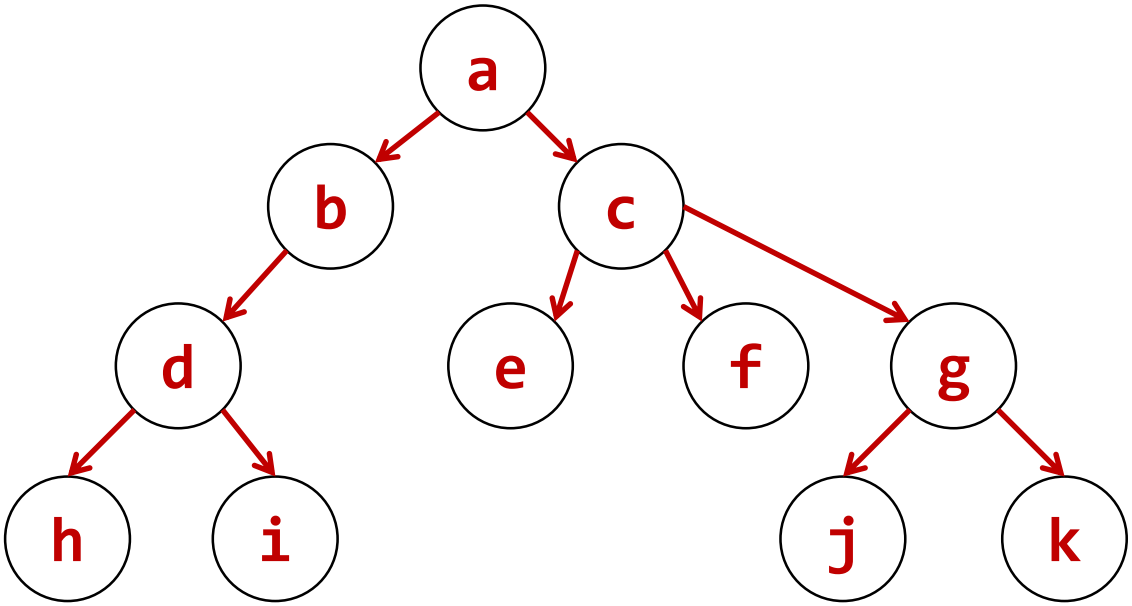
Algoritmi i strukture podataka

- predavanja -

9. Stabla

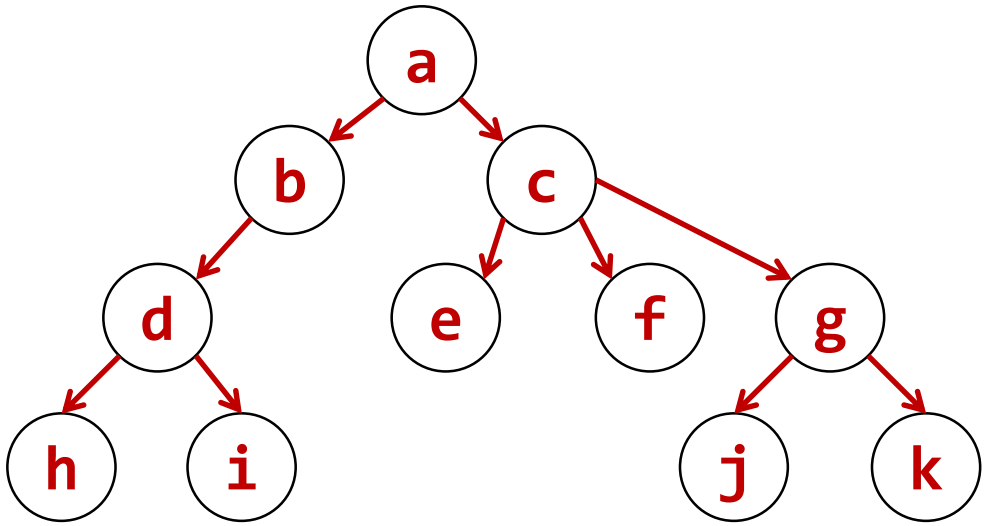
Svojstva stabla

- stablo je konačan skup čvorova sa svojstvima:
  - postoji poseban čvor koji se naziva **korijen** (*root*)
  - ostali čvorovi su podijeljeni u *k* disjunktnih podskupova T_1, .., T_k, od kojih je svaki stablo
  - T_1, .., T_k se nazivaju i **podstabla**
- Primjer:



Osnovni pojmovi - I

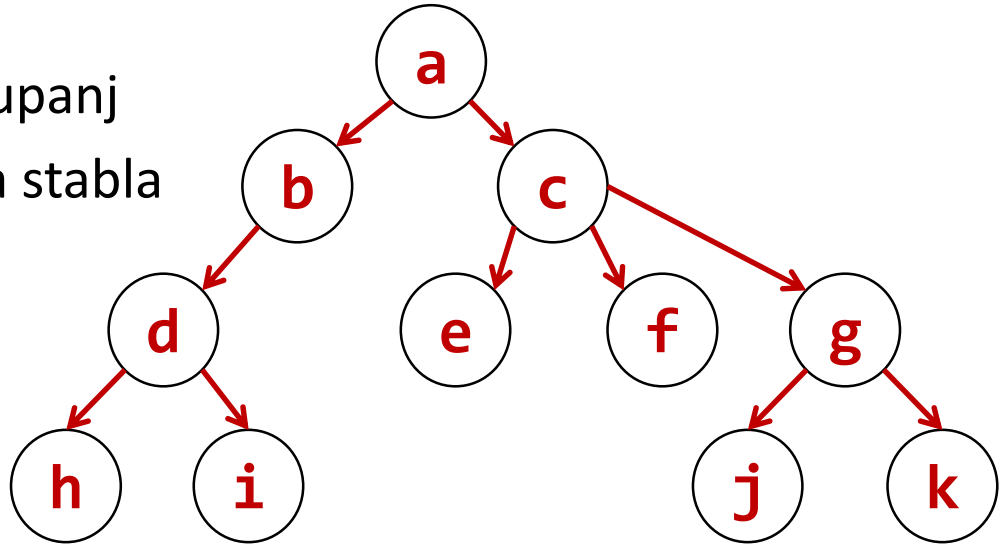
- *a* je korijen stabla
- stupanj čvora *a* je 2
  (stupanj je broj podstabala nekog čvora, npr. čvor *c* ima stupanj 3)



