

Test S10 - Liste

oritmi

Dragoș Remetea

Question 8
Not yet answered
Marked out of 1.00
Flag question

Care este complexitatea în termeni de $O(f(n))$ pentru o funcție care numără elementele unei liste?

a. $O(n)$
 b. $O(\log n)$
 c. $O(n^2)$
 d. $O(1)$

Previous page Next

Comparat cu un tablou care conține aceleși informații utile, o structură de date de tip listă:

a. ocupă mai multă memorie;
 b. are aceeași dimensiune în memorie;
~~c. are avantajul accesului mai rapid la un anumit element; nu are acces direct~~
 d. ocupă mai puțină memorie;

Paul-Ioan Miclăuș

Next page

Question 4
Answer saved
Marked out of 1.00
Flag question

Funcția reverse() e folosită pentru a insera elementele unei liste înăntărite. Ce linie lipsește la finalul acestei funcții?

```
struct node
{
    int data;
    struct node* next;
};

void reverse(struct node** head_ref)
{
    struct node* prev = NULL;
    struct node* current = *head_ref;
    struct node* next;
    while (current != NULL)
    {
        next = current->next;
        current->next = prev;
        prev = current;
        current = next;
    }
    /*LINIE LIPSA*/
}
```

a. `*head_ref = current;`
 b. `*head_ref = NULL;`
 c. `*head_ref = next;`
 d. `head_ref = prev;`

Bogdan ANCA

Diagram illustrating the reversal of a linked list:

- Initial state: Four nodes labeled 0, 1, 2, 3. Node 0 is the head.
- Step 1 (I): `prev = NULL`, `current = head`. $meot = 1$.
- Step 2 (II): `next = current->next`, `current->next = prev`, `prev = current`, `current = next`. $meot = 2$, $current \rightarrow m = \text{NULL}$, $prev = 0$.
- Step 3 (III): `next = current->next`, `current->next = prev`, `prev = current`, `current = next`. $meot = 3$, $current \rightarrow m = \text{NULL}$, $prev = 1$.
- Step 4 (IV): `next = current->next`, `current->next = prev`, `prev = current`, `current = next`. $meot = \text{NULL}$, $current \rightarrow m = \text{NULL}$, $prev = 2$.
- Final state: $head_ref = prev$ (Node 0).

Clear my choice

Care este complexitatea în termeni de $O(f(n))$ pentru scoaterea unui element dintr-o stivă?

a. $O(n/2)$
 b. $O(1)$
 c. $O(n)$
 d. $O(n-1)$

și push tot $O(1)$

Pentru o listă înăntăuită p, funcția f returnează 1 doar dacă:

```
struct item  
{  
    int data;  
    struct item * next;  
};  
  
int f(struct item *p) → lista  
{  
    return (  
        (p == NULL) ||  
        (p->next == NULL) ||  
        (( p->data <= p->next->data) && f(p->next))  
    );  
}
```

- a. nici un răspuns corect
- b. elementele din listă sunt sorteate în ordine crescătoare a valorii lor
- c. elementele din listă sunt sorteate în ordine descrescătoare a valorii lor
- d. toate elementele din listă au valori diferite

Activitatea 30 în Sistem

itm

Question 2
Not yet answered
Marked out of 1.00
Flag question

În cazul cel mai nefavorabil, numărul de comparații necesare pentru a face o căutare într-o listă simplă înăntăuită de lungime n este:

- a. $n/2$
- b. $\log_2 n$
- c. $\log_2 n - 1$
- d. n

Previous page

Question 9
Not yet answered
Marked out of 0
Flag question

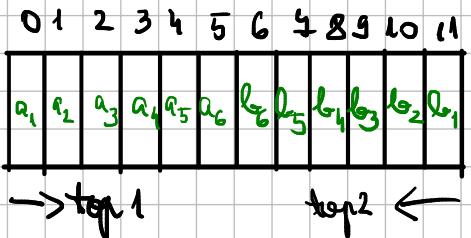
În cazul funcțiilor recursive, fiecare autoapel al funcției presupune salvarea în stivușistemului a:

