ALGORITMI I STRUKTURE PODATAKA

STUDIJSKI PROGRAMI:

SOFTVERSKO INŽENJERSTVO, RAČUNARSKA TEHNIKA, INFORMATIKA I MATEMATIKA

NASTAVNIK: DOC. DR ULFETA MAROVAC, UMAROVAC@NP.AC.RS

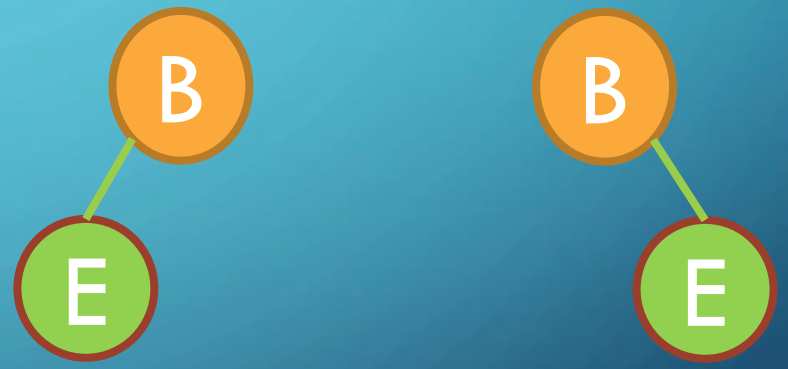
BINARNA STABLA

BINARNA STABLA

- Jedna od najvažnijih i najčešće korišćenih vrsta stabla je **binarno stablo**
- *Binarno stablo* se rekurzivno definiše kao konačan skup čvorova koji je:
  - ili prazan
  - ili se sastoji od korena sa dva posebna podstabla, levim i desnim, koja su, takođe, binarna stabla.
- To znači da svaki čvor ima:
  - ili oba sina,
  - ili samo levog sina,
  - ili samo desnog sina ili nema sinova.

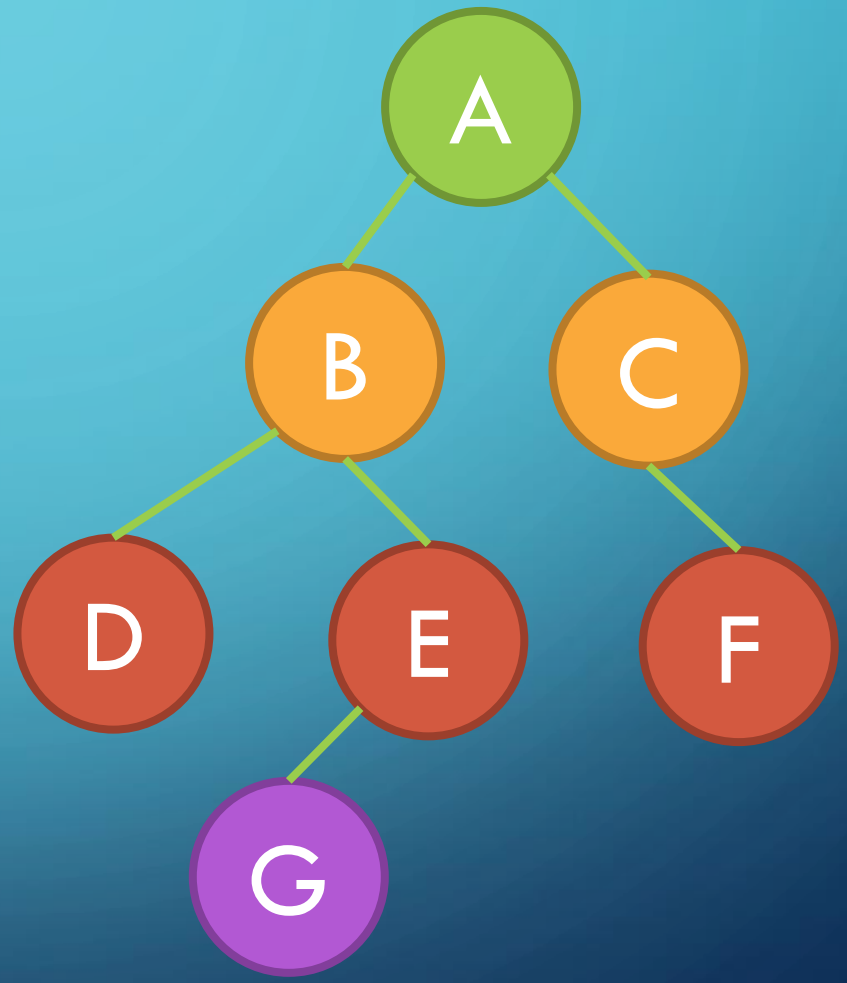
POREDAK BINARNOG STABLA

- Poredak sinova je bitan, tako da dva stabla sa slike nisu ista, iako "liče".
- Kao uređena stabla ona bi bila jednaka, ali kao binarna nisu, jer je čvor B u prvom stablu levi sin čvora A, a u drugom stablu njegov desni sin.



BINARNA STABLA

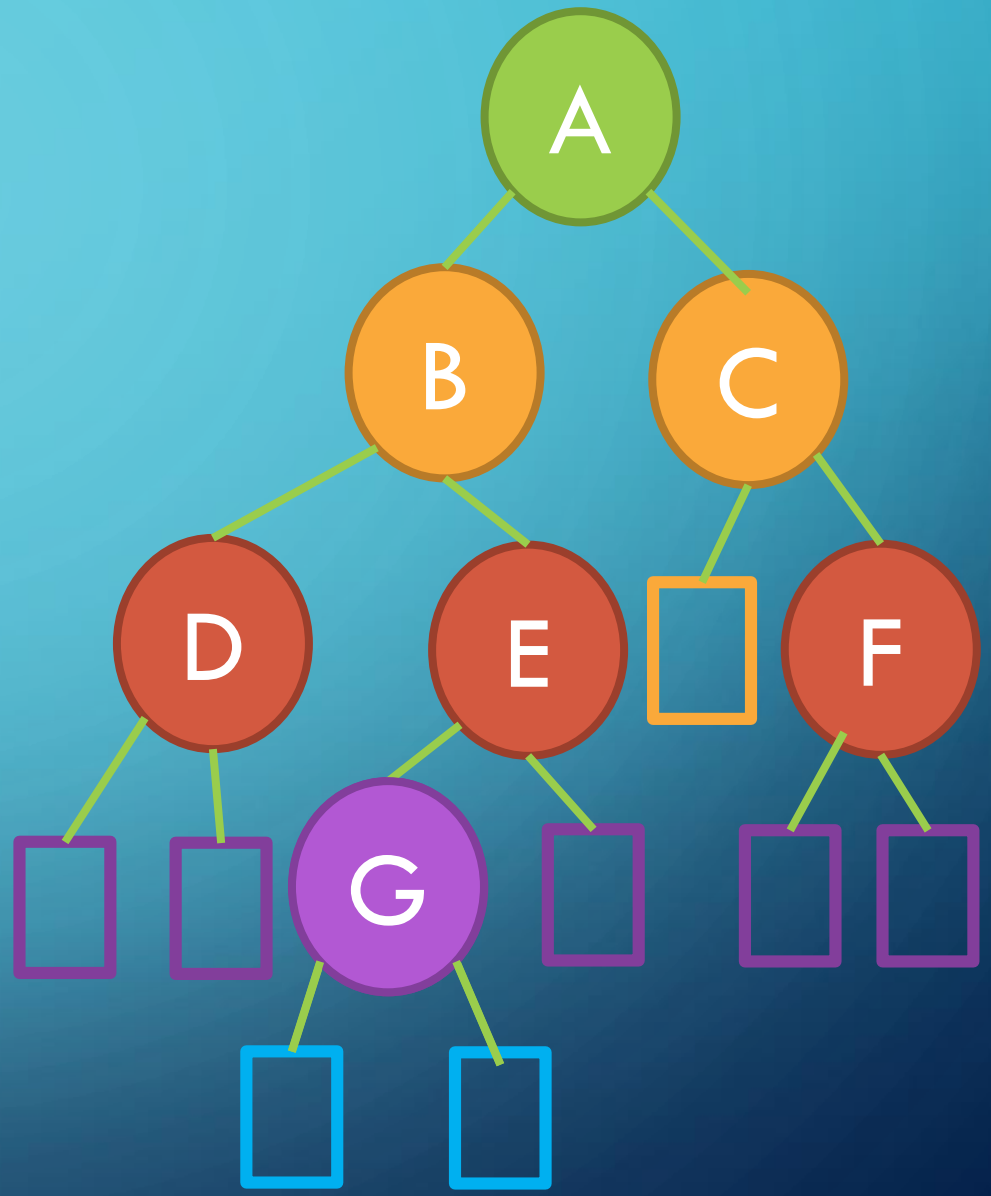
- Prema tome, binarno stablo nije izvedena varijanta uređenog stabla sa stepenom m = 2, jer kod takvog stabla kad neki čvor ima samo jedno podstablo nije bitno da li je ono levo ili desno.
- Očigledno, binarno stablo je po svojoj definiciji poziciono stablo
- Stablo sa slike je, takode, primer binamog stabla.