- skup {*h*, *i*, *e*, *f*, *j*, *k*} je skup krajnjih čvorova (listova)
- korijeni podstabala nekog čvora su djeca tog čvora (npr. čvorovi *e*, *f*, *g* su djeca od *c*), a taj čvor nazivamo roditeljem (npr. *g* je roditelj od *j*).
  - slični pojmovi se koriste i za ostale odnose (*djed, braća, predci*)

Osnovni pojmovi - II

- stupanj stabla je maksimalni stupanj među stupnjevima svih čvorova stabla (u ovom primjeru 3)



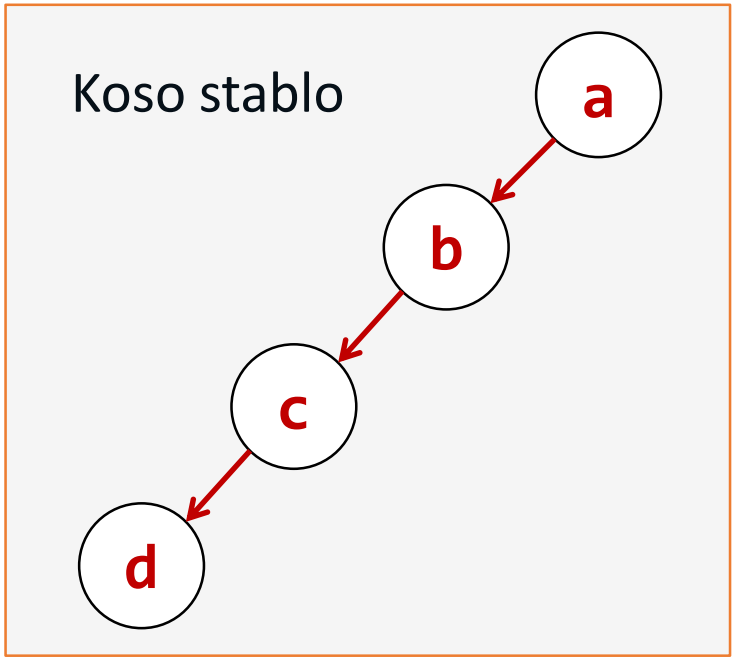
- razina (*level*) nekog čvora određuje se iz definicije da je korijen razine 1, a da su razine djece nekog čvora razine *k* jednaki *k*+1
- dubina (*depth*) stabla je jednaka maksimalnoj razini nekog čvora u stablu

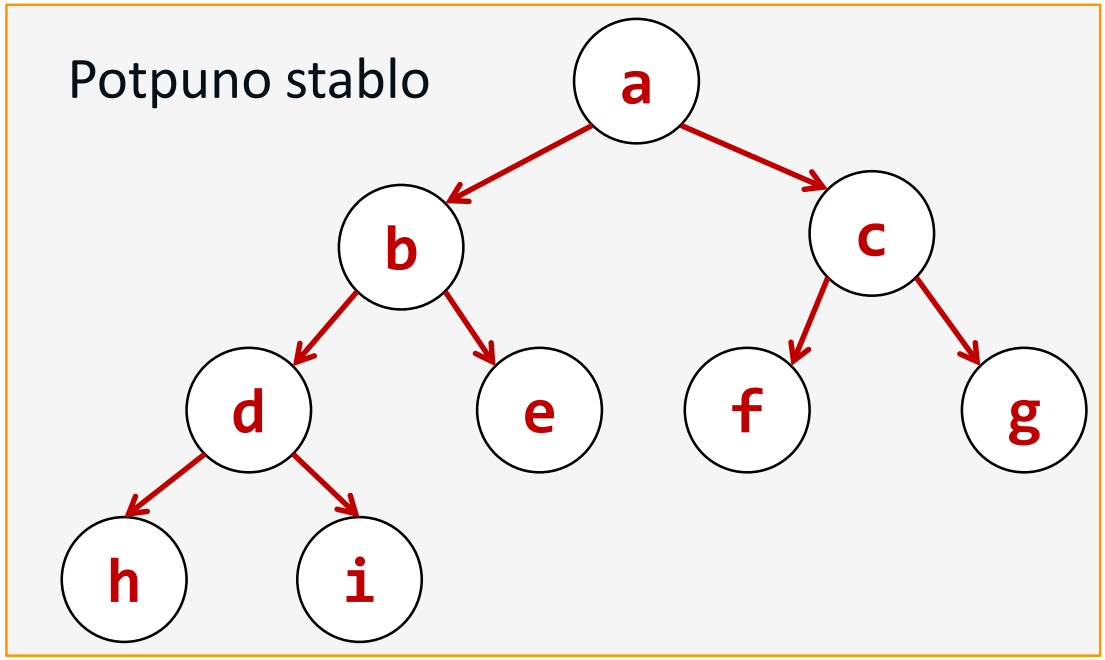
Rekurzivna stabla u prirodi

Najveće zmajevo drvo na svijetu...