- a. contextului apelului → parametrii funcție → adresa de returnare ↴ valoarea variabilelor locale
- b. variabilelor globale astăzi
- c. parametrilor pe linie de comandă ai programului care apelează funcție
- d. unui contor care retine numarul de apeluri

Previous page

Un singur tablou este folosit pentru a implementa două stive. Stivele cresc de la cele două extremități. Variabilele top1 și top2 ($\text{top1} < \text{top2}$) indică spre locația elementului din vîrf al fiecărei stive. Dacă spațiul este folosit eficient, condiția pentru stive pline este:

- Select one:
- a. $\text{top1} == (\text{top2}-1)$
 - b. $(\text{top1} == \text{MAXSIZE}/2) \&\& (\text{top2} == \text{MAXSIZE}/2+1)$
 - c. $(\text{top1} == \text{MAXSIZE}/2) \|\ (\text{top2} == \text{MAXSIZE})$
 - d. $\text{top1} + \text{top2} == \text{MAXSIZE}$



size = 12

$$6 - 1 = 5$$

Activati Windows Next page
Accesati Setari pentru a activa Windows

Question 2
Answer saved
Marked out of 2.00
Flag question

Un nod se numește dacă adăugându-l în tabloul soluție există posibilitatea să ajungem la o soluție completă.

Select one:

- a. succesor
- b. acceptabil
- c. invalid
- d. predecesor

Clear my choice

Explicație:

Un nod **acceptabil** este un nod care, atunci când este adăugat în tabloul soluției, există posibilitatea să conduce la o soluție completă. Acest termen este des utilizat în algoritmi de căutare și rezolvare a problemelor, cum ar fi **backtracking**, pentru a evalua dacă un anumit pas (sau stare) poate face parte dintr-o soluție validă.

De ce nu celelalte opțiuni?

- a. succesor – Aceasta indică un nod derivat dintr-un alt nod, dar nu implică neapărat că este acceptabil.
- c. invalid – Un nod invalid nu poate fi parte a unei soluții, deci este opusul unui nod acceptabil.
- d. predecesor – Se referă la nodurile anterioare din arborele de căutare și nu are legătură cu caracterul acceptabil al soluției.

Previous page

IRO2-SDA ▶ Teste Laborator ▶ Test Backtracking

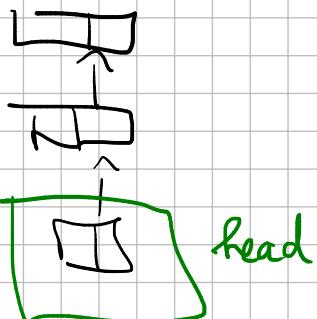
Question 1
Not yet answered
Marked out of 2.00
Flag question

Pentru o stivă implementată cu liste simplu înaintaute, dacă în cazul operației de adăugare în stivă ("push") se adaugă un nod la finalul listei, atunci prin operația de extragere din stivă ("pop") se elimină un nod din capul listei.

Select one:

- True
- False

Next page



In timpul unui apel de funcție stiva sistem poate fi goală.

Select one:

- True
- False

→ la apel, nu poate fi goală
→ avem năcar contextul apelului

▶ Teste Laborator ▶ Test Backtracking

Question 3
Not yet answered
Marked out of 2.00
Flag question

Dacă pe nivelul k ($k > 1$) al stivei am verificat toate valorile posibile, atunci?

Select one:

- a. Se sare un nivel
- b. Se trece pe nivelul următor
- c. Se revine pe nivelul anterior
- d. Algoritmul se încheie

Clear my choice

→ am verificat una
→ le-am verificat pe toate

Question 1Not yet
answeredMarked out of
2.00

Flag question

Stiva este o structura de tip FIFO (First In First Out).

Select one:

 True FalseLast in
First out (LIFO)

← Test Liste

Jump to...

Accesul la elementul din varful stivei se face este de complexitate O(1).

