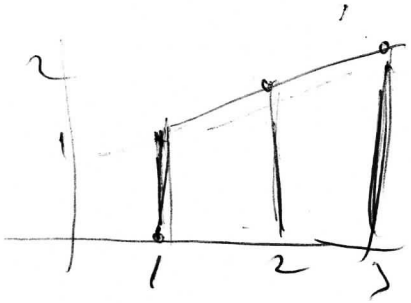
$\frac{(3+6)\cdot 4}{2}$ $\frac{3+6}{2}$

```
sum=0;
for(int i=1; i< p.count; i++)
    sum=sum+(p.data[i]+p.data[i+1])/2;
```

3 1 2

1 2 3

1 3 2



$\frac{1+2}{2} + \frac{2+3}{2} = \frac{8}{2} = 5$

$\frac{1+3}{2} + \frac{3+2}{2} = \frac{9}{2}$