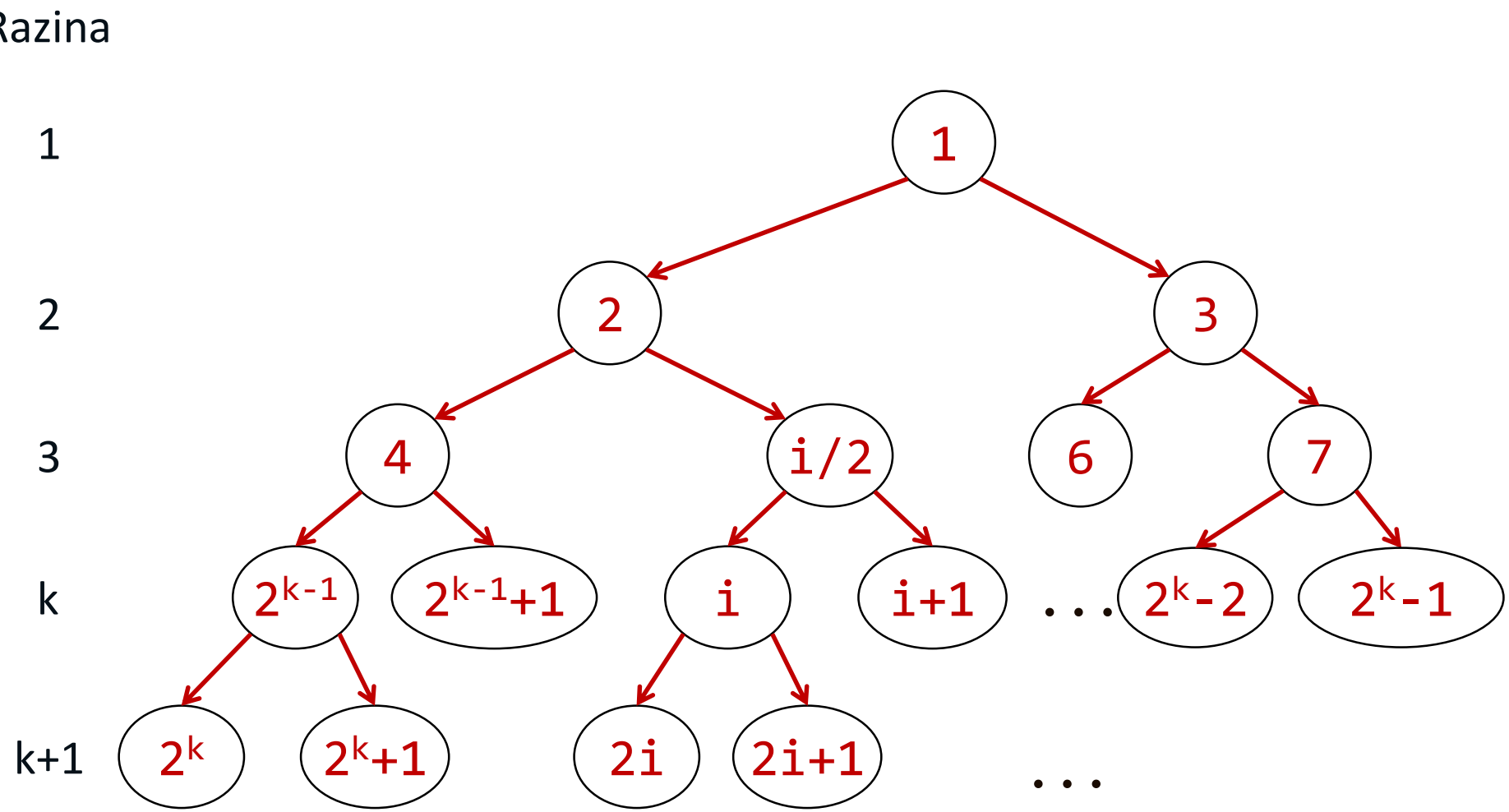
Binarno stablo - I

- binarno stablo je stablo koje se sastoji od nijednog, jednog ili više čvorova drugog stupnja
  - kod binarnog stabla razlikujemo lijevo i desno podstablo svakog čvora
  - nazivlje uvedeno za stabla koristi se i kod binarnih stabala





