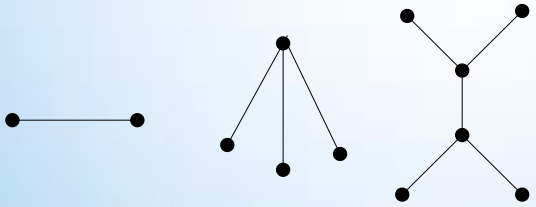
# STABLA

- ***Stabla*** ili ***drveta*** predstavljaju najjednostavniju, ali i najvažniju klasu grafova.
- **Na primer**:

  porodična stabla su je jedna vrsta stabla, organizaciona struktura firme su takođe vrsta stabala i td.

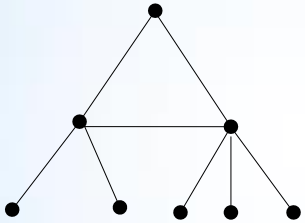
- Postoji više ekvivalentnih definicija stabla.

  ***DEF:***
- ***Stablo*** ili ***drvo*** je povezan graf koji ne sadrži cikluse ili konture.

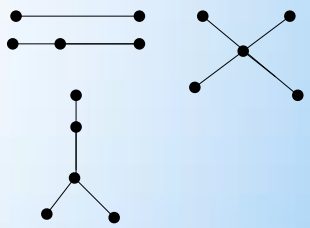


- ***Stablo*** je povezan graf sa *v* čvorova i *e=v-1* grana**.**
- ***Stablo*** je minimalno povezan graf.

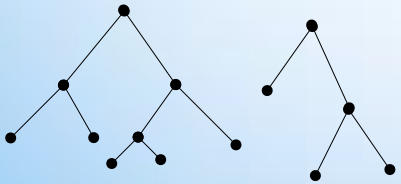
- **Primer:**
  Graf na sledećoj slici nije stablo jer sadrži konturu- ciklus.



- ***Stablo*** sadrži bar dva čvora stepena 1.
- Za svaki par čvorova *(u,v)* postoji ***tačno jedan put*** koji ih povezuje.
- Udaljavanjem bilo koje grane iz stabla dobija se nepovezan graf.
- Ako se u stablo uključi proizvoljna grana, dobija se graf koji može da ima ima tačno jednu konturu.
- Svaki neorijentisan multigraf bez petlji sadrži kao delimični graf u obliku stabla.
- ***Stablo*** je bipartitivni graf.



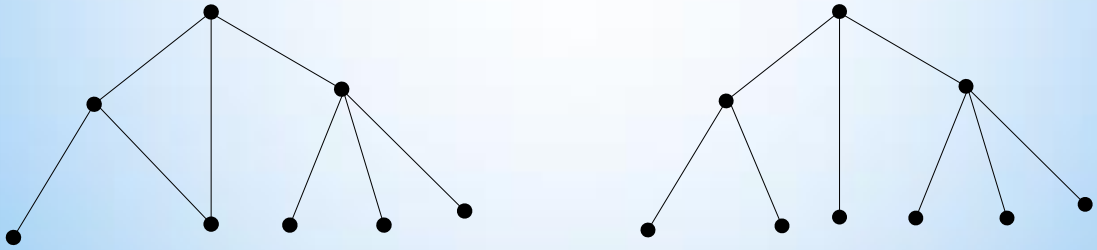
- ***Šuma*** je graf kome su komponente stabla.



# RAZAPINJUĆA STABLA

- Stablo koje se dobija iz grafa G uklanjanjem određenog broja grana iz grafa, a da graf ostane povezan, sa istim brojem čvorova zove se ***razapinjuće stablo ili razapeta stabla.*** *(engl. spanning tress)*

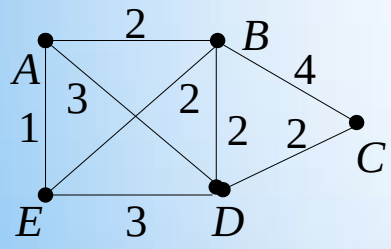
- Svaki povezan graf ima razapeto stablo.