BINARNA STABLA

- Ukoliko se binarno stablo sa n čvorova proširi posebnim, eksternim čvorovima, kao na slici, može da se uspostavi lineama relacija između inteme i eksteme dužine puta, kao

PE = PI + *2n.*



DOKAZ

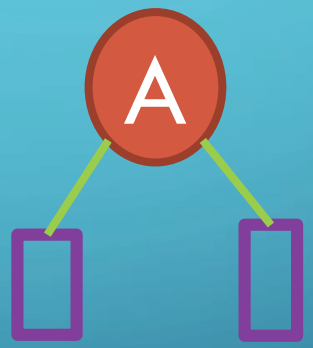
- Ovo svojstvo može da se dokaže indukcijom.
- Za n = 1 stablo ima samo jedan interni čvor (koren) i dva eksterna čvora, pa je:

PI= $\sum in_i = 0 * 1 = 0$

PE= $\sum ie_i$=1*2=2

PE= $\mathrm{PI} + 2 * 1$

i formula očigledno važi. …



DOKAZ

Neka važi n, dokazimo za n+1

Tada jedan eksterni čvor postaje interni, dužina puta do njega je k (cvor H) . Pa je $PI(n + 1) = PI(n) + k$ (dodat jedan interni cvor na nivou k) I umesto jednog eksternog cvora na nivou k dobijemo dva eksterna na nivou k+1 pa je :

$$PE(n + 1) = PE(n) - k + 2(k + 1).$$

Otuda dokaz za $PE = PI + 2n$

1: $PI = 0, PE = 2$

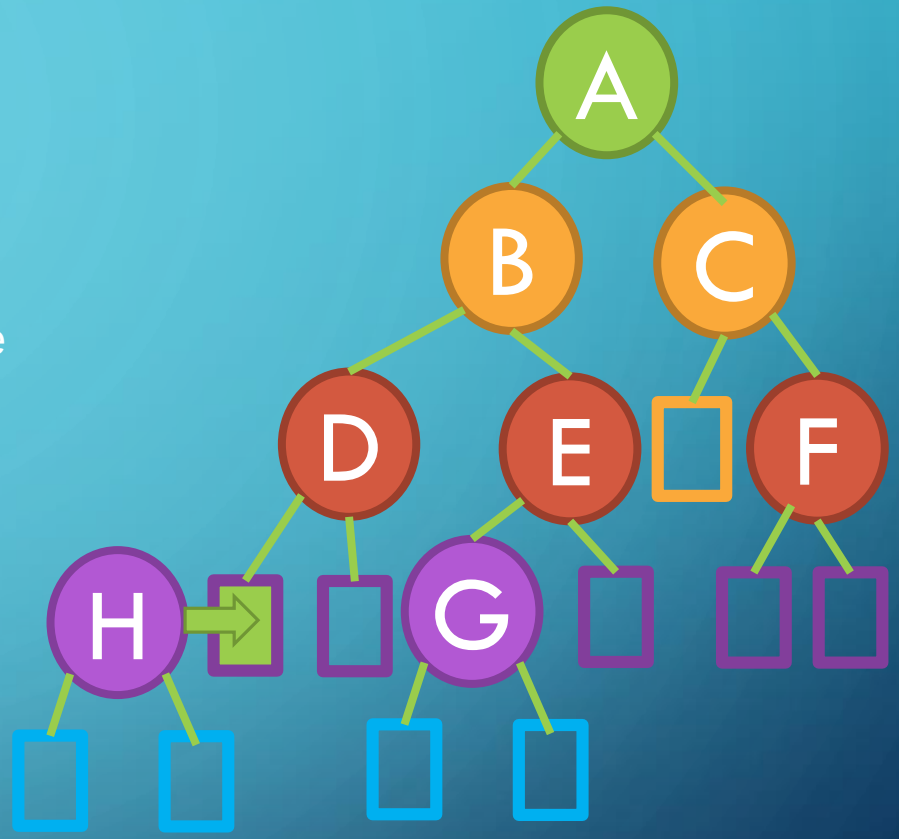
n : $PE(n) = PI(n) + 2n$

$n+1$: $PE(n + 1) = PE(n) - k + 2(k + 1)$

$= PE(n) + k + 2$

$= PI(n) + 2n + k + 2$

$= PI(n + 1) + 2(n + 1)$

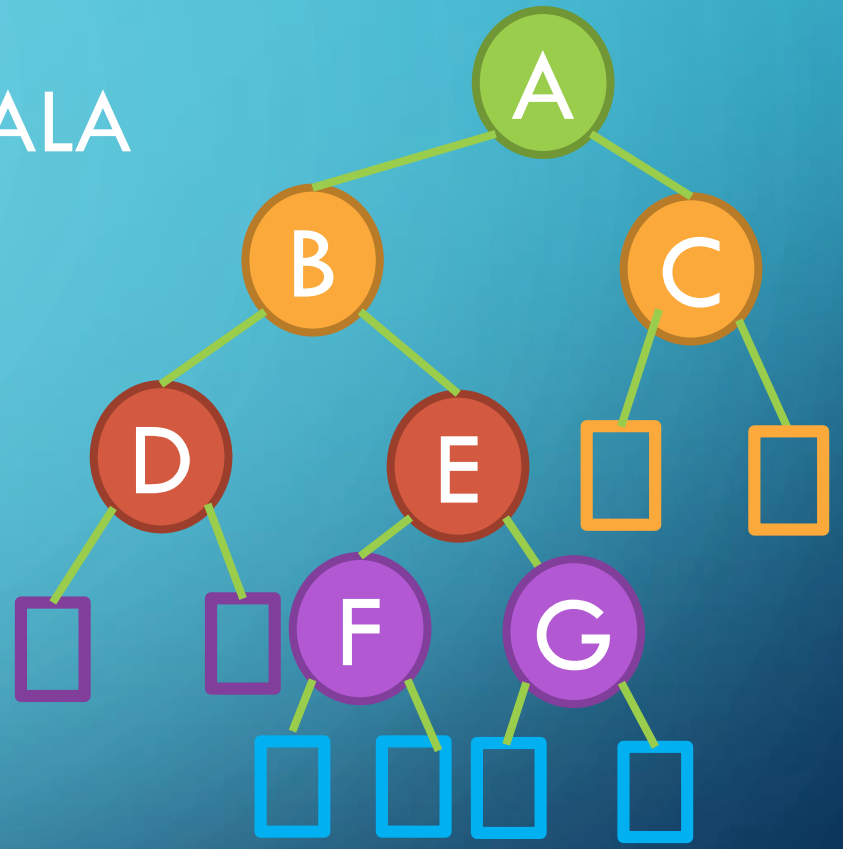


OSOBINE I VRSTE BINARNIH STABALA

- Neka je broj čvorova $n=n_0+n_1+n_2$, gde broj u indeksu označava broj dece koje čvor ima.
- Broj ulaznih grana n-1(samo u koren nema ulazne grane),
- a broj izlaznih je $1* n_1 + 2*n_2$
- $2n_2+n_1=n_0+n_1+n_2-1$
- $n_2=n_0-1$