Binarno stablo - II

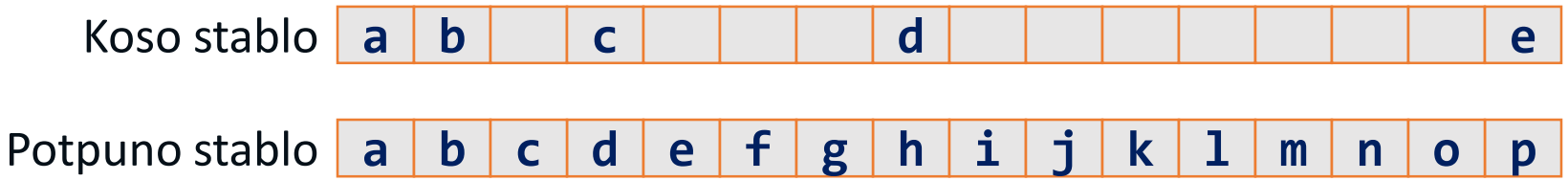


Binarno stablo - III

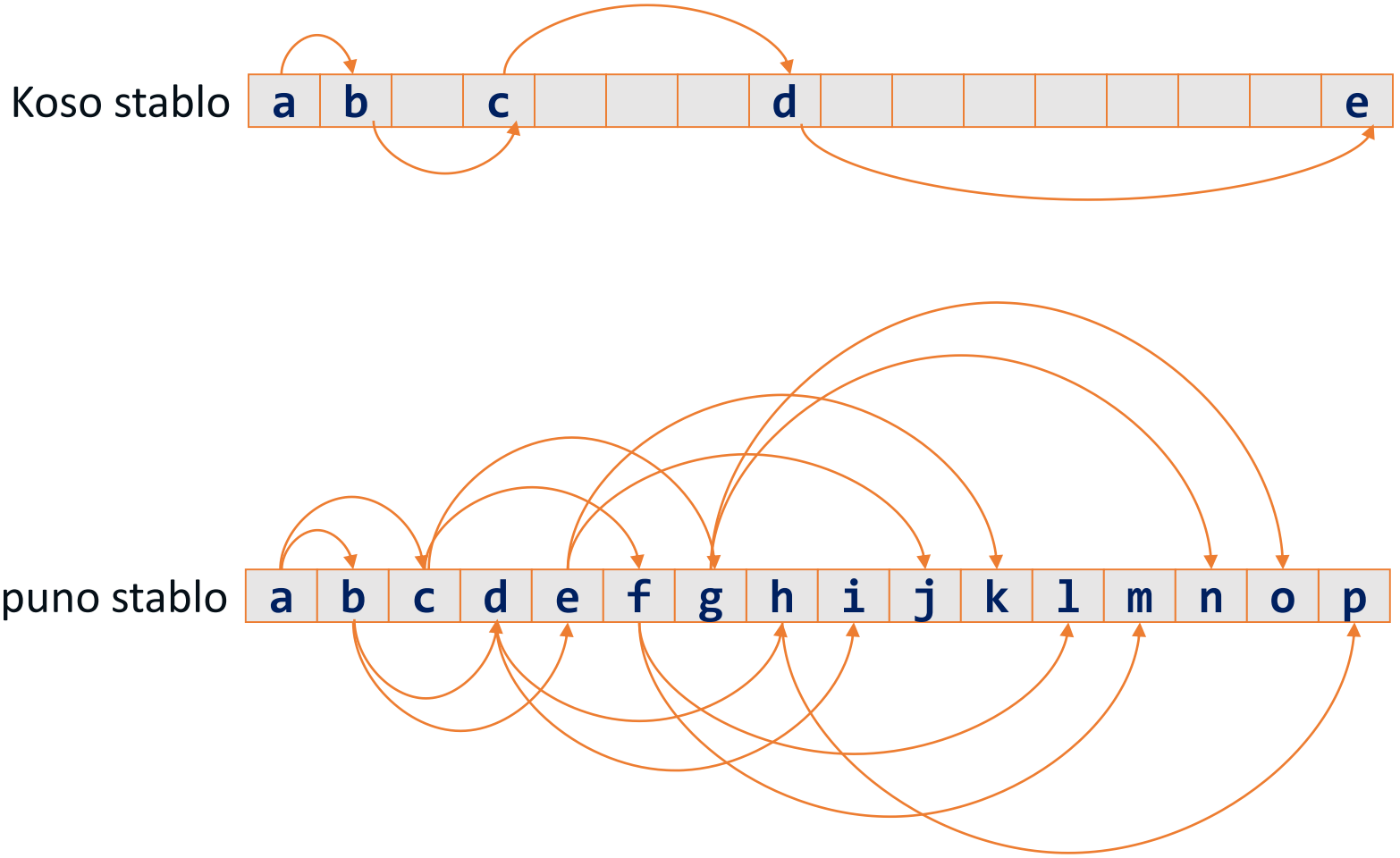
- iz definicije binarnog stabla slijede zaključci da je:
  - maksimalni broj čvorova na ***k***-toj razini jednak je 2^{k-1}
  - maksimalni broj čvorova binarnog stabla dubine ***k*** jednak je $2^k - 1$ za $k > 0$
  - stablo koje je visine ***k*** i ima $2^k - 1$ elemenata naziva se **puno** (*full*) binarno stablo
  - binarno stablo s ***n*** čvorova dubine ***k*** je **potpuno** (*complete*) ako i samo ako njegovi čvorovi odgovaraju čvorovima punog binarnog stabla dubine ***k*** koji su numerirani od **1** do ***n***
    - kao posljedica, razlika razina krajnjih čvorova potpunog stabla najviše je jedan

Prikaz stabla statičkom strukturom polje

- potpuno se binarno stablo jednostavno prikazuje jednodimenzionalnim poljem, bez podataka za povezivanje i koristi se pravilima za određivanje odnosa u stablu
  - korištenje polja počet će od člana s indeksom 1 radi jednostavnosti izraza
- problem kod prikaza stabla statičkom strukturom polje:
  - teško umetanje i brisanje čvorova, jer ti zahtjevi mogu tražiti pomicanje puno elemenata



Koso i potpuno stablo



Pravila kod prikaza stabla poljem

Potpuno stablo

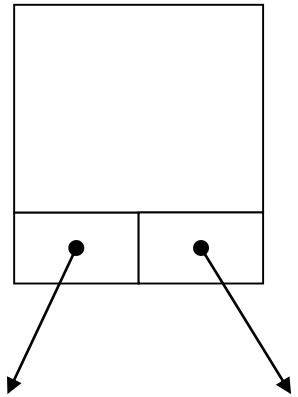
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p

- pravila za ***i***-ti čvor u potpunom binarnom stablu s ***n*** čvorova su:
  - roditelj(i) = $\lfloor i / 2 \rfloor$ za $i \neq 1$;
    - kada je $i = 1$, čvor i je korijen, pa nema roditelja
  - lijevo_dijete(i) = $2 * i$ ako je $2 * i \leq n$;
    - kada je $2 * i > n$, čvor i nema lijevog djeteta
  - desno_dijete(i) = $2 * i + 1$ ako je $2 * i + 1 \leq n$;
    - kada je $2 * i + 1 > n$ čvor i nema desnog djeteta
- ovako se mogu prikazati sva binarna stabla, ali se tada memorija ne koristi učinkovito
- najgori slučaj su **kosa** (*skewed*) stabla koja koriste smo ***k*** lokacija od $\mathbf{2^k - 1}$ lokacija predviđenih za to stablo