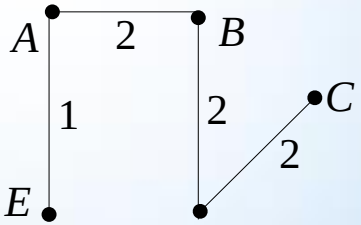


- Ako iz datog grafa želimo da dobijemo razapeto stablo, problem je u suštini jednostavan , ali obično se prave razapeta stabla koja ispunjavaju neke uslove, napr. min ili max.

**Primer:**

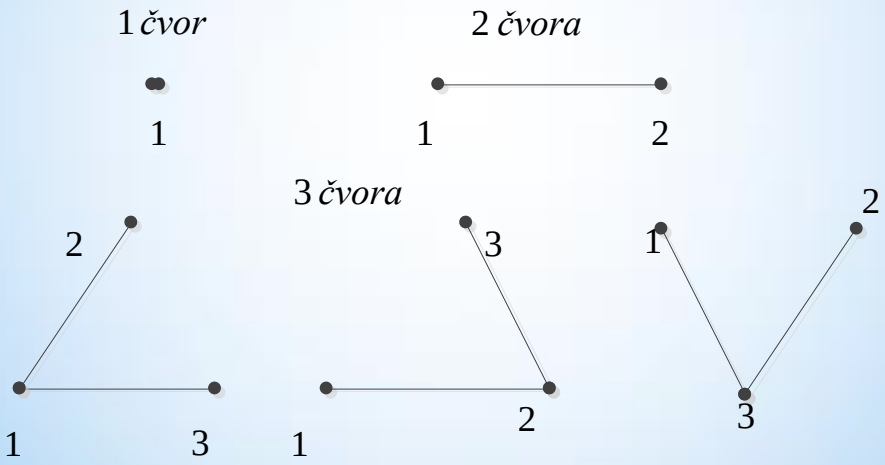
- Grafu sa slike, odgovara na primer sledeće min razapeto stablo.



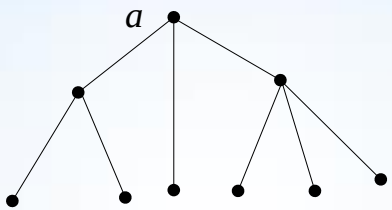

- Za dobijanje razapetih stabala postoje razni algoritmi, ali najpoznatiji su Primov i Kruskalov algoritam.

- **Primer:**
  Odrediti razapinjuća stabla sa 1,2,3 čvora.

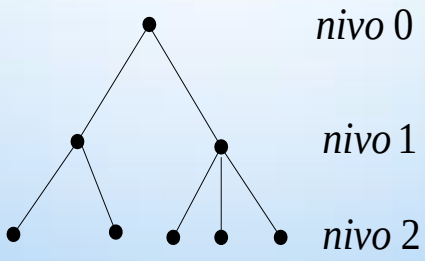


# KOREN STABLA

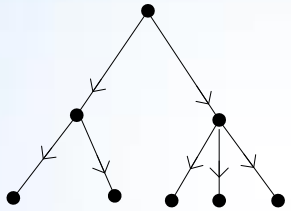
- Stablo u kome je jedan čvor posebno označen naziva se ***koreno stablo*.**
- Čvor na vrhu stabla naziva se ***korenom.***




- Svaki čvor korenog stabla povezan je jedinstvenim putem za koren stabla.
- Broj grana u ovom putu predstavlja ***nivo*** tog čvora.
- Koren stabla ima nivo 0, a najveći nivo imaju od korena najudaljeniji čvorovi.



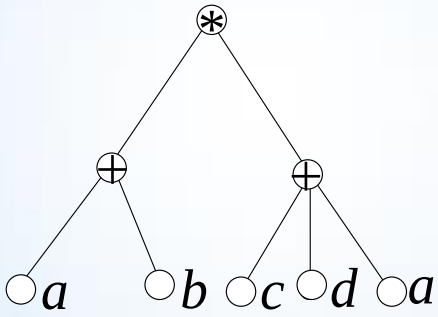
- Koreno stablo može da bude i ***orijentisano*****.** Grane se orijentišu od čvorova manjih nivoa, ka čvorovima viših nivoa. Ulazni stepen korena je 0.