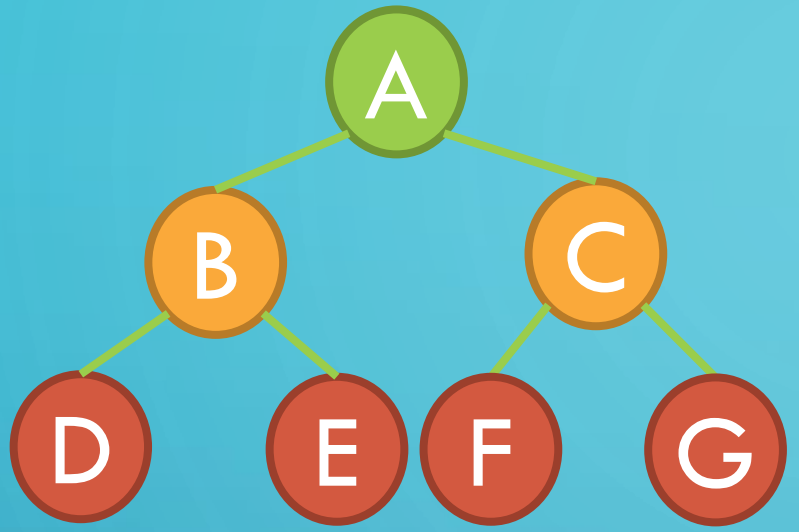
OSOBINE I VRSTE BINARNIH STABALA

- Stablo u kojem svi interni čvorovi imaju oba sina naziva se ***punim*** stablom (n_1=0)
- Kod njega je broj listova za jedan veći od broja čvorova grananja.
- Kod ovog stabla ukupan broj čvorova je obavezno neparan dok je broj eksternih cvorova uvek paran.

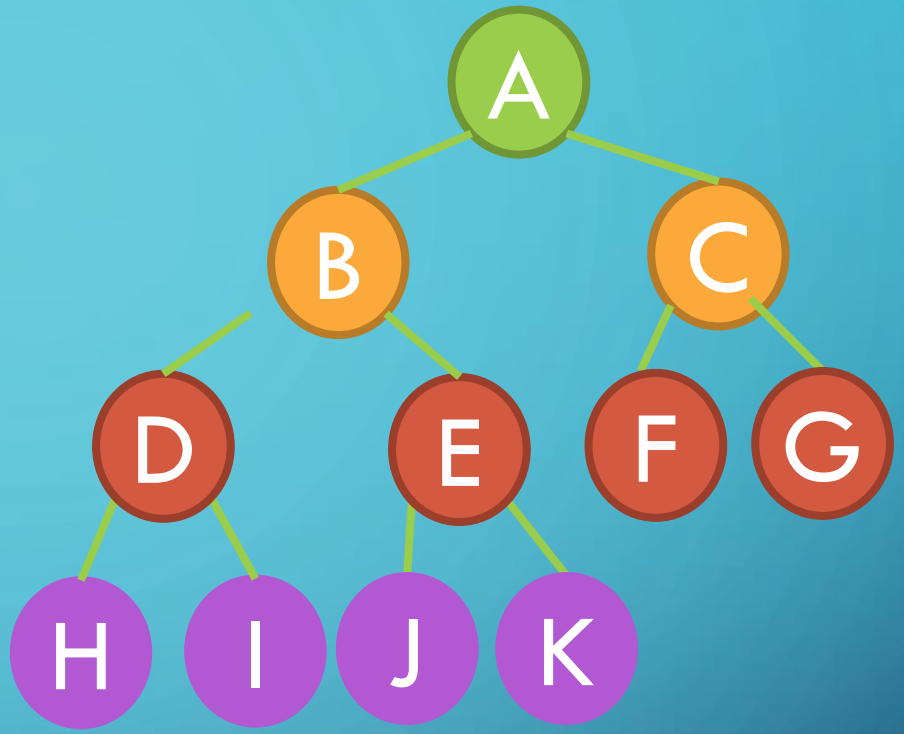


OSOBINE I VRSTE BINARNIH STABALA

- Binarno stablo koje je puno, a svi listovi su na istom nivou je ***kompletno*** binarno stablo. Slika a)
- *Skoro* kompletno stablo se od kompletnog stabla razlikuje po tome što poslednji nivo *h* nije potpuno popunjen, već je sukcesivno popunjen pocevši od krajnjeg levog lista sa brojem čvorova manjim od 2^h
- Dok je kompletno stablo obavezno i puno, skoro kompletno može, ali ne mora, da bude puno.
- Tako skoro kompletno stablo sa slike b) nije puno i ima paran broj cvorova.



A) KOMPLETNO STABLO



B)SKORO KOMPLETNO STABLO

PUNO I KOMPLETNO STABLO

- Broj čvorova sa dva potomka je za jedan manje od broja listova.

$n_0 = n_2 + 1$

- U punom stablu broj čvorova je obavezno neparan
- $n = n_2 + n_0 = 2n_2 + 1$
- Broj čvorova komletnog stable na i-tom nivou je 2^i pa je ukupan broj čvorova je $\sum_{i=0}^{h} 2^i = 2^{h+1} - 1$ gde je h visina.
- Kompletno stablo je binarno stablo minimalne visine sa n čvorova i ta visina iznosi

