

```

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 initializata data ca parametru de intrare, functia returneaza 1 daca si numai daca:

Select one:

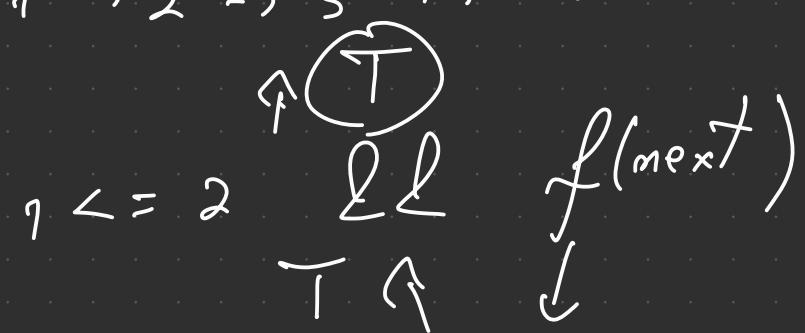
a. elementele listei sunt in ordine descrescatoare

b. elementele listei au valori diferite, doua cate doua

c. lista e vida sau are exact un element

d. elementele listei sunt in ordine crescatoare

$1 \rightarrow 2 \rightarrow 3 \rightarrow \text{NULL}$



$2 \leq 3$ f(next)

T \uparrow \downarrow

$3 \rightarrow \text{next} = \text{NULL}$ \checkmark

\uparrow

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

Select one:

True

False

→ La finalul listei

Test Liste

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

Select one:

- True
- False

Test Liste

Care din urmatoarele sortari poate fi folosita pentru a sorta o lista simplu, imantuita cu o eficienta a timpului de sortare cat mai mare?

Select one:

- a. Quicksort
- b. Sortarea secentuala
- c. Bubblesort
- d. Heapsort

[Next page](#)

Coada bazata pe prioritate ("priority queue") este structura de date abstracta care permite inserția unui element și suprimarea celui mai vechi element in mod direct (cu o complexitate egala cu $O(1)$).

Select one:

- True
- False

inserția după prioritate \rightarrow mutare neutral în spate

Implementarea cozilor bazate pe prioritate folosind liste neordonate este potrivita in situatii in care se fac multe insertii si mai putine extrageri.

Select one:

- True
- False

Fals? *inserție* $O(m)$
extragere $O(1)$

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

struct nod{

int info;

struct nod*next;

};

struct nod *prim;

....

* prim : capul liste
prim -> next : al doilea nod
prim -> next -> info : info nr 2

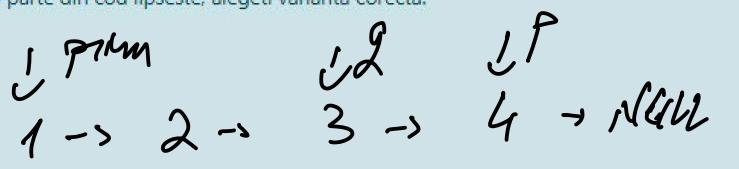
Select one:

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

Urmatoare functie primeste ca argument o lista simplu intantuita. Modifica lista, mutand ultimul element pe prima pozitie si returneaza lista modificata. O parte din cod lipseste, alegeți varianta corecta:

```
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;
}
```



$p \rightarrow \text{next} = \text{prim}$

$q \rightarrow \text{next} = \text{NULL}$

$\text{prim} = p$

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;

[Clear my choice](#)

Pentru o implementare cu pointeri a unei liste simplu înlăturăte avem nevoie de o structură Nod cu minim două câmpuri, 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

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

Implementarea cozilor bazată pe liste ordonate este potrivită dacă prioritatile elementelor care se inserează au tendința de a fi appropriate ca valoare de prioritatea minima.

Select one:

- True
- False

Test Liste

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

Select one:

- True
- False

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 -> daca stiva nu este goala -> corect

 scrie "incorrect" si iesi

)

scrie "corect"

Pentru ce secvență INCORRECTĂ, codul va afisa "corect".

Select one:

- a. ((0))

stack
(())

Talking:

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
- False

push \rightarrow adaugare la inceput
pop \rightarrow scoatere de la inceput

Dacă 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?

Selectați răspunsul corect:

- a. *p=8x;
- b. p=%x
- c. p=&x

[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)

{

 extrag un caracter din stiva

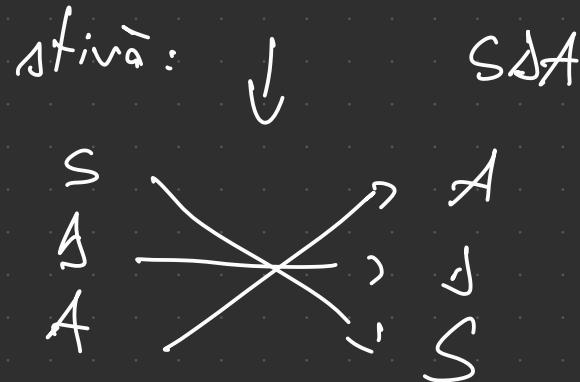
 afiseaza caracterul pe ecran

)

Pentru datele de intrare "SDA" se va afisa

Selectați răspunsul corect:

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



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

Selectați o opțiune:

- Adevărat
- Fals

[Șterge alegerea mea](#)