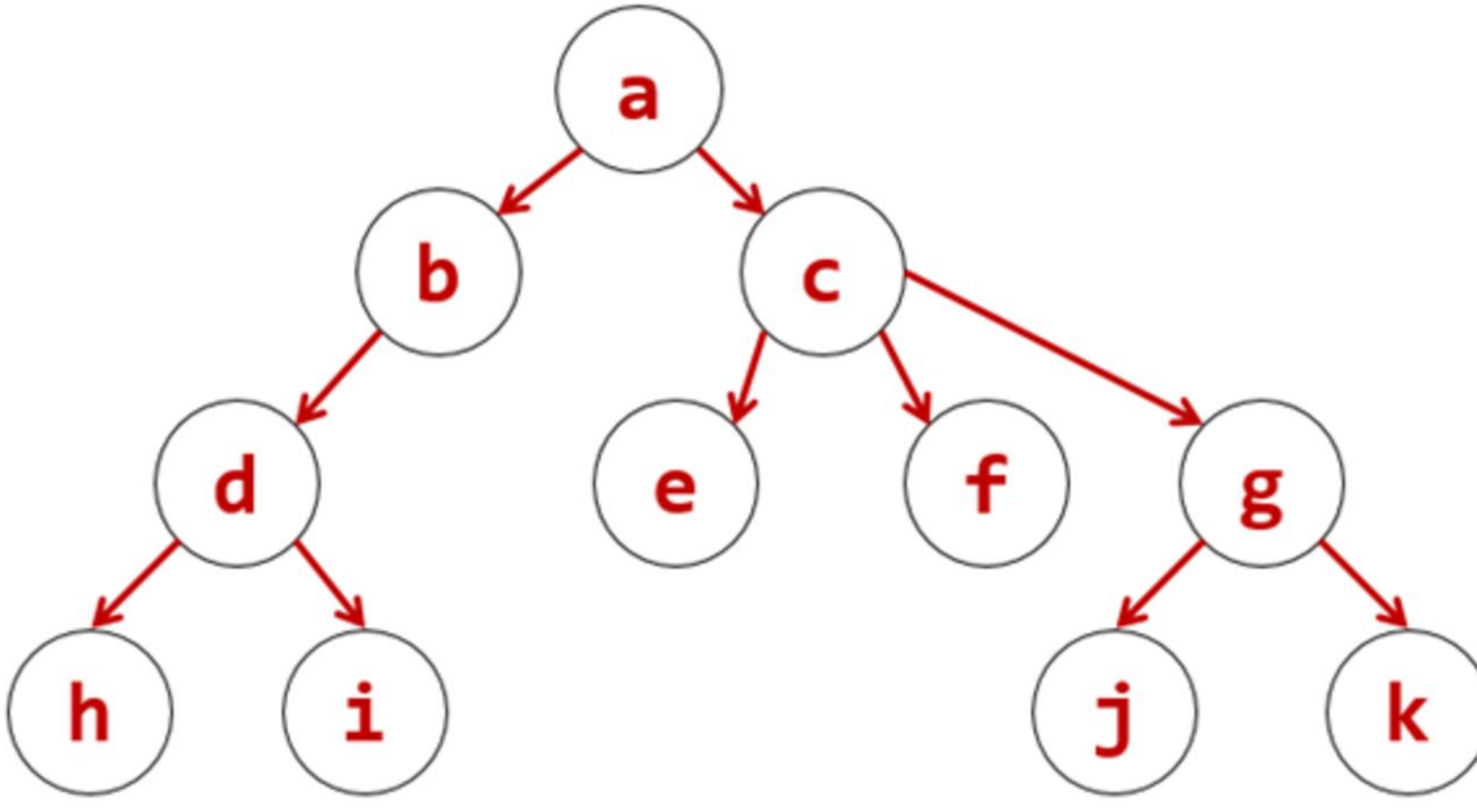
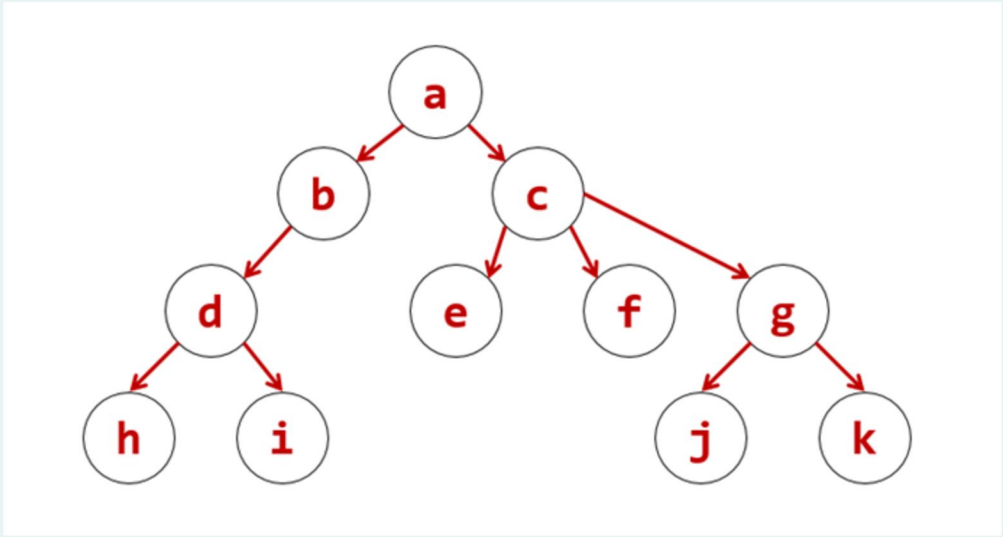
Zadano je stablo sa slike. Kojeg je stupnja navedeno stablo:




- a. 0
- b. 3
- c. 2
- d. 1

Zadano je stablo sa slike. Kolike je dubine navedeno stablo:




- a. 3
- b. 4 ✔

Prema definiciji binarnog stabla, maksimalni broj čvorova na razini k je:

- a. 2^k+1
- b. 2^k-1
- c. 2^(k+1)
- d. 2^k
- e. 2^(k-1) ✔

Prema definiciji binarnog stabla, maksimalni broj čvorova binarnog stabla dubine k je:

a. 2^k-1 ✔

Zadane su klase ListElement i List koje kojima je ostvaren rad s listom. Pažljivo proučite sav kôd, a posebice temeljito proučite funkcije PrintRec() i PrintRecInternal() klase List. Navedite koje programske linije treba umetnuti na pozicije a) i b) tako da ispis dobiven pozivom PrintRec() i ispis dobiven Print() nad nekom listom budu isti.

```
#include <iostream>
using namespace std;

template <typename T> class ListElement{
   public:
      T el;
      ListElement<T> *next;
};

template <typename T> class List{
   public:
      ListElement<T> *head = nullptr;

      void Add(T newVal){
         ListElement<T> *newEl = new (nothrow) ListElement<T>();
         newEl->el = newVal;
         newEl->next = nullptr;
         if(head == nullptr){
            head = newEl;
         }
         else{
            newEl->next = head;
            head = newEl;
         }
      }

      void Print(){
         ListElement<T> *tempHead = head;
         while(tempHead != nullptr){
            std::cout <<tempHead->el << " ";
            tempHead = tempHead->next;
         }
      }

      void PrintRec(){
         PrintRecInternal(head);
      }
      void PrintRecInternal(ListElement<T> * listElement){
         if(listElement != nullptr){
            //a)
            std::cout << listElement->el << " ";
            //b)
         }
      }
};

int main(){
   List<int> mojaLista;
   mojaLista.Add(4);
   mojaLista.Add(3);
   mojaLista.Add(2);
   mojaLista.Add(1);
   mojaLista.Print();
   std::cout << std::endl;
   mojaLista.PrintRec();
   std::cout << std::endl;
   return 0;
}
```

◉ b. a) //ništa
b) PrintRecInternal(listElement->next);