- Čvorovi do kojih vode grane koje polaze iz čvora *x,* nazivaju se ***sinovi*** čvora *x*, a sam čvor *x* je njihov ***otac*****.**
- Svi predhodni čvorovi u odnosu na *x* nazivaju se ***pretci***, a naredni njihovi ***potomci.***
- Čvor bez dece naziva se ***list*****.** Listovi su završni čvorovi.
- ***Listovi*** su čvorovi stepena 1.
- Ostali čvorovi se nazivaju ***unutrašnjim čvorovima*****.**
- ***Visina stabla*** je dužina najdužeg mogućeg puta od korena do lista.

**Primer:**

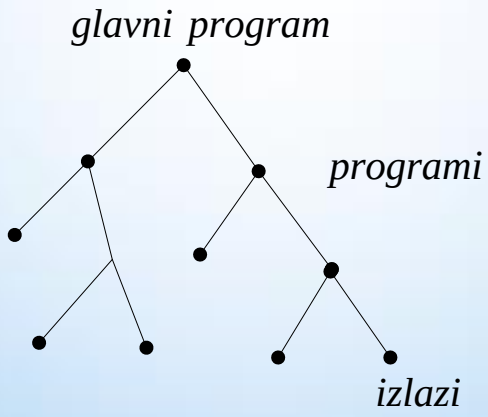
- Napisati koreno stablo koje predstavlja formulu $(a+b)\cdot(c+d+a)$



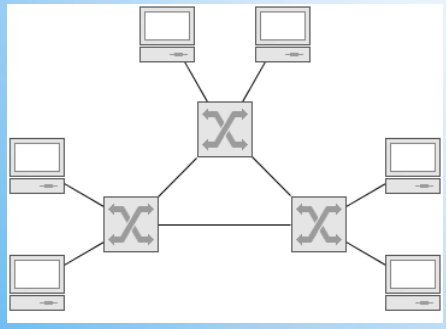

- Koren stabla odgovara formuli, a listovi su ulazne promenljive. Pod stabla
- odgovaraju pod formulama.

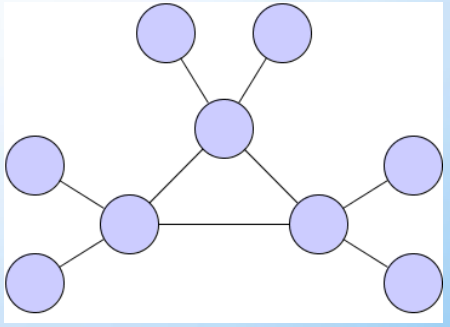
- **Primer:**

  Stabla se mogu iskoristiti da se predstave neki od složenih algoritama, gde je glavni program podeljen na pod programe, kao međusobno nezavisne celine. Kako svaki od pod programa ima svog samo jednog prethodnika , onda znamo koji su mu podaci i kako radi. Potprogrami su pod stabla. Na osnovu grafa možemo da vidimo odakle je sve pod program pozvan.