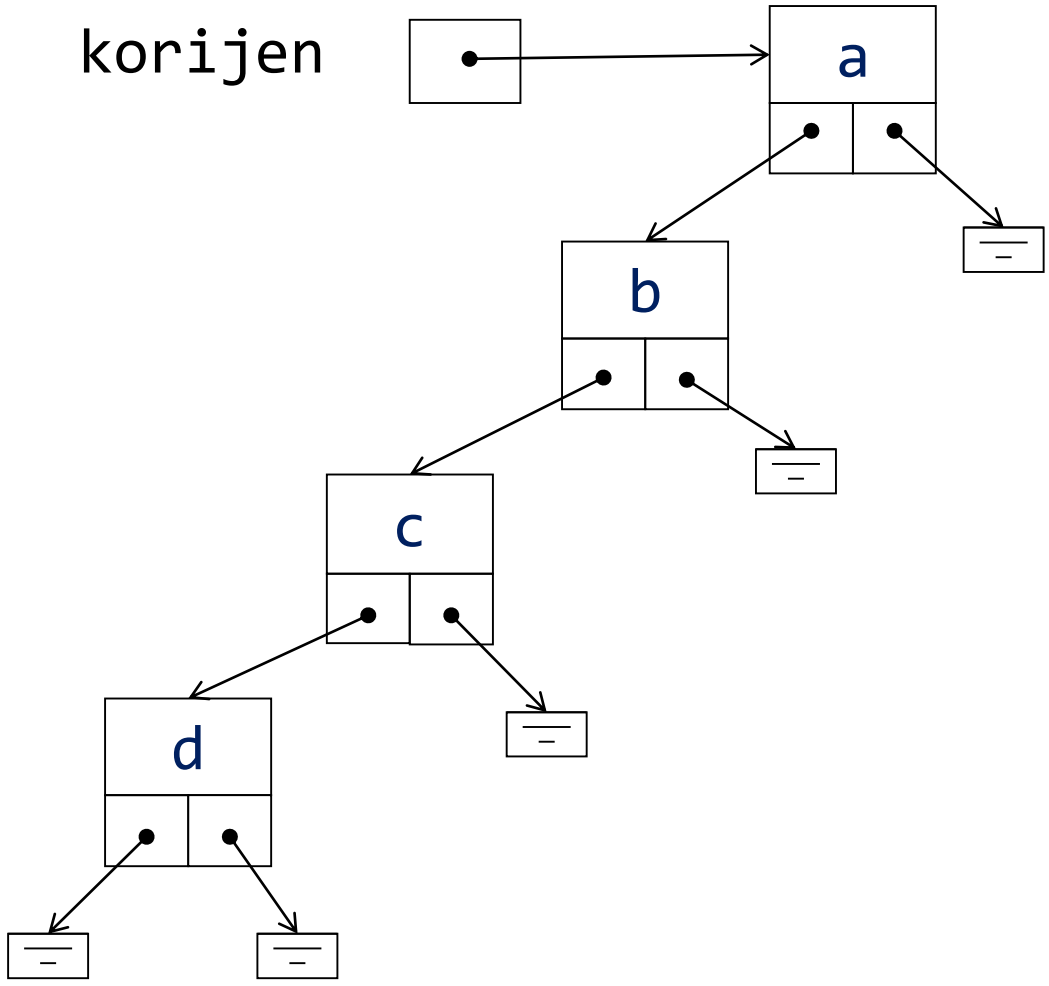
Prikaz stabla dinamičkom strukturom

- problem se rješava korištenjem strukture s pokazivačima
  - ovakva struktura se često upotrebljava i zadovoljava većinu potreba
  - ponekad se dodaje pokazivač na roditelja