$h_{min} = \lceil \log(n + 1) \rceil - 1$

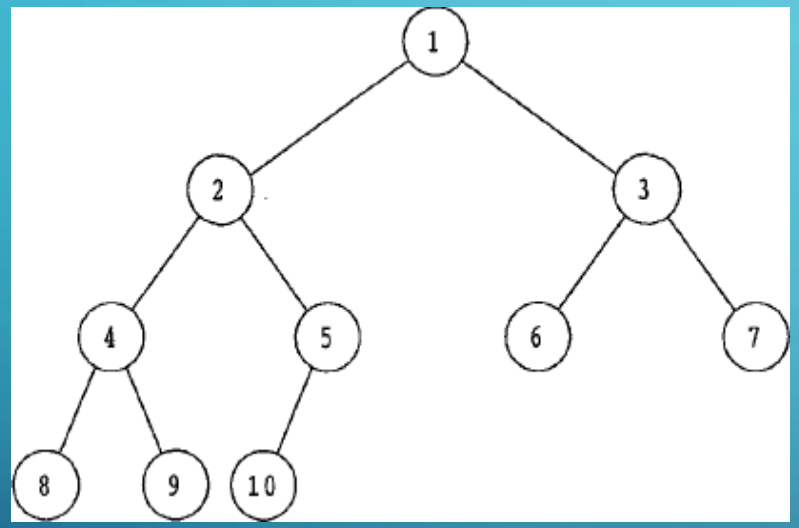
MEMORIJSKA REPREZENTACIJA BINARNIH STABALA

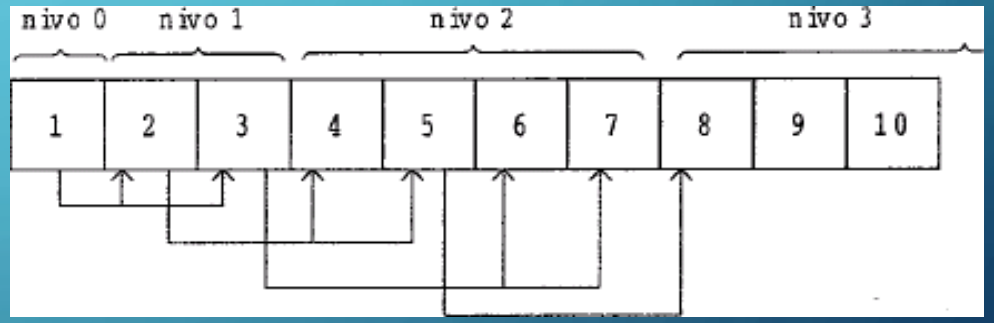
- Pored ulančane predstave, za kompletna i skoro kompletna binarna stabla veoma pogodna je i sekvencijalna reprezentacija.
- Tada se stablo može predstaviti u vidu vektora, bez ikakvih pokazivača, jer su relacije otac-sin inherentno određene njihovim pozicijama u vektoru.
- Zato je ova reprezentacija veoma prostorno efikasna.
- Neka se čvorovi numerišu od 1 do n počevši od korena, a zatim na svakom narednom nivou sukcesivno sleva udesno
- Onda se u vektoru čvoroviređaju po nivoima i rastućim brojevima čvorova

VEKTORSKA REPREZENTACIJA STABLA

- U kompletnom binarnom stablu sa n čvorova predstavljenom vektorom , za svaki čvor sa indeksom i važi :
- otac čvora i je na poziciji [i/2], ako je i>1 (za i=1 čvor je koren i nema oca)
- levi sin čvora i je na poziciji 2i, ako je 2i<=n (za 2i>n čvor nema levog sina)
- desni sin čvora i je na poziciji 2i+1, ako je 2i+1<=n (za 2i+1>n čvor nema desnog sina)

VEKTORSKA REPREZENTACIJA STABLA





MINIMIZACIJA INTERNE DUŽINE PUTA

- Najgori slučaj
  - degenerisano stablo - jedan čvor po nivou
  - $PI = n(n-1)/2 \sim O(n^2)$
- Prosečan slučaj $O(n\sqrt{n})$
- Najbolji slučaj
  - čvorovi što bliže korenu
  - $PI_{min} = 0+1+1+2+2+2+2+\ldots = \sum [\log n\ ]$
  - $\Rightarrow O(n \log n)$
  - svi listovi na dva susedna nivoa

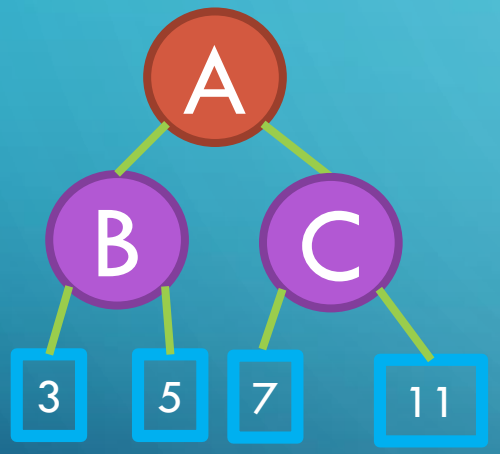
TEŽINSKA EKSTERNA DUŽINA PUTA

- Eksternim čvorovima pridružene “težine” *w*

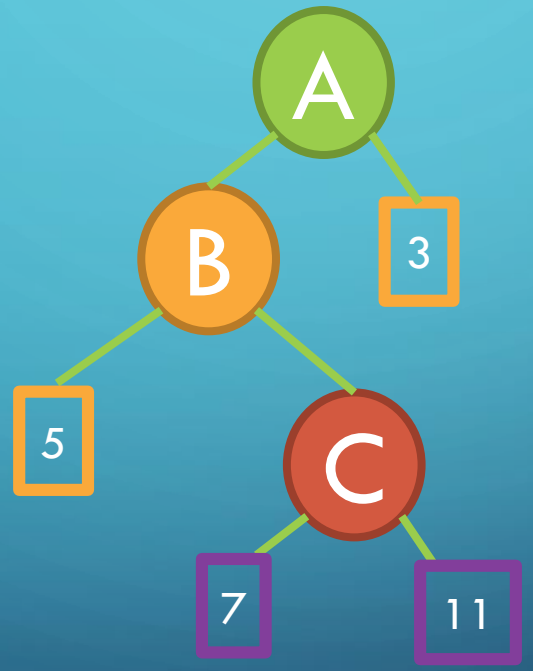
- Težinska eksterna dužina puta
  - $PWE = \sum w_i l_i$
  - w_i *je težina čvora a* l_i *dužina puta od korena do tog eksternog čvora*

TEŽINSKA EKSTERNA DUŽINA PUTA

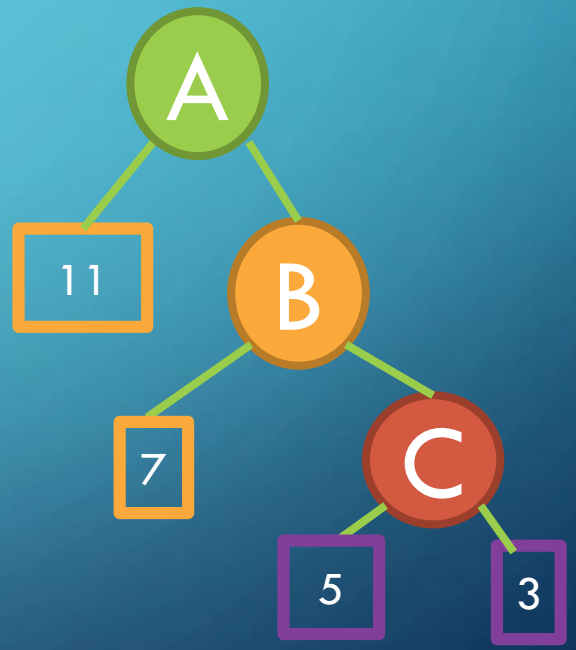
a)



b)



c)



IZRAČUNAJTE EKSTENRNE DUŽINE PUTA

- A) $PWE = \sum w_i l_i = 3*2+5*2+7*2+11*2=52$
- B) $PWE = \sum w_i l_i = 3*1+5*2+7*3+11*3= 67$
- C) $PWE = \sum w_i l_i = 3*3+5*3+7*2+11*1=49$
- Dakle najmanja eksterna dužina puta je primeru c)
- Kako doći do stabla koje sadrži odgovarajuće eksterne čvorove sa težinama w1,..wn a da je minimalna eksterna dužina puta
- Ovim problemom se bavi Huffman'ov algoritam

HUFFMAN-OV ALGORITAM

- Ulaz u algoritam predstavlja broj eksternih čvorova i njihove pridružene težine date u vektoru Q[1:e] , a algoritam izgrađuje stablo sa minimalnom težinskom eksternom dužinom puta i vraća adresu njegovog korena
- Polje w u eksternim čvorovima ovog stabla sadrži njihovu težinu, a u ostalim čvorovima predstavlja zbir težina svih eksternih čvorova koji su potomci tog čvora

```
HUFFMAN(W, e)
INIT-QUEUE(H, e)
for i = 1 to e do
    z = GETNODE
    w(z) = W[i]
    PQ-INSERT(H, z)
end_for
for i = 1 to e - 1 do
    z = GETNODE
    x = PQ-MIN-DELETE(H)
    y = PQ-MIN-DELETE(H)
    w(z) = w(x)+w(y)
    left(z) = x
    right(z) = y
    PQ-INSERT(H, z)
end_for
return z
```