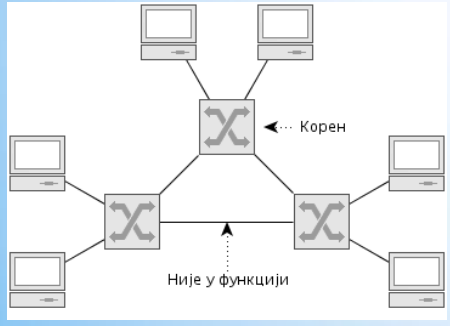


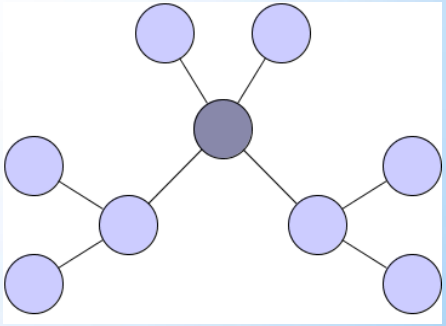
- **Primer:**
- Razapinjuća stabla, odnosno korena stabla, igraju važnu ulogu u lokalnim računarskim mrežama. Problem sa kojim se često srećemo je kako poslati podatak-paket sa jednog računara na više odredišta. Kada se podaci šalju ka više odredišta kroz mrežu ( prva slika ), onda može da zbog petlji dođe do zagušenja rada mreže, a zatim i do njenog potpunog otkazivanja. Razlog tome je beskonačno mnogo paketa koji su namenjeni za isporuku svim članovima mreže. Druga slika prestavlja graf ove mreže.





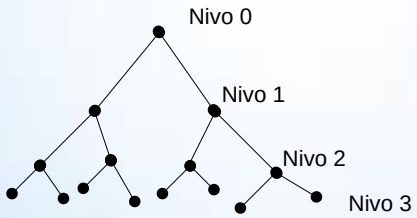
- Da bi se problem rešio koristi se teorija grafova kojom se zadati graf mreže transformiše u razganato stablo. Eliminacijom grana stabla koja u mrežama predstavljaju redundantne veze ( konture u grafu) dobija se razgranato stablo. U takvoj mreži ne postoje zatvorene petlje i ne može da dođe do zagušenja u saobraćaju. Do svakog računara u mreži postoji jedinstvena putanja.



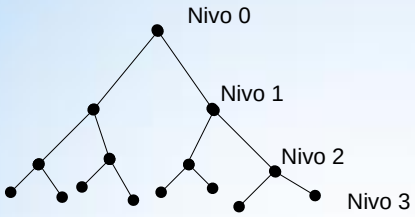



# BINARNA STABLA

- Ako je najveći izlazni stepen, bilo kog čvora stabla, jednak *m*, tada se to stablo naziva *m*- ***arnim stablom*.**
- U posebnom slučaju, ako je *m=2*, dobijamo ***binarno stablo.***
- U binarnom stablu svaki otac ima najviše 2 sina i svako dete se posmatra kao ***levo*** ili ***desno*** dete.
- Ako su u binarnom stablu ***završni čvorovi svi istog nivoa***, binarno stablo se naziva ***potpuno.***




- ***Binarna stabla*** predstavljaju jedan od važnijih pojmova računarskih nauka.
- ***Binarno stablo*** je u informatici struktura namenjena čuvanju podataka. ***Čvor stabla*** je jedna memorijska ćelija stabla.




- Na nivou *n* postoji tačno $2^n$ čvorova
- T: Ako potpuno binarno stablo ima pored nivoa 0 još *k* nivoa, tada je broj čvorova *v* u stablu jednak

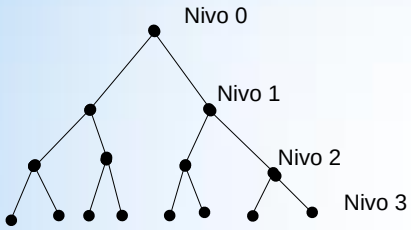
$$v = 1 + 2 + 2^2 + \ldots + 2^k = 2^{k+1} - 1$$

- T: Broj završnih čvorova ( listova ) je $l = 2^k = \frac{v+1}{2}$

- T: Visina *k* stabla je $k = \log_2 (v+1) - 1$

- **Primer:**

Graf na slici ima 3 nivoa, znači ima




$$v = 2^{3+1} - 1 = 15$$

$$l = 2^3 = \frac{15+1}{2} = 8$$

$$h = \log_2 (15+1) - 1 = 3$$

- ***Binarno stablo*** je u informatici struktura namenjena čuvanju podataka.
- ***Čvor stabla*** je jedna memorijska ćelija.