Select one:

 True False

True

Jump to...

**Question 1**Not yet
answeredMarked out of
2.00

Flag question

Pentru o stiva implementata cu liste simplu inlantuite, daca in cazul operatiei de adaugare in stiva ("push") se adauga un nod la finalul listei, atunci prin operatia de extragere din stiva ("pop") se elimina un nod din capul listei.

Select one:

 False

Next page

Question 1Not yet
answeredMarked out of
2.00

Flag question

O stiva poate fi implementata cu ajutorul unei liste simplu inlantuite

Select one:

 True False

Facebook Test Liste (page 5 of 5) Sdanc | Jitsi Meet

cv.upt.ro/mod/quiz/attempt.php?attempt=2056308&cmid=1124998&page=4

Campus Virtual UPT Cel Dashboard Preferences My Menu My courses English (en)

Finish attempt ... Time left 0:11:42

Not yet answered Marked out of 2.00 Flag question

```

struct nod
{
    int data;
    struct nod * next;
};

int functie(struct nod*p)
{
    return (
        (p == NULL) ||
        (p->next == NULL) ||
        ((p->data <= p->next->data) && functie(p->next))
    );
}

```

Pentru o lista simplu înaintuită date ca parametru de intrare, funcția returnează 1 dacă și numai dacă:

Select one:

- a. elementele listei sunt în ordine descrescătoare
- b. elementele listei au valori diferite, două cîte două
- c. lista e vida sau are exact un element
- d. elementele listei sunt în ordine crescătoare

*e și mai sus
grila*

Type here to search 4:34 PM 11/23/2020

Dacă se dorește crearea unei liste în ordinea furnizării elementelor, atunci este nevoie de o secvență care inseră un nod la începutul unei liste.

→ la final

Select one:

- True
- False

Test Liste

Accesul la elementul din varful stivei se face este de complexitate $O(1)$.

Select one:

- True
- False

Tutor: Cioară Razvan - vcard +7 More Enrolled users Calendar Grades

Test Liste

Coada bazată pe priorităte ("priority queue") este structura de date abstractă care permite inseră unui element și suprimarea celui mai vechi element în mod direct (cu o complexitate egală cu $O(1)$).

! = prioritatea cea mai mare

Select one:

- True
- False

Next page

 $O(\log^N)$

Implementarea cozilor bazate pe priorităate folosind liste neordonate este potrivită în situații în care se fac multe inserții și mai puține extrageri.

Select one:

True
 False

Implementarea cozilor bazate pe priorităate folosind liste neordonate este potrivită în situații în care:

- Se fac multe inserții și mai puține extrageri:
 - Inserția într-o listă neordonată are complexitatea $O(1)$, deoarece elementul poate fi adăugat la sfârșitul listei fără niciun cost suplimentar.
 - În schimb, extragerea unui element cu prioritățile maximă (sau minimă) implică căutarea elementului dorit, ceea ce are o complexitatea $O(n)$, deoarece lista nu este ordonată.
- Cazuri de utilizare:
 - Dacă numărul de operații de inserție este mult mai mare decât cel al operațiilor de extragere, această implementare devine eficientă datorită costului redus al inserției.

De ce nu liste ordonate?

În cazul în care extragerile sunt mai frecvente decât inserțiile, o implementare cu liste ordonate sau structuri precum heap-uri este mai potrivită, deoarece operația de extragere devine mai rapidă.

Astfel, răspunsul este True deoarece implementarea cu liste neordonate este optimă pentru scenarii cu multe inserții și puține extrageri.