HUFFMAN-OV ALGORITAM

- Red Q[4]={3,5,7,11}
- Algoritam inicijalizuje prioritetni red **H**,
- Alocira prostor za **e** eksternih čvorova, upisuje njihove težine u polje **w** i upisuje ih u prioritetni red **H** koji je organizovan po rastućim vrednostima polja **w**.
- Tako se polazi od šume od **e** stabala koja se sastoje od po jednog eksternog čvora.

c1	W=11

c2	W=7

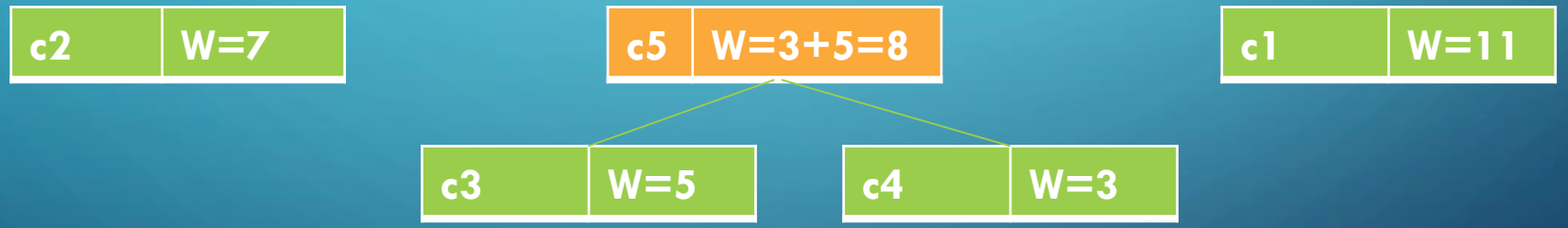
c3	W=5

c4	W=3

- H={c4,c3,c2,c1}
- U svakoj iteraciji algoritma se formira 1 čvor grananja čija je težina jednaka zbiru težina svih eksternih čvorova koji su njegovi potomci, a kako je broj čvorova grananja za 1 manji od broja eksternih, potrebno je e-1 iteracija.
- Garantuje se da ima minimalnu težinsku eksternu dužinu puta.

HUFFMAN-OV ALGORITAM

- U prvoj iteraciji iz reda H izvlače dva čvora sa najmanjim težinama x i y, napravi novi čvor **z** sa težinom koja je jednaka njihovom zbiru, pa **z** postaje koren stabla u kojem su **x** i **y** koreni levog i desnog podstabla. Koren novog stabla se, zatim, ubaci u prioritetni red H.

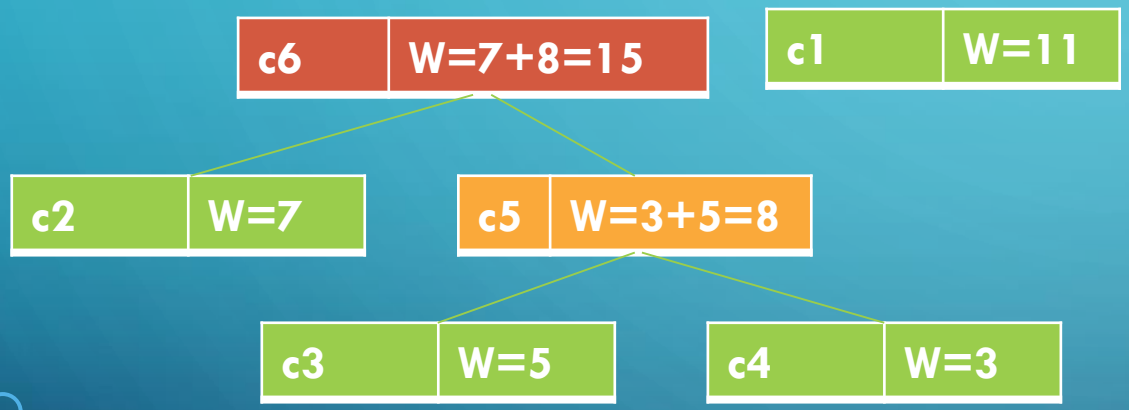


- H={c2,c5,c1}

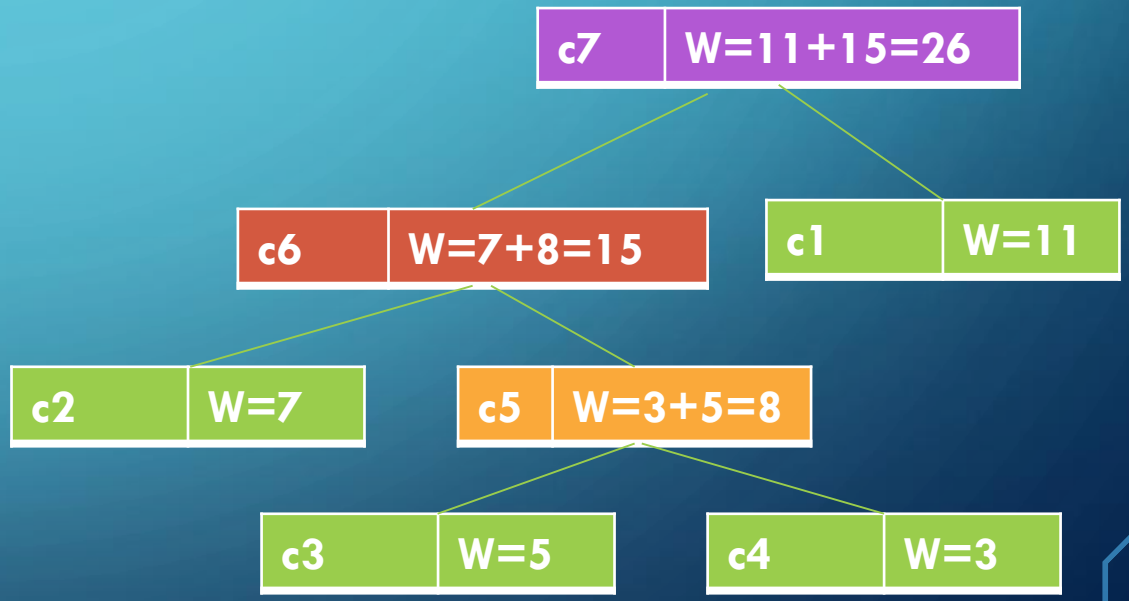
HUFFMAN-OV ALGORITAM

- U svakom narednom koraku iz reda H se biraju dva korena sa najmanjom težinom i od njihovih stabala pravi novo stablo sa korenom čija je težina jednaka zbiru njihovih težina.
- Tako se broj stabala u šumi koju predstavlja red H smanjuje za 1 u svakoj iteraciji, a postupak se završava kad šuma postane stablo. U ovom stablu listovi su polazni eksterni čvorovi, a koren ovog stabla ima težinu koja je jednaka zbiru težina svih eksternih čvorova.
- Za rezultujuće stablo se garantuje da ima minimalnu težinsku eksternu dužinu puta.