- Stabla generalno, a binarna stabla posebno, koriste se za definisanje, ***uređivanje i pretragu podataka***.
- Pomoću njih se svaki podatak može lako pronaći, utvrditi šta nedostaje, dodati ili izbaciti nepotreban podatak.
- Da bi se to moglo uraditi mora da postoji neko utvrđeno ***pravilo***, koje se zove ***ključ***, a može da bude ***numerički ili alfabetski***.
- Dogovorno se uzima da su leva deca su manja ili jednaka od roditelja, a čvor sa najmanjom vrednošću je naj levlji. Desna deca su veća ili jednaka od roditelja, a čvor sa najvećom vrednošću je naj dešniji.

# FORMIRANJE STABLA

- Jedan od algoritama da se od zadatih podataka formira binarno stablo glasio bi:

- ALGORITAM:

1. Definiše se ključ –pravilo
2. pretraga počinje od korena stabla
3. ukoliko je element  veći od oca, idi na desno dete i ponovi ispitivanje
4. ukoliko je element  manji od oca, idi na  levo dete i ponovi ispitivanje.

**Primer.**

- Formirati sledeće binarno stablo pretrage.

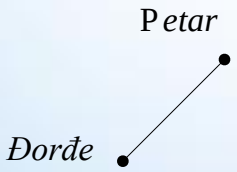
  Poređajmo sledeća imena po abecedi uzimajući redom imena:

  Petar, Đorđe, Sima, Helena, Stoja, Rista, Dunja, Martin, Vasa i Laza.

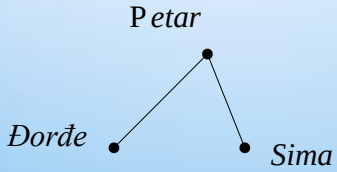
  Uzeti Petra za koren stabla.

  *Napomena*: abeceda- a,b,c,č,ć,d,dž,đ,e,f,g,h,i,j,k,l,lj,m,n,nj,o,p,r,s,š,t,u,v,z,ž

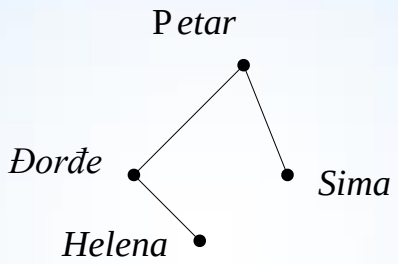
- Poćićemo od imena Petra koje ćemo postaviti za koren stabla. Pošto se ime Đorđe nalazi u nizu posle njega (Đ<P), a abecedno je ispred imena Petar, on će postati njegovo levo dete.



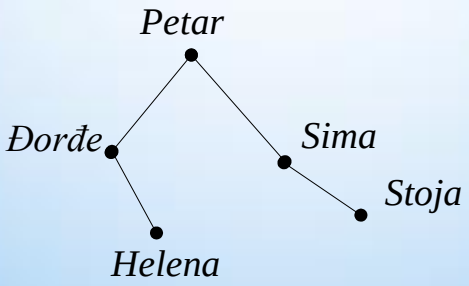

- Sledeće ime je Sima, koje se nalazi iza imena Petar (S>P), pa će zato postati njegovo desno dete.



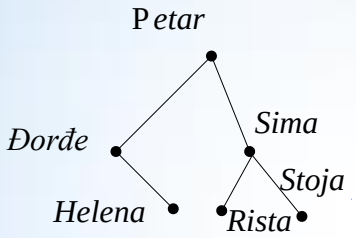
- Sledeće ime Helena. Abecedno je ispred imena Petar (H<P) i spuštamo se do levog deteta, Đorđe, a kako je abecedno iza imena Đorđe (H>Đ), to je njegovo desno dete.



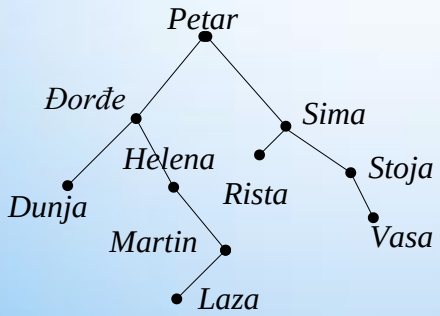

- Ako bi ovako nastavili, sledeće ime je Stoja, ona je Petrovo desno dete (P<S), a iza Sime, pa je Simino desno dete



- Sledeće ime Rista. Abecedno je iza imena Petar (R>P) i spuštamo se do desnog deteta Sime a kao je R abecedno ispred S (R<S), Rista postaje Simino levo dete.




