ФАКУЛТЕТ ЗА ИНФОРМАТИЧКИ НАУКИ
И КОМПЈУТЕРСКО ИНЖЕНЕРСТВО

НЕЛИНЕАРНИ ДИНАМИЧКИ СТРУКТУРИ

АЛГОРИТМИ И ПОДАТОЧНИ СТРУКТУРИ

- предавања -

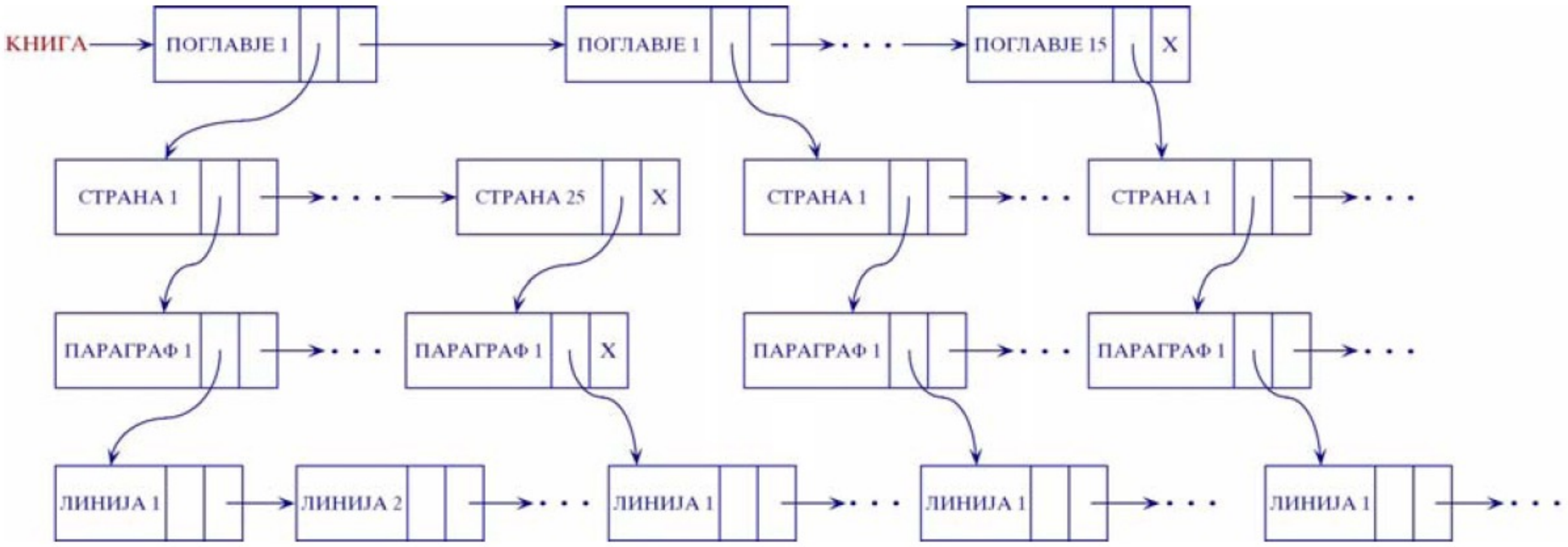
А
П
С

Комплексни листи

- сложени динамички структури
  - листи во кои јазелот покажува кон нова листа

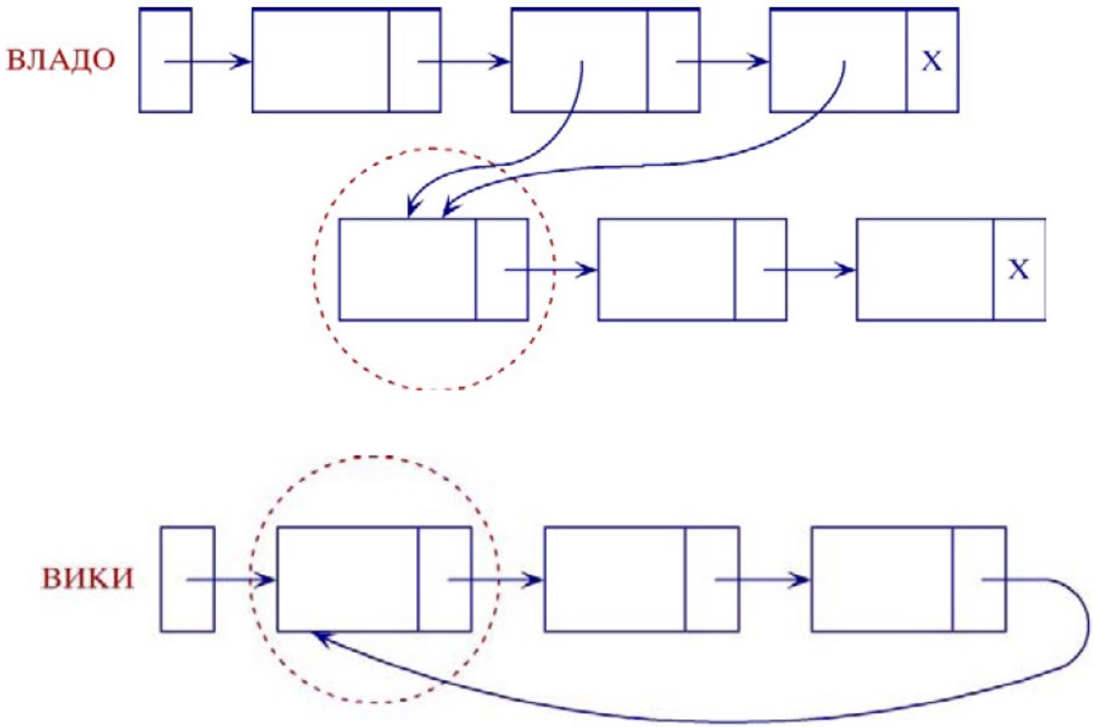
Комплексни листи

- сложени динамички структури
  - листи во кои јазелот покажува кон нова листа
- **пример**: структура на книга (**хиеарахија**)