Zadane su klase ListElement i List koje kojima je ostvaren rad s listom. Pažljivo proučite sav kôd, a posebice temeljito proučite funkcije PrintRec() i PrintRecInternal() klase List. Navedite koje programske linije treba umetnuti na pozicije a) i b) tako da ispis dobiven pozivom PrintRec() i ispis dobiven Print() nad nekom listom budu isti.

```cpp
#include <iostream>
using namespace std;

template <typename T> class ListElement{
   public:
      T el;
      ListElement<T> *next;
};

template <typename T> class List{
   public:
      ListElement<T> *head = nullptr;

      void Add(T newVal){
         ListElement<T> *newEl = new (nothrow) ListElement<T>();
         newEl->el = newVal;
         newEl->next = nullptr;
         if(head == nullptr){
            head = newEl;
         }
         else{
            newEl->next = head;
            head = newEl;
         }
      }

      void Print(){
         ListElement<T> *tempHead = head;
         while(tempHead != nullptr){
            std::cout <<tempHead->el << " ";
            tempHead = tempHead->next;
         }
      }

       void PrintRec(){
          PrintRecInternal(&head);
       }
       void PrintRecInternal(ListElement<T> ** listElement){
          if(*listElement != nullptr){
             //a)
             std::cout << (*listElement)->el << " ";
             //b
          }
       }
};

int main(){
   List<int> mojaLista;
   mojaLista.Add(4);
   mojaLista.Add(3);
   mojaLista.Add(2);
   mojaLista.Add(1);
   mojaLista.Print();
   std::cout << std::endl;
   mojaLista.PrintRec();
   std::cout << std::endl;
   return 0;
}
```

d.

a) //ništa

b) PrintRecInternal(&((*listElement)->next));

Zadane su klase ListElement i List koje kojima je ostvaren rad s listom. Pažljivo proučite sav kôd, a posebice temeljito proučite funkciju X() klase List. Navedite što će se ispisati nakon izvođenja glavnog programa.

Napomena: Iako se program može pokrenuti, izvršava se i daje ispis, moguće je da se pokazivači unutar funkcije X() koriste na način koji je drukčiji od načina pokazanog u primjerima kôda s predavanja. Molimo da zato vrlo pažljivo pogledate kôd funkcije X() klase Lista.

```cpp
#include <iostream>
using namespace std;

template <typename T> class ListElement{
   public:
      T el;
      ListElement<T> *next;
};

template <typename T> class List{
   public:
      ListElement<T> *head = nullptr;

      void Add(T newVal){
         ListElement<T> *newEl = new (nothrow) ListElement<T>();
         newEl->el = newVal;
         newEl->next = nullptr;
         if(head == nullptr){
            head = newEl;
         }
         else{
            newEl->next = head;
            head = newEl;
         }
      }

      void X(T newVal){
         ListElement<T> *newEl = new (nothrow) ListElement<T>();
         newEl->el = newVal;
         newEl->next = nullptr;
         while(head != nullptr){
            if(head->el > newVal){
               break;
            }
            head = (head->next);
         }
      }
```

```
    void Print(){
        ListElement<T> *tempHead = head;
        while(tempHead != nullptr){
            std::cout <<tempHead->el << " ";
            tempHead = tempHead->next;
        }
    }
};

int main(){
    List<int> mojaLista;
    mojaLista.Add(4);
    mojaLista.Add(3);
    mojaLista.Add(2);
    mojaLista.Add(1);
    mojaLista.X(5);
    mojaLista.Print();
    std::cout << std::endl;
    return 0;
}
```

- a. 1 2 3 4 5
- b. 5 4
- c. 1 2 3 4
- d. 5
- e. ništa (prazna lista)

Zadane su klase ListElement i List koje kojima je ostvaren rad s listom. Pažljivo proučite sav kôd, a posebice temeljito proučite funkciju X() klase List. Navedite što će se ispisati nakon izvođenja glavnog programa.

Napomena: Iako se program može pokrenuti, izvršava se i daje ispis, moguće je da se pokazivači unutar funkcije X() koriste na način koji je drukčiji od načina pokazanog u primjerima kôda s predavanja. Molimo da zato vrlo pažljivo pogledate kôd funkcije X() klase Lista.

```
#include <iostream>
using namespace std;

template <typename T> class ListElement {
 public:
   T el;
   ListElement<T> *next;
};

template <typename T> class List {
 public:
   ListElement<T> *head = nullptr;

   void Add(T newVal) {
      ListElement<T> *newEl = new (nothrow) ListElement<T>();
      newEl->el = newVal;
      newEl->next = nullptr;
      if (head == nullptr) {
         head = newEl;
      } else {
         newEl->next = head;
         head = newEl;
      }
   }

   void X(T newVal) {
      ListElement<T> *newEl = new (nothrow) ListElement<T>();
      newEl->el = newVal;
      newEl->next = nullptr;
      while (head != nullptr) {
         if (head->el > newVal) {
            break;
         }
         newEl->next = head;
         head = head->next;
      }
      head = newEl;
   }
```

```
    void Print() {
        ListElement<T> *tempHead = head;
        while (tempHead != nullptr) {
            std::cout << tempHead->el << " ";
            tempHead = tempHead->next;
        }
    }
};

int main() {
    List<int> mojaLista;
    mojaLista.Add(4);
    mojaLista.Add(3);
    mojaLista.Add(2);
    mojaLista.Add(1);
    mojaLista.X(5);
    mojaLista.Print();
    std::cout << std::endl;
    return 0;
}
```