- Ako bi ovako nastavili do kraja dobili bismo stablo

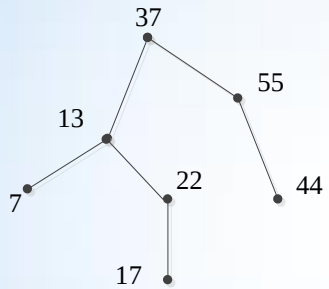


# TRAŽENJE I UBACIVANJE ELEMENTA U STABLO

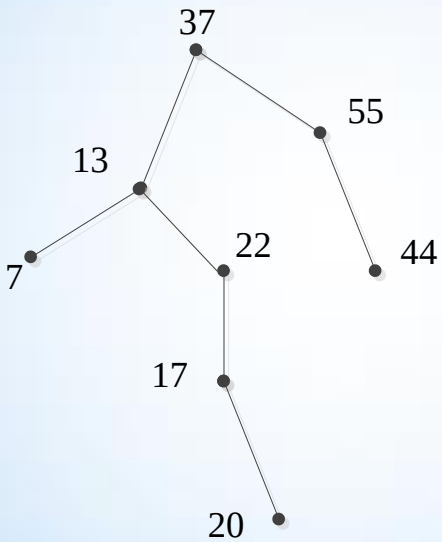
- Pretraga i ubacivanje elementa u binarno stablo definisana je narednim algoritmom. Algoritam nalazi traženi element ili ga ubacuje u stablo ako ga ne nađe.
-
- ALGORITAM:
    1. Početi od korena stabla
    2. uporedi traženi element sa korenom stabla
    3. ukoliko je element manji od korena , idi na levo dete
    4. ukoliko je element veći od korena , idi na desno dete
    5. ponavljati korake 2 i 3 do trenutka
- a) našli smo element uspešno
- b) nismo našli element, dodajemo čvor i pridružujemo mu element

- **Primer.**

Dat je graf. Proveri da li se element 20 nalazi u grafu i ako nije ubaci ga.




1. Uporedi element 20 sa korenom. Kako je 20<37 pređi na levo dete korena, a to je 13
2. Uporedi element 20 sa elementom 13. Kako je 20>13 pređi na njegovo desno dete , a to je 22
3. Uporedi element 20 sa elementom 22. Kako je 20<22 pređi na njegovo desno dete , a to je 17
4. Uporedi element 20 sa elementom 17. Kako je 20>17, a 17 nema desno dete, unesi 20 kao desno dete od 17.


37
55
13
22
44
7
17
20

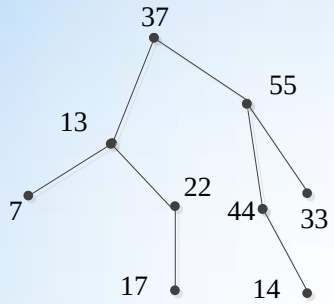
# BRISANJE ELEMENTA IZ STABLA

- ALGORITAM:

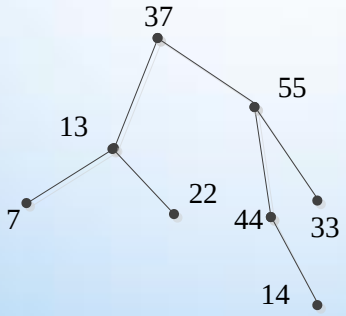
1. Ako čvor v nema dece ukloni ga
2. ako čvor v ima jedno dete, ukloni čvor i zameni ga detetom
3. ako čvor ima dvoje dece , prvo idi na desno dete, a zatim levo dete.

   Redom uzimaj levo dete svakog narednog čvora dok ne naiđeš do čvora koji nema levo dete. Polazni čvor v zameni tim čvorom i neka njegovo desno dete postane levo dete njegovog roditelja .