Комплексни листи

- структури кои дозволуваат повеќе врски да покажуваат (да делат) на еден јазел



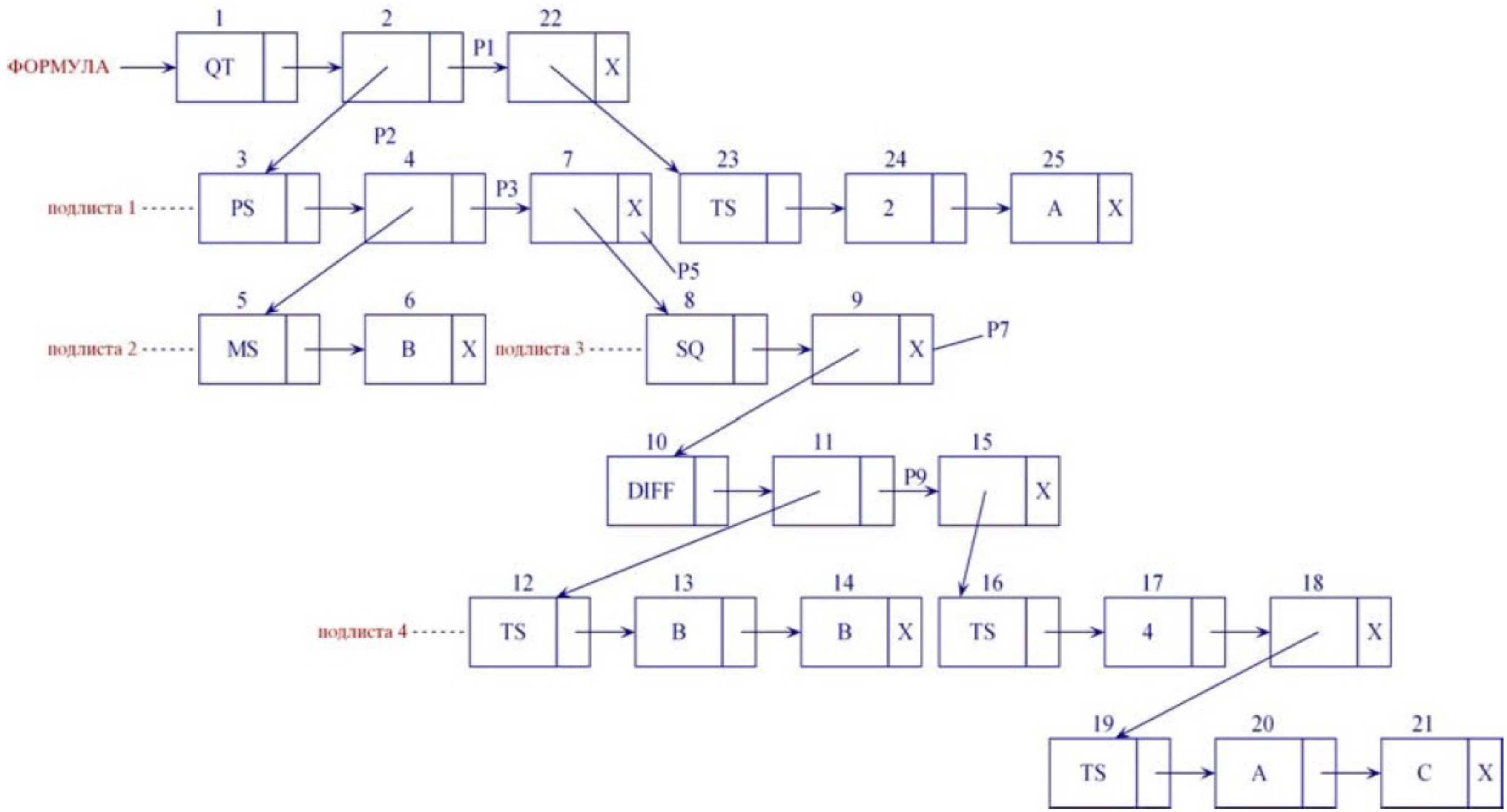
Комплексни листи

- Операции за работа со комплексните листи:
  - внесување на јазел
  - бришење на јазел
  - изминување на листата

Комплексни листи

- Изминувањето на хиерархиските сложени листи може да се опише на следниот начин:
  - Пристапи до првиот јазел (доколку постои)
  - Процесирај го јазелот до кој си пристапил
  - Доколку јазелот е комплексен, измини ја листата (односно листите) кон која тој покажува.
  - Пристапи кон следниот јазел (доколку постои)

Комплексни листи



Дрва