Question 5
 Answer saved
 Marked out of 2.00
[Flag question](#)

Care este secvența de cod corectă pentru a accesa informația celui de-al doilea nod, dacă *prim indica spre primul nod al listei?

```
struct nod{
    int info;
    struct nod*next;
};

struct nod *prim;
....
```

Select one:

a. prim->info
 b. prim->info->next
 c. prim->next->info
 d. prim->next->next->info

[Clear my choice](#)

Următoare funcție primește ca argument o listă simplu întântuită. Modifica lista, mutând ultimul element pe prima poziție și returnează lista modificată. O parte din cod lipsește, alegeti varianta corectă:

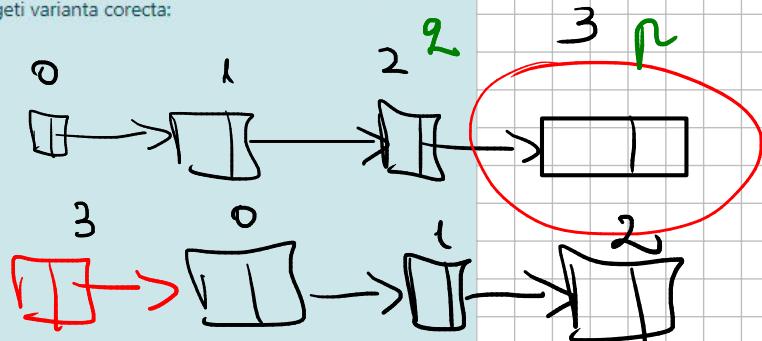
```
typedef struct node
{
    int data;
    struct node *next;
}Node;

Node *muta_in_fata(Node *prim)
{
    Node *p, *q;
    if ((prim == NULL) || (prim->next == NULL))
        return prim;
    q = NULL; p = prim;
    while (p->next != NULL)
    {
        q = p;
        p = p->next;
    }
}
```

return prim;

Select one:

- a. q->next = NULL; prim = p; p->next = prim;
 b. q = NULL; p->next = prim; prim = p;
 c. prim = p; p->next = q; q->next = NULL;
 d. q->next = NULL; p->next = prim; prim = p;



I. $p = 0$
 $q = 0$
 $r = 1$

II. $q = 1$
 $p = 2$
 $r = 3$

III. $q = 2$
 $p = 3$
 $r = 0$

$p->next = \text{prim (mod 0)}$
 $q->next = \text{NULL}$
 $\text{prim} = p$

[Clear my choice](#)

Question 4

Answer saved

Marked out of
2.00

Flag question

Pentru o implementare cu pointeri a unei liste simplu înaintuite avem nevoie de o structură Nod cu minim două campuri, unul de date și unul

Select one:

- a. pointer la un index
- b. pointer la structura Nod
- c. de tip Index
- d. pointer la o clasa

Clear my choice

Next p

Previous page

Question 2

Answer saved

Marked out of
2.00

Flag question

Implementarea cozilor bazată pe liste ordonate este potrivită dacă prioritățile elementelor care se inserează au tendința de a fi apropriate ca valoare de prioritățea minima.

Select one:

- True
- False

Previous page

◀ Test Siruri de Caractere

Jump to...

False**Explicație:**

Implementarea cozilor bazată pe liste ordonate este potrivită pentru cazurile în care:

- Extragerea elementului cu prioritate minimă (sau maximă) este o operație frecventă, deoarece lista fiind ordonată, elementul cu prioritatea minimă/maximă poate fi găsit direct la un capăt al listei (complexitate $O(1)$).
- Nu conținează valoarea specifică a priorităților elementelor. Această implementare este mai puțin dependență de distribuția valorilor priorităților.

De ce este False?

- Dacă prioritățile elementelor inserate au tendința de a fi apropriate ca valoare de prioritățea minimă, nu este un argument care să justifice utilizarea listelor ordonate.
- Lista ordonată introduce un cost mare pentru inserție ($O(n)$), deoarece trebuie să găsească poziția corectă în funcție de prioritate, iar tendința valorilor similare nu reduce acest cost.

Pentru astfel de situații (priorități apropriate), o implementare mai eficientă poate fi utilizarea altor structuri de date, cum ar fi heap-urile.

Astfel, răspunsul este False.

Test Liste

Coada bazată pe prioritate este structura de date abstractă care permite inserția unui element și suprimarea celui mai putin prioritari.