H={c6,c1}

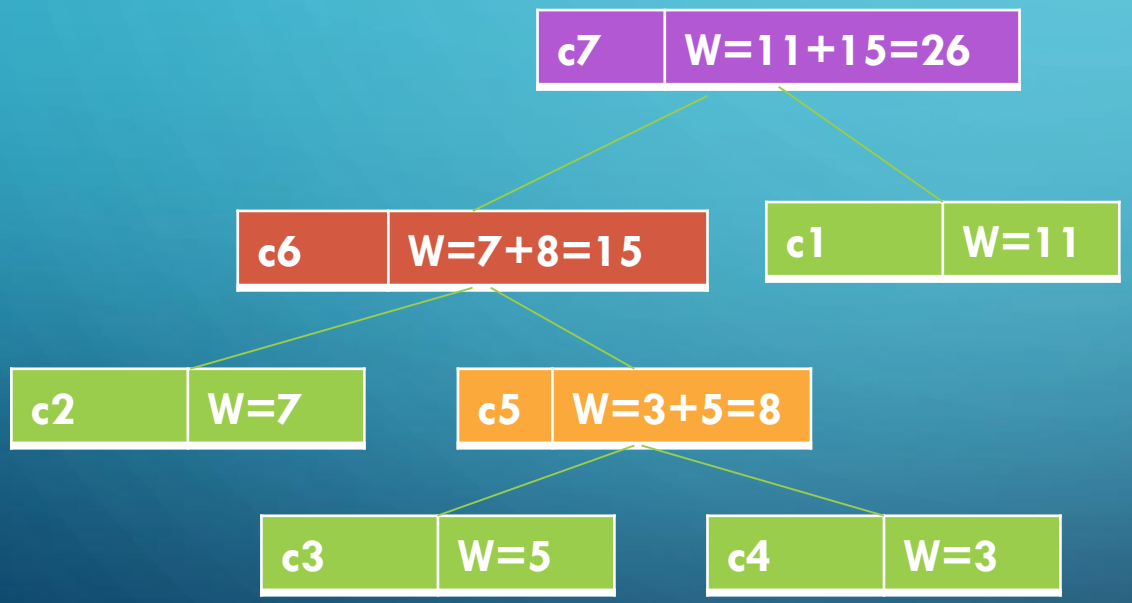

c6
W=7+8=15
c1
W=11
c2
W=7
c5
W=3+5=8
c3
W=5
c4
W=3

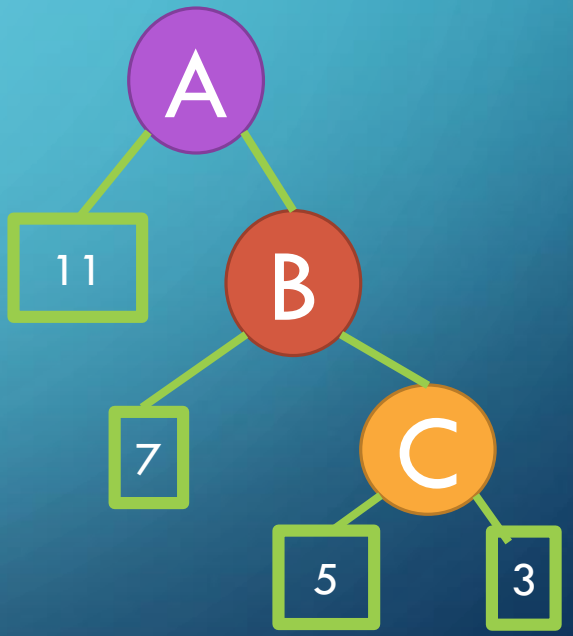
H={c7}


c7
W=11+15=26
c6
W=7+8=15
c1
W=11
c2
W=7
c5
W=3+5=8
c3
W=5
c4
W=3

RESENJE

- Stablo pod c) je stablo sa minimalnim externim tezinskim putem

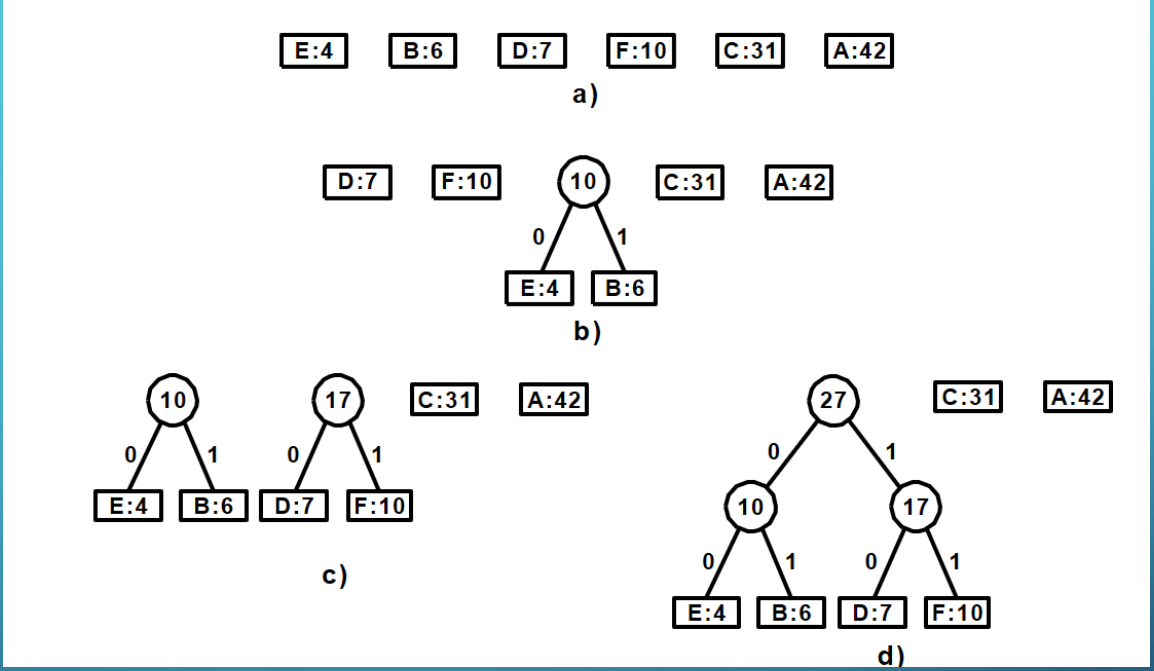




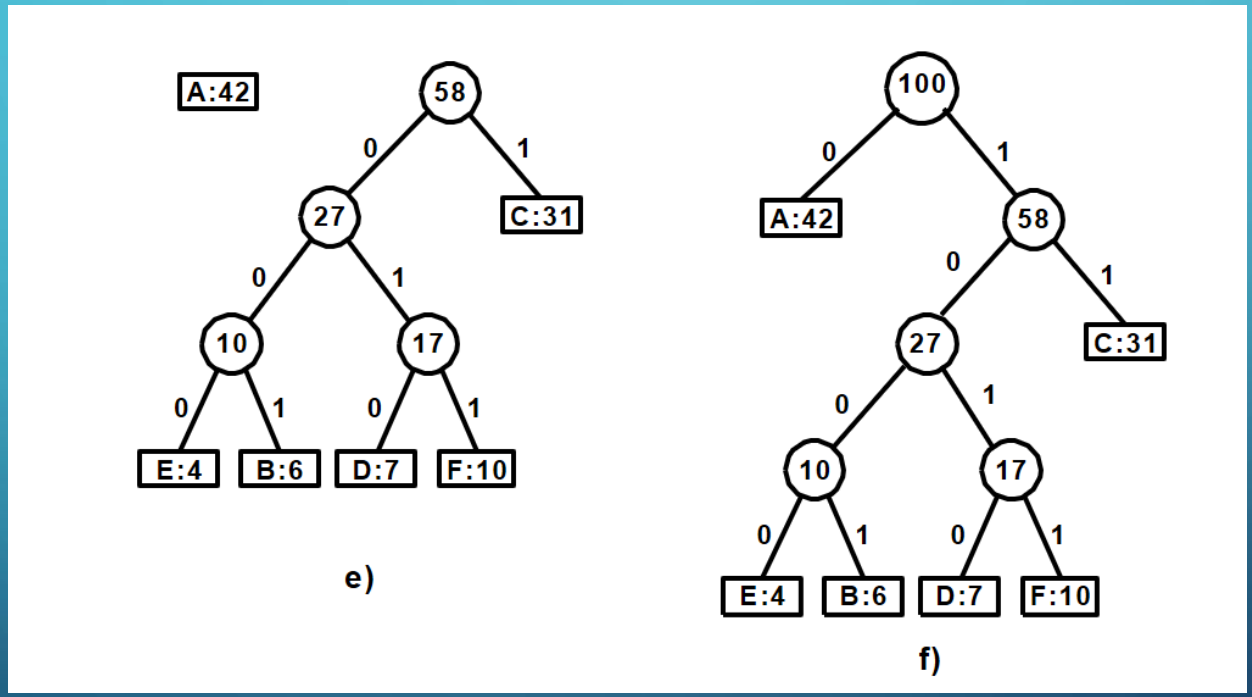
PRIMENA

- Skraćenje dužine kodovane poruke postiže se kodovima različite dužine. Zbog jednoznačnosti dekodiranja kodovi su prefiksni (binarni kod ni jednog simbola ne sme da bude prefix koda bilo kog drugog simbola).

KORACI IZGRADNJE STABLA



KORACI IZGRADNJE STABLA

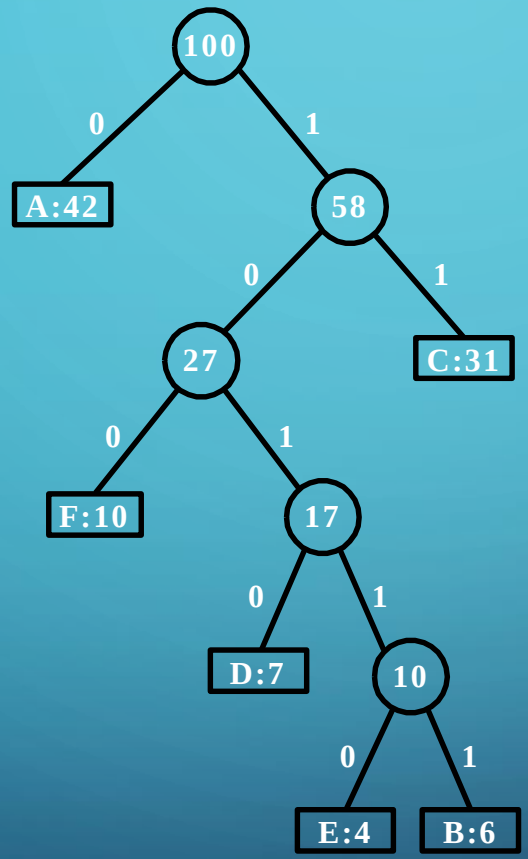


KREIRANJE KODOVA

- Svako slovo kodiramo putanjom 0 I 1 od korena do eksternog cvora u kom se nalazi

Simboli	A	B	C	D	E	F
Verovatnoće	42	6	31	7	4	10
Kodovi	0	1001	11	1010	1000	1011

ALTERNATIVNO REŠENJE SA STABLOM VEĆE VISINE



OPERACIJE SA BINARNIM STABLIMA

- Uobičajene operacije u binarnom *stablu* su obilazak stabla, umetanje i brisanje čvora.
- Obilazak stabla na 3 načina