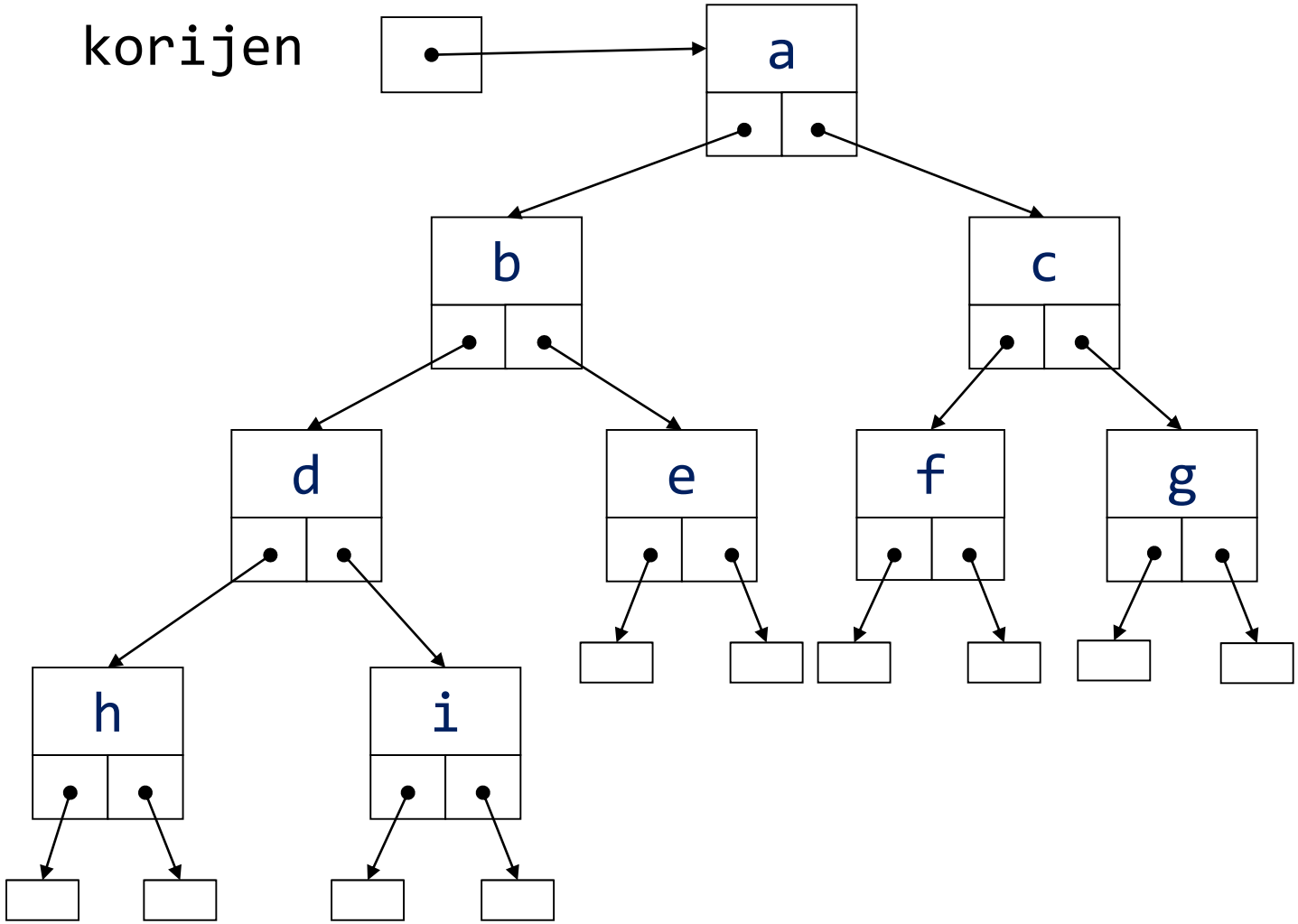
```
template <typename T> struct Node {
   T item;
   shared_ptr<Node<T>> left;
   shared_ptr<Node<T>> right;
    Node(const T &newItem) : item(newItem),
                             left(nullptr),
                             right(nullptr) {}
};
```



Koso stablo



Potpuno stablo



k-stabla

- prirodna generalizacija binarnih stabala su k-stabla
  - k predstavlja stupanj stabla, $k > 2$, s istim mogućnostima prikazivanja
- općenita stabla, s raznim stupnjevima, mogu se transformirati u binarna stabla
  - to rezultira manjim i učinkovitijim algoritmima te manjim potrebama za memorijom

Stablo za traženje

BSTree.cpp
BSTree2.cpp

- može se oblikovati stablo za traženje (uređeno stablo) po nekom od podataka (ključu) koji se upisuju u pojedini čvor
- upis novog čvora počinje pretragom od korijena stabla
- uspoređuje se već upisani podatak u čvorovima s novim podatkom:
  - ako je ključ novog čvora manji od ključa upisanog čvora, nastavlja se usporedba u lijevom podstablu
  - ako je ključ novog čvora veći od ključa upisanog čvora, nastavlja se usporedba u desnom podstablu
  - ako je ključ novog čvora jednak ključu upisanog čvora, dojavi pogrešku (iznimka)
  - ako upisani čvor nema podstablo u traženom smjeru, novi čvor postaje dijete upisanog čvora

Binarno stablo za pretraživanje

BSTree2.cpp

```
template <typename T> class BSTree {
protected:
   shared_ptr<Node<T>> root;
public:
   BinarySearchTree() : root(nullptr) {}
   void insert(shared_ptr<Node<T>> &node, const T &newItem)
…
};
```

Dodavanje elementa u stablo

```
template <typename T>
void BSTree<T>::insert(shared_ptr<Node<T>> &node, const T &newItem) {
   if (node) {
      if (node->item < newItem) {
         insert(node->right, newItem);
      } else if (node->item > newItem) {
         insert(node->left, newItem);
      } else {
         throw std::invalid_argument("Error: Item "
            + std::to_string(newItem)
            + " already exists in the tree.");
      }
   } else {
      node = std::make_shared<Node<T>>(newItem);
   }
}
```

rekurzivno traži mjesto za novu stavku

ako je element već u polju

ako je mjesto prazno, vrati novi čvor

Pretraživanje stabla

```
template <typename T>
bool BSTree<T>::search(shared_ptr<Node<T>> &node, const T &item) {
   if (node) {
      if (node->item < item) {
         return search(node->right, item);
      } else if (node->item > item) {
         return search(node->left, item);
      } else { // found item
         return true;
      }
   }
   return false;
}
```

spusti se niz odgovarajuće podstablo

element je pronađen

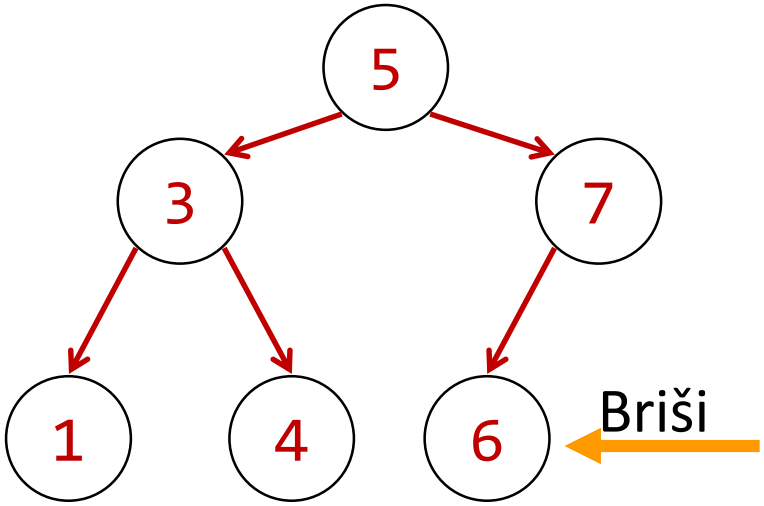
osnovni slučaj – prazno (pod)stablo; tražena vrijednost nije nađena