* celui mai prioritari

Select one:

- True
- False

Next p

Question 1

Not yet answered

Marked out of
2.00

1 Flag question

In cele ce urmeaza este prezentat un algoritm incorrect, care ar trebui sa determine daca o secventa de paranteze rotunde este corecta.

declaratie stiva de caractere

cat timp (avem date de intrare)

{

citeste un caracter

daca (caracterul este '(')

adauga-l in stiva



altfel

daca (caracterul este ')' si stiva nu este goala)



extrage un caracter din stiva

altfel

scrie "Incorrect" si iesi

}

scrie "corect"

Pentru ce secventa incorrecta, codul va afisa "corect"

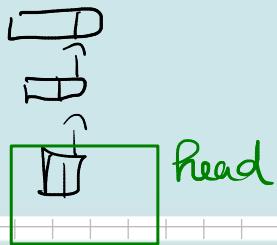
Talking

$\text{pt } ()()() \rightarrow \text{corect}$

Select one:

a. ✓b. 

Pentru o stiva implementata cu liste simplu inlantuite, daca in cazul operatiei de adaugare in stiva ("push") se adauga un nod la finalul listei, atunci prin operatia de extragere din stiva ("pop") se elimina un nod din capul listei.



Select one:

 True FalseDacă pe nivelul k ($k > 1$) al stivei am verificat toate valorile posibile, atunci?

- a. Se revine pe nivelul anterior
- b. Se sare un nivel
- c. Se trece pe nivelul urmator
- d. Algoritmul se încheie

Care din instructiunile de mai jos asigneaza in mod corect pointerului p adresa variabilei x?

Selectati raspunsul corect:

 a. $*p = \&x;$ $\& \rightarrow \text{dereferentiat}$ b. $p = \%x$ c. $p = \&x$

Sterge alegerea mea

Ce este o lista circulara?

Selectați răspunsul corect:

- a. O multi-lista folosita pentru a implementa figuri geometrice circulare (cerc, elipsa etc.).
- b. O lista in care ultimul nod indica catre primul prin campul sau de legatura implementat cu pointer
- c. O lista in care ultimul nod indica catre el insusi prin campul sau de legatura implementat cu pointer

[Șterge alegerea mea](#)

Care este sevența de cod corecta pentru a accesa informatie celui de-al doilea nod, daca *prim indica spre primul nod al listei?

```
struct nod{  
    int info;  
    struct nod*next;  
};  
struct nod *prim;  
....
```

Selectați răspunsul corect:

- a. prim->next->info
- b. prim->info->next
- c. prim->info
- d. prim->next->next->info

[Șterge alegerea mea](#)

Fie urmatorul pseudocod:

Se declara o stiva de caractere

Se primeste ca parametru cuvant de tip sir de caractere

cat timp (mai sunt caractere de citit din cuvant)

```
{  
    citeste un caracter  
    adauga caracterul in stiva (push)  
}
```

cat timp (stiva nu este goala)

```
{  
    extrage un caracter din stiva  
    afiseaza caracterul pe ecran  
}
```

Pentru datele de intrare "SDA" se va afisa

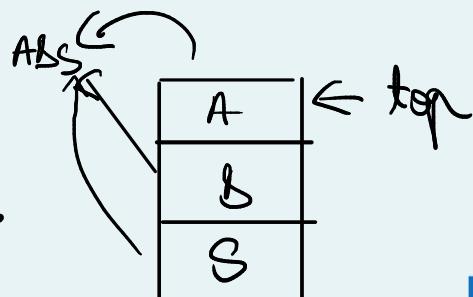
Selectați răspunsul corect:

- a. SDA
- b. ADSSDA
- c. SDAADS
- d. ADS

[Șterge alegerea mea](#)

push sir de caractere

A&S



Implementarea cozilor bazata pe liste ordonate este potrivita daca prioritatile elementelor care se insereaza au tendinta de a fi apropriate ca valoare de prioritatea minima.

Selectati o optiune:

Adevarat

Fals