  - Preorder
  - Inorder
  - Postorder

PREORDER

- Poseti koren
- Obidji levo podstablo na preorder način
- Obidji desno podstablo na preorder način

INORDER

- Obidji levo podstablo na inorder način
- Poseti koren
- Obidji desno podstablo na inorder način

POSTORDER

- Obidji levo podstablo na postorder način
- Obidji desno podstablo na postorder način
- Poseti koren

REKURZIVNE REALIZACIJE OBILAZAKA

```
PREORDER(root)
if (root ≠ nil) then
        P(root)
        PREORDER(left(root))
        PREORDER(right(root))
end_if
```

```
POSTORDER(root)
if (root ≠ nil) then
        POSTORDER(left(root))
        POSTORDER(right(root))
        P(root)
end_if
```

```
INORDER(root)
if (root ≠ nil) then
        INORDER(left(root))
        P(root)
        INORDER(right(root))
end_if
```

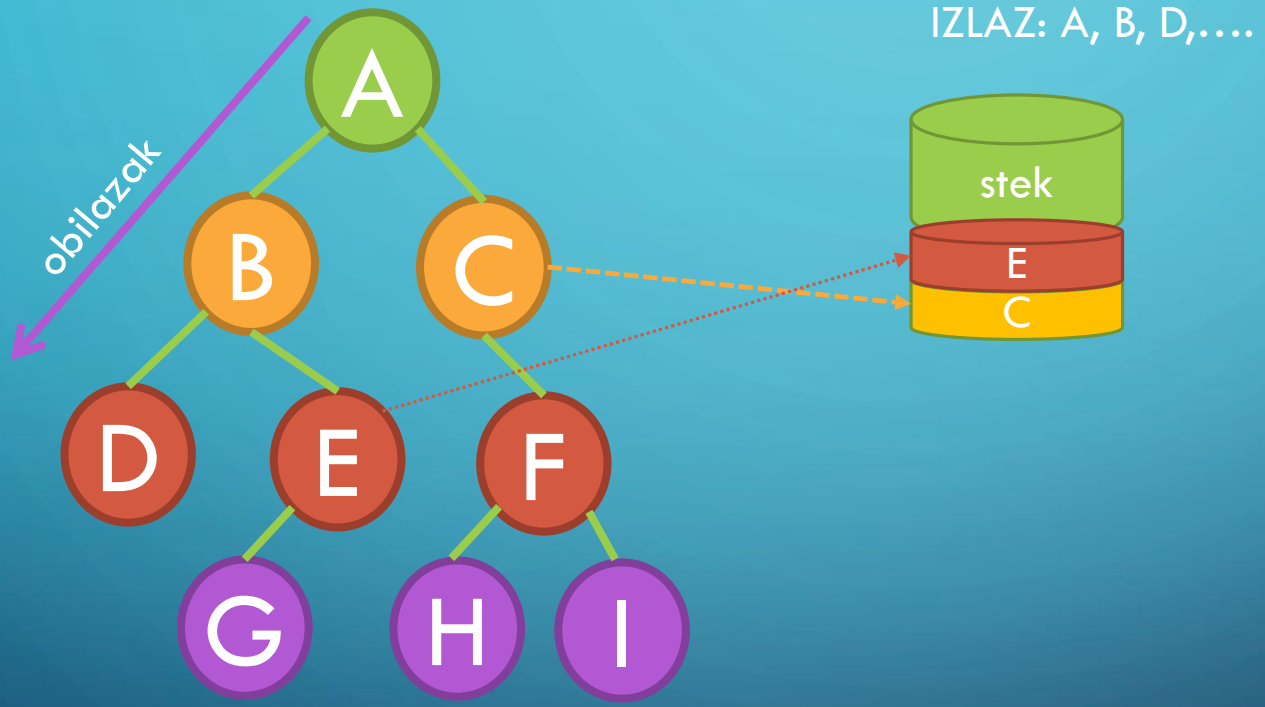
ITERATIVNE REALIZACIJE OBILASKA STABLA

- Zbog svoje efikasnosti od velikog značaja su i iterativne realizacije obilaska stabla.
- Pri iterativnoj realizaciji uvodimo korisnički stek
- Stek čuva put kojim smo dosli do odredjnog cvora