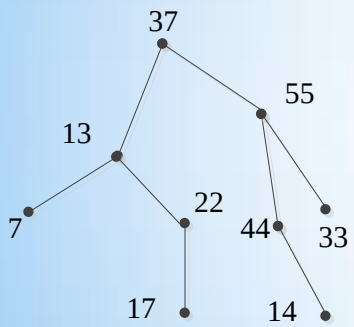
- **Primer.**
  Dat je graf

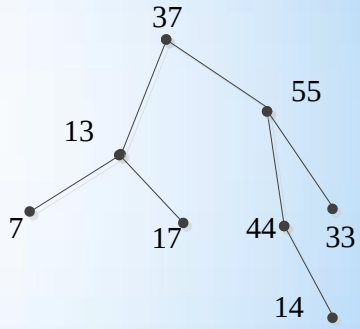



Ako se ukloni element 17 iz grafa, dobijamo sledeći graf –( pravilo 1: Ako čvor v nema dece ukloni ga)



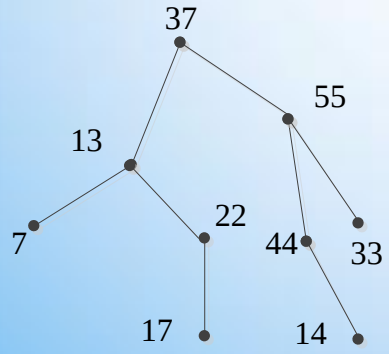
- Ako se ukloni  element 22 iz grafa, dobijamo sledeći graf- ( pravilo 2- ako čvor v ima jedno dete, ukloni čvor i zameni  ga detetom )

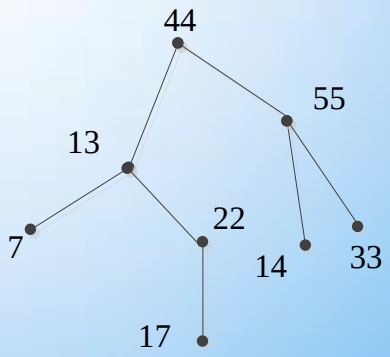


(Pravilo 3- ako čvor ima dvoje dece , prvo idi na desno dete, a zatim levo dete. Redom uzimaj levo dete svakog narednog čvora dok ne naiđeš do čvora koji nema levo dete. Polazni čvor v zameni tim čvorom i neka njegovo desno dete postane levo dete njegovog roditelja) .

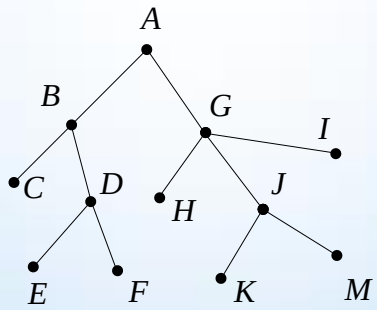
- Da bi se uklonio element 37 iz grafa koji ima 2 deteta, prvo idemo na njegovo desno
- dete 55, a zatim na levo dete 44. Pošto čvor 44 nema levo dete , on postaje novi čvor,
- čvor 14 će postati levo dete čvora 55.



# OBILASCI BINARNIH STABALA

- Standardni načini obilaska čvorova binarnih stabala su:
- KLD, LKD i LDK, gde L predstavlja levo pod stablo, D je desno pod stablo, K je koren i označava kojim redosledom obavljamo obilazak.
- Ako je zadato stablo



1. ***KLD obilazak*** (engl. preorder) bi bio obilazak kod koga se prvo obilazi koren zatim levo podstablo i tek onda desno.

A B C D E F G H J K M I

2. ***LKD obilazak*** (engl. inorder) bi bio obilazak kod koga se prvo obilazi levo pod stablo, zatim koren  i tek onda desno.

C B E D F A K J M H G I

3. ***LDK obilazak*** (engl. postorder) bi bio obilazak kod koga se prvo obilazi levo pod stablo, zatim desno i koren i na kraju.

C E F D B K M J H I G A