a. 5 4

U red A je stavljeno n elemenata. U red B je stavljeno također n elemenata. Najbolja moguća složenost funkcije (Θ - Theta) koja prazni oba reda je:

a. `Theta( logn )`

b. `Theta( n*n )`

c. `Theta( n )` ✔

Na stog je stavljeno n elemenata. Najbolja moguća složenost funkcije (Θ - Theta) koja izbacuje svaki parni element sa stoga je:

a.

Theta( n*n )

b. Theta( logn )

c. Theta( n ) ✔

Algoritam Quick sort (uzlazno) implementiran je korištenjem metode medijana tri elementa, uz skrivanje stožera. Ako je zadano polje p={9, 5, 1, 4, 3, 2, 6, 7, 1}, koji element će se nalaziti na poziciji i=4 nakon što se obavi prvo skrivanje stožera u radu algoritma nad poljem?
Napomena: indeksi polja idu od 0 do 8.

a. 7 ✔

Algoritam Quick sort (uzlazno) implementiran je korištenjem metode medijana tri elementa, uz skrivanje stožera. Ako je zadano polje p={9, 5, 1, 4, 3, 2, 6, 7, 1}, koji element će se nalaziti na poziciji i=7 nakon što se obavi prvo skrivanje stožera u radu algoritma nad poljem? Napomena: indeksi polja idu od 0 do 8.

a. 9

b. Niti jedan odgovor nije točan

c. 3

Kako izgleda polje p={9, 5, 1, 4, 3, 2, 6, 7, 11, 1} nakon sortiranja algoritmom Shell sort (silazno) samo za korak hk=4?

a. Niti jedan odgovor nije točan

b. 11, 5, 6, 7, 9, 2, 1, 4, 3, 1

Kako izgleda polje p={9, 5, 1, 4, 3, 2, 6, 7, 11, 1} nakon sortiranja algoritmom Shell sort (uzlazno) samo za korak hk=4?

a. 3, 1, 1, 4, 9, 2, 6, 7, 11, 5

Koja je od navedenih tvrdnji istinita?

a. Merge sort za sortiranje ne treba dodatno polje koje je veličine polja kojeg sortira

b. Shellov sort tipično koristi metodu medijana za odabir stožera

c. Merge sort ne radi ništa brže, ako je ulazni niz već sortiran ✔

U red je stavljeno n elemenata. Najbolja moguća složenost funkcije (Θ) koja izbacuje svaki parni element iz reda:

a. Θ(n*n)

b. Θ( n ) ✔

Prvi stožerni element (Quicksort) prilikom korištenja metode medijana temeljem 3 elemenata za polje {2, 4, 6, 6, 8, 5, 3} je:

- a. 5
- b. 6
- c. 3 ✔

Na stog A je stavljeno n elemenata (cijelih brojeva). Na stog B je stavljeno također n elemenata (cijelih brojeva). Najbolja moguća složenost funkcije (Θ) koja pronalazi najveći element od svih elemenata na stogovima A i B je:

a. Θ(n*n)

b. Θ( n )

Koji od navedenih sortova ima složenost u najlošijem slučaju Θ(n*logn)?

a. Insertion sort

b. Shellov sort

c. Quick sort

d. Merge Sort ✔

Koja je od navedenih tvrdnji istinita?

a. U Quick sortu se polje uvijek dijeli na dvije polovice

b. Shellov sort uvijek završava s korakom 1 ✔

Koja je od navedenih tvrdnji istinita za analizu vremena izvođenja za uzlazno sortiranje korištenjem Insertion sorta?

a. Najbolji slučaj: sortiran niz