PREORDER OBILAZAK ITERATIVNI NAČIN

- Na stek se spusta koren stabla
- Metodom P se posecije cvor koji se podize sa steka
- Zatim se spusta po nepraznim levim pokazivacima i obilazi cvorove sve dok ne stigne do najnizeg levog podstabla pamteci pritom adrese korena desnih postabala koje treba obici kasnije u obrnutom redosledu
- Kada vise ne moze da se spusta po levoj strani algoritam se vraca na poslednje zapamceno desno podstablo

PRIMER STABLA ZA OBILAZAK



PREORDER-ITERATIVNA REALIZACIJA

```
PREORDER-I(root)
PUSH(S, root)
while (not STACK-EMPTY(S)) do
    next = POP(S)
    while (next ≠ nil) do
        P(next)
        if (right(next) ≠ nil) then
            PUSH(S, right(next))
        end_if
        next = left(next)
    end_while
end_while
```

$\Rightarrow O(n)$

S	*next*	Posećeni čvor	Preorder poredak
A			
	A	A	A
C	B	B	AB
CE	D	D	ABD
CE	nil		
C	E	E	ABDE
C	G	G	ABDEG
C	nil		
	C	C	ABDEGC
F	nil		
	F	F	ABDEGCF
I	H	H	ABDEGCFH
I	nil		
	I	I	ABDEGCFHI
	nil		

POSTORDER-ITERATIVNA REALIZACIJA

```
POSTORDER-I(root)
next = root
while (next ≠ nil) do
      PUSH(S, next)
next =left(next)
end_while
while (not STACK-EMPTY(S)) do
      next = POP(S)
      if (next > 0) then
      PUSH(S, -next)
      next = right(next)
       while (next ≠ nil) do
                  PUSH(S, next)
                  next = left(next)
            end_while

             else
            next = - next
            P(next)
      end_if
end_while
```

~ O(*n*)

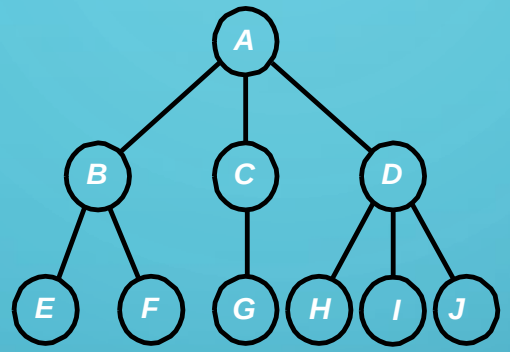
STABLA VIŠEG STEPENA

- Problem u stablima stepena $m > 2$
  - ✓ neefikasno korišćenje prostora u ulančanoj reprezentaciji
  - ✓ neiskorišćeni pokazivači $n(m-1)+1$ iskorišćeni pokazivači $n-1$

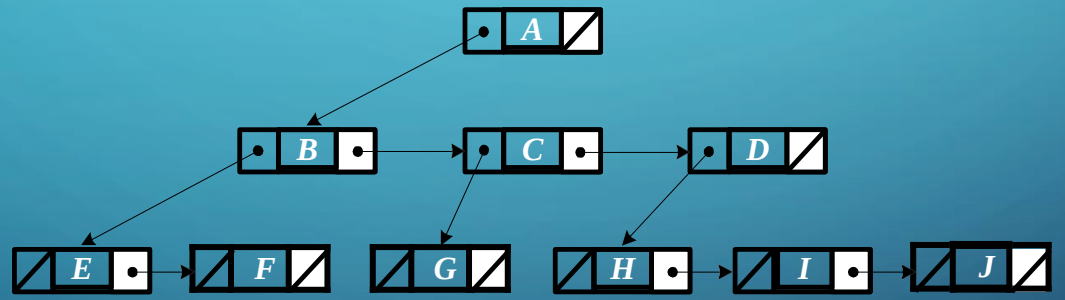
- Rešenje
  - ✓ odgovarajuće binarno stablo iste semantike
  - ✓ binarna relacija “najlevlji sin – desni brat”
  - ✓ svi sinovi istog oca u ulančanoj listi

STABLA VIŠEG STEPENA

a)



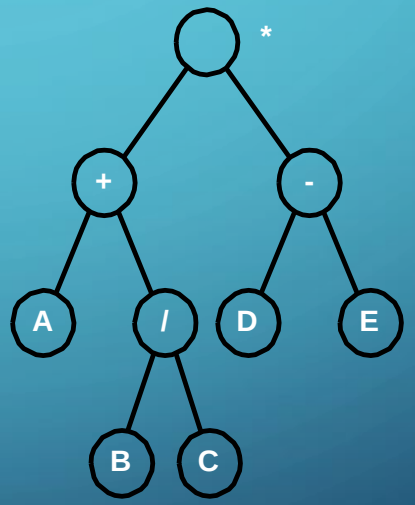
b)



PREDSTAVLJANJE ARITMETIČKIH IZRAZA

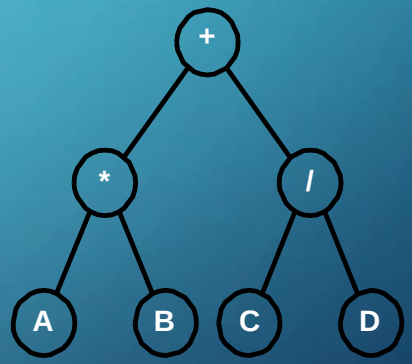
- Binarno stablo predstavlja aritmetički izraz
  - ✓ čvorovi grananja – unarni i binarni operatori
  - ✓ listovi - operandi

(A+B/C)(D-E)



a)

A*B+C/D



b)

PREDSTAVLJANJE ARITMETIČKIH IZRAZA

- Obilazak stabla
  - ✓ preorder daje prefiksni izraz
  - ✓ postorder daje postfiksni izraz
  - ✓ inorder daje infiksni izraz (ako nema zagrada!)

```
CALC-EXP(r)
case (info(r)) of
    op__add:return(CALC-EXP(left(r)) + CALC-
    EXP(right(r))  op_sub:return(CALC-EXP(left(r)) -
    CALC-EXP(right(r))
    . . .
    operand:return(VAL(right(r)))
end_case
```

- Izračunavanje izraza predstavljenog stablom

TEST PITANJA

1. Šta može da sadrži svaki čvor binarnog stabla?
2. Kada kazemo da su dva binarna stabla ekvivalentna
3. Kako se definise interna a kako eksterna duzina puta i koja je veza izmedju njih kod binarnog stabla.
4. Dajte primer punog i skoro kompletnog stabla (n>10) Na ovom primeru prikazite veze izmedju broja cvorova, broj cvorova sa dva potomaka, broja listova i druge poznate osobine.
5. Kolika je minimalana a kolika maksimakna visina binarnog stabla od 10 cvorova

TEST PITANJA

6. Predstavite vektorski stablo iz primera 5. ko je otac cvoru pod rednim brojem 10 a gde su mu sinovi
7. Neka su dati cvorovi sa tezinama 4,7, 9,12,3 kreiraj stablo minimalne eksterne dužine puta
8. Šta je cilj Hafmanovog algoritma? Neka su dati čvorovi 9 10 3 7 1 primeni Hafmanov algoritam za dobijanje kreiranje stabla sa najmanjom eksternom dužinom puta
9. Opišite stanje steka, elementa next, trenutno posećenog čvora i poredak posečenih cvorova iterativnim algoritmom za preorder obilazak proizvoljno izabranog stabla.
10. Za stablo iz prethodnog primera ispisite korak po korak realizaciju rekurzivnog inorder obilasaka.