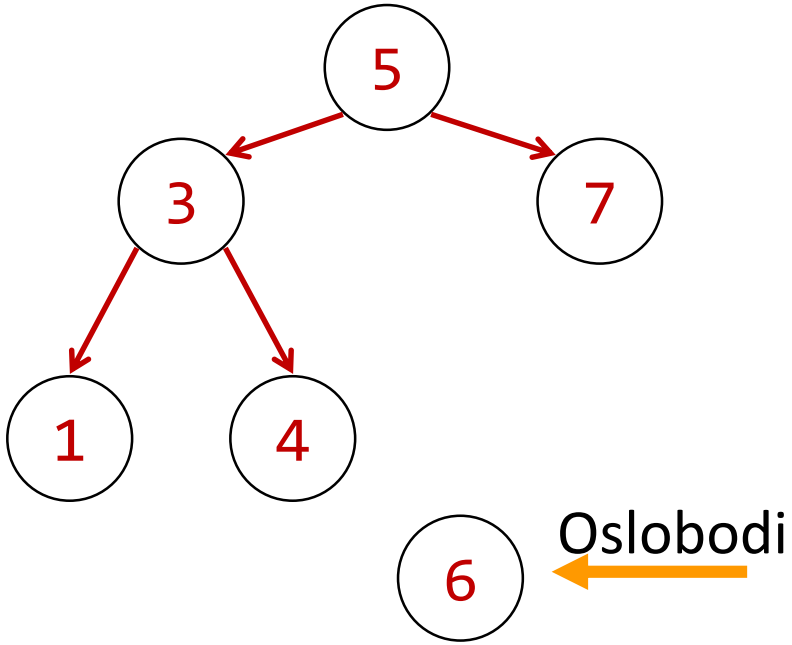
Obilazak stabla

- postoje 3 standardna načina obilaska stabla kojima se osigurava da je svaki čvor bio "posjećen"
  - *inorder:* lijevo podstablo → korijen → desno podstablo
  - *preorder:* korijen → lijevo podstablo → desno podstablo
  - *postorder:* lijevo podstablo → desno podstablo → korijen
- radi se o rekurzivnim postupcima koji sežu do listova stabla, a zatim povratci iz rekurzije predstavljaju kretanje prema korijenu stabla
- drugi način *inorder* (analogno i za ostale načine obilaska stabla):
  - desno podstablo → korijen → lijevo podstablo

Brisanje čvora - list

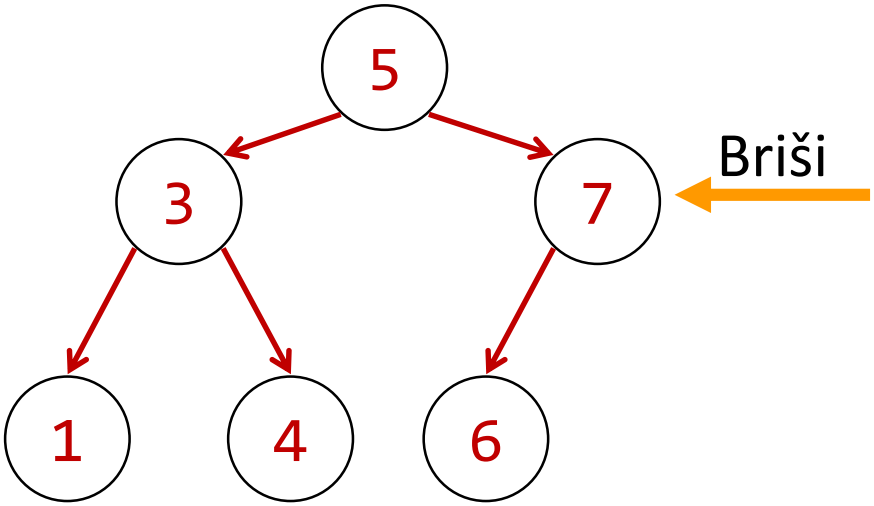
- najjednostavniji slučaj je brisanje lista, npr. 6.

