TEST DE EVALUARE SUMATIVĂ – INFORMATICĂ

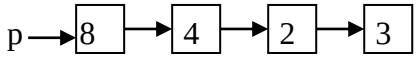
Clasa a XI-a (matematică-informatică – intensiv, limbajul C/C++)

Capitolul - STRUCTURI DE DATE ALOCATE DINAMIC

Vlad Elena
Colegiul Național „Emil Botta”- Adjud

1. Ştiind că adresa de început a listei reprezentate în desenul următor este memorată în variabila **p** şi că în fircare nod al listei reţine în câmpul **inf** numărul scris în desen şi în câmpul **adr_urm** adresa următorului element, stabiliţi ce reprezintă expresia de mai jos:

p ->adr_urm ->adr_urm ->nr;



a) adresa elementului al doilea (elementul ce memorează valoarea 4);

b) valoarea memorată în nodul al treilea (valoarea 2);

c) adresa elementului al treilea (elementul ce memorează valoarea 2);

d) valoarea memorată în nodul al doilea (valoarea 4); **0.25p**

2. Stiind că există o listă liniară simplu înlănţuită nevidă, fiecare nod reţinând în câmpul **ref** adresa elementului următor al listei, şi ştiind că variabilele **v** şi **s** reţin adresa primului şi respectiv adresa ultimului element al listei, explicaţi care este efectul instrucţiunii:

s->ref=v; …….

0.25p

3. Într-o listă înlănţuită cu cel puţin trei elemente, fiecare element reţine în câmpul **nr** un număr întreg, şi în câmpul **urm** adresa următorului element din listă. Dacă variabila **prim** reţine adresa primului element din listă, stabiliţi care dintre următoarele secvenţe afişează suma tuturor numerelor memorate în listă, mai puţin cele reţinute de primul şi ultimul element:

```
a. s=0; p=prim;
   while (p->urm!=NULL) {p=p->urm; s+=p->nr;}   cout << s;
b. s=0; p=prim;
   while (p!=NULL) { s+=p->nr; p=p->urm; }   cout << s;
c. s=0; p=prim->urm;
   while (p!=NULL) { s+=p->nr; p=p->urm; }   cout << s;
d. s=0; p=prim;
   while (p->urm!=NULL) {p=p->urm; s+=p->nr;}   cout << s-p->nr;;
```

1p

4. Dacă **L** este o coadă, cu **p** adresa primului şi **u** adresa ultimului element, iar **r** este adresa unui element ce urmează a fi adăugat în coadă, stabiliţi care dintre următoarele este o operaţie corectă de adăugare:

a. **r->adr=p; p=r;**

b. **u->adr=r; u=r;**

c. **r=u->adr; u=r;**

d. **r->adr=u; u=r;** **1p**

5. Se consideră o listă liniară simplu înlănţuită ale cărei noduri reţin în câmpul **urm** adresa nodului următor sau **NULL** dacă nu există un element următor. Pentru inserarea unui nod aflat la adresa **p** imediat după un nod al listei aflat la adresa **q** se utilizează unele din următoarele atribuiri:

1) **p->urm=q**; 2) **q->urm=p**; 3) **p=q->urm**; 4) **q=p->urm**;

5) **p->urm=q->urm;** 6) **q->urm=p->urm**;

Stabiliţi care dintre acestea se utilizează şi în ce ordine:

a. 3 6 b. 2 4 c. 5 2 d. 2 3

1p

6. Fiecare element al unei liste liniare simplu înlănţuită alocată dinamic reţine în câmpul **adru** adresa elementului următor din listă. Dacă **p** reţine adresa primului element, iar lista are cel puţin trei elemente, care din următoarele secvenţe de instrucţiuni şterge al doilea element al listei?

a) **q=p->adru; p->adru=q->adru; delete q**; b) **p->adru = p->adru->adru; delete p->adru**;

c) **q= p->adru; delete q; p->adru= q->adru**; d) **delete p->adru;**

1p

7. Într-o listă liniară dublu înlănţuită care începe cu elementul memorat la adresa **p** şi conţine cel puţin patru elemente, fiecare element reţine în câmpul **urm** adresa elementului următor, iar în câmpul **pre** adresa elementului precedent, iar în câmpul **inf** o valoare întreagă. Care dintre următoarele variante tipăreşte valoarea celui deal treilea element al listei?

a) **cout << p->urm->urm->pre->inf;**

b) **cout << p->urm->urm->urm->pre->inf;**

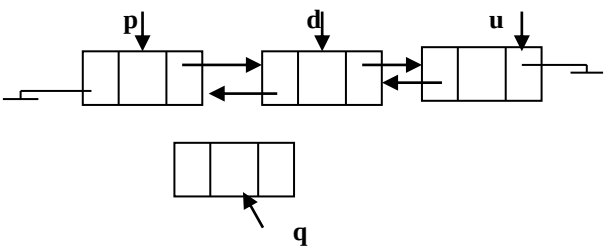
c) **cout << p->urm->urm->urm**;

d) **cout << p->urm->urm**;

1p

8. În figura de mai jos se consideră implementarea nodurilor şi declararea variabilelor necesare definirii unor operaţii într-o structură dinamică de tip listă liniară dublu înlănţuită, precum şi reprezentarea grafică:

```
struct nod { int info; nod ls, ld;}
int info_c;
nod *p, *q, *r, *s, *d;
```



a) Se consideră următoarele grupuri de instrucţiuni. Precizaţi ordinea în care trebuie scrise astfel încât prin parcurgerea lor să se realizeze inserarea nodului **q** la stânga nodului indicat de **d.**

```
1. while(info_c!=d->info && d!=NULL) d=d->ld;
2. cout << ”info_c=”; cin >> info_c;
3. d=p;
4. if (d==NULL) cout << ”NU”;
   else
    {5. d->ls=q;
     6. q->ls=d ->ls;
     7. cout << ”info”; cin >> q->info;
     8. d->ls->ld=q;
     9. q->ld=d;
    10. q=new pnod;
     }
```

0.2p x 10 = 2p

b) Reprezentaţi pe desen realizarea inserării din care să rezulte şi ordinea grupurilor de instrucţiuni găsită la punctul 1**.** **0.5p**

9. Penrtu implementările şi declaraţiile de structuri dinamice de tip listă dublu înlănţuită de la punctul **8**. să se scrie un subprogram care realizează ştergerea nodului din stânga nodului indicat de **d.**

1p

Notă: Toate cerințele sunt obligatorii. Timp de lucru 50 de minute. Se acordă un punct din oficiu.

BAREM DE CORECTARE

Răspuns	**Punctaj**
1. b	0.25
2. este legătura de reacţie pentru realizarea unei liste circulare	0.25
3. a	1
4. b	1
5. c	1
6. a	1
7. b	1
8. a) 3,2,1,4,10,7,5,6,8,9	0.2 x 10 = 2 se acordă 0.2p pentru fiecare instrucţiune corect poziţionată
8. b) realizare desen	0.5

9.
void sterge_s()
{d=p;
cout<<"info_c"=;
cin>>info_c ;
while(info_c!=d->info && d!=NULL)
d=d->ld;
if(d==NULL) cout<< "NU";
else
{q=d->ls;
d->ls->ls->ld=d->ls->ld;
d->ls=d->ls->ls;
delete q;}
}
	-0.5p pentru identificarea elementului căutat;
	
	-0.5p penrtu ștergerea elementului găsit

Se acordă un punct din oficiu.