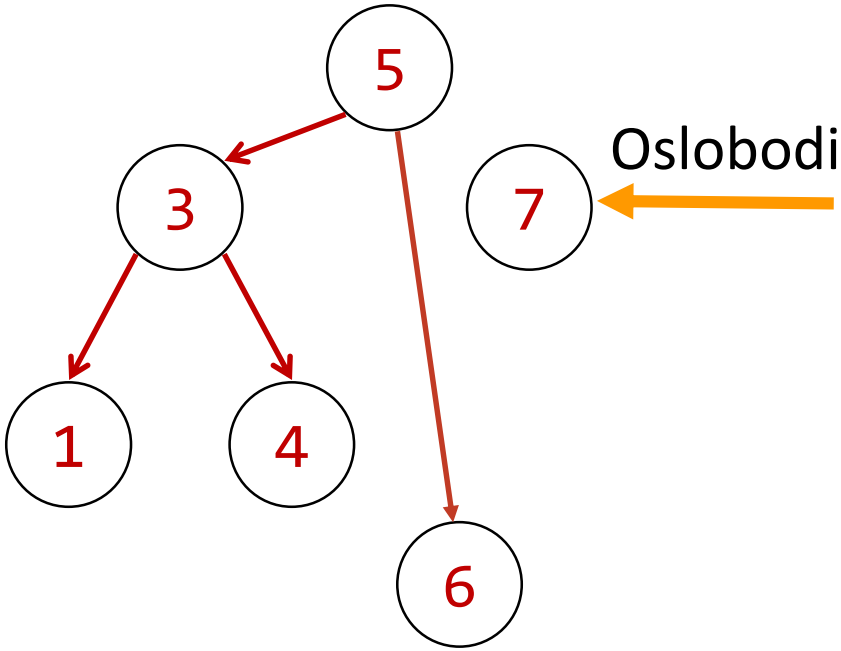




Brisanje čvora – jedno dijete

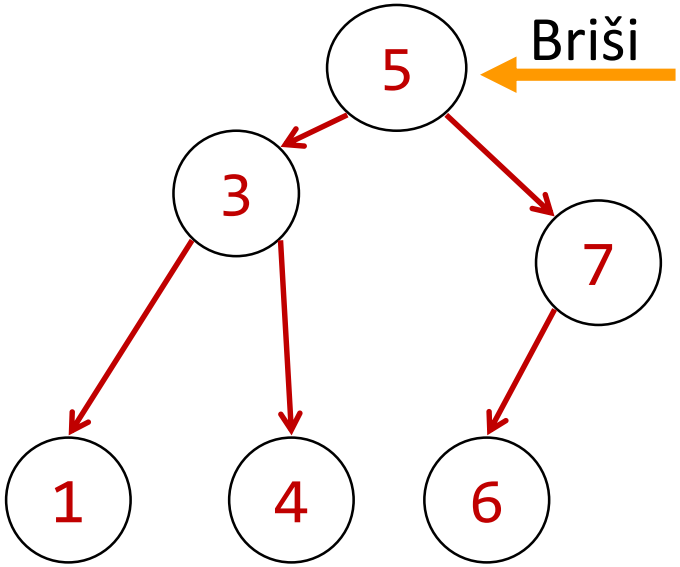
- jednostavan je i slučaj je brisanje čvora s jednim djetetom, npr. 7

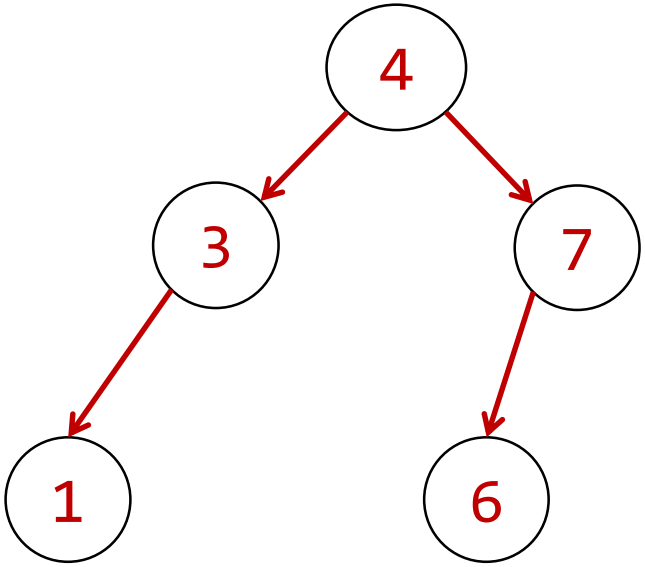




Brisanje čvora – dvoje djece

- složeniji je slučaj je brisanje čvora dvoje djece, npr. 5





Zadaci za vježbu (1)

- Napišite program koji će:
  a. ispisati broj čvorova u stablu
  b. ispisati dubinu stabla
  c. ispisati vrijednost najmanjeg i najvećeg elementa stabla
  d. napraviti i ispisati zrcalnu kopiju zadanog stabla
  e. za dva zadana stabla, napisati jesu li identična ili ne

Zadatci za vježbu (2)

Napisati funkciju za ispis elementa memorijski rezidentnog već oblikovanog uređenog binarnog stabla u čije čvorove su upisani

- cijena artikla (cijeli broj)
- naziv artikla (string)

Stablo je uređeno po cijeni artikala; lijevo jeftiniji, desno skuplji.

Ispis treba biti poredan po cijeni od najjeftinijeg do najskupljeg artikla. Koristiti razred BinarnoStablo (str. 17).

Zadatci za vježbu (3)

- U binarno stablo pohranjuje se niz podataka:

  12, 15, 5, 3, 7, 2, 18, 11

  a) treba nacrtati sortirano binarno stablo (lijevo manji, desno veći), ako je stablo popunjavano redom kako su dolazili podatci
  b) poredati ulazne podatke tako da nastupi neki od najlošijih slučajeva
  c) nacrtati binarno stablo koje predstavlja najbolji slučaj
  d) koliko je apriorno vrijeme izvođenja za pronalaženje pojedinog čvora za b)
  e) koliko je apriorno vrijeme izvođenja za pronalaženje pojedinog čvora za c)

Zadatci za vježbu (4)

Koristiti razred BinarnoStablo (str. 17).

- U neko memorijski rezidentno binarno stablo upisane su šifre (string). Napisati člansku funkciju koja će provjeriti postojanje neke zadane šifre u stablu. Funkcija vraća logičku istinu, ako podatak postoji u stablu, a laž, ako ne postoji.
- U neko memorijski rezidentno uređeno binarno stablo (lijevi čvor manja vrijednost, desni čvor veća vrijednost) upisani su matični brojevi (cijeli broj kao ključ) i visine osoba u cm (tipa `float`). Napisati člansku funkciju koja će izračunati prosječnu visinu osoba (tipa `float`) čiji se podatci nalaze upisani u stablu.