- хиерархиска колекција на елементи
- Дрвото е:
  - колкеција од елементи - јазли
  - еден јазел е специјален - корен
  - релација „е родител на“
  - секој јазел има точно еден родител
  - секој јазел чува податоци од било кој податочен тип

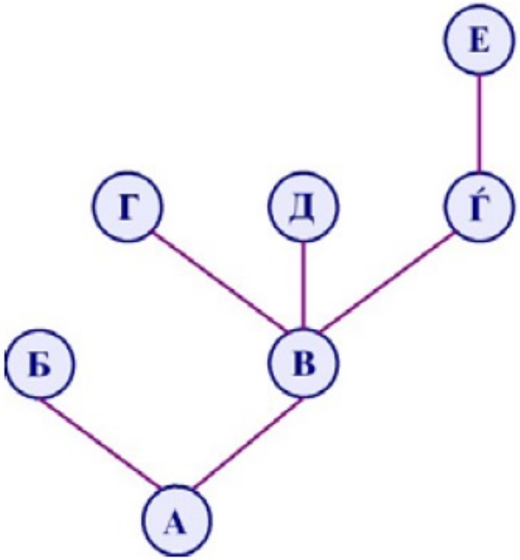
Дрва

- Формална дефиниција на дрво:
  - Јазел сам за себе претставува дрво. Тогаш, јазелот е и корен на дрвото
  - Нека n е јазел и T1, T2, ..., Tk се дрва со корени n1, n2, …, nk соодветно. Тогаш дрво може да се конструира ако јазелот n го направиме корен на дрвото што ги содржи поддрвата T1, T2, ..., Tk. Јазлите n1, n2, …, nk ги нарекуваме деца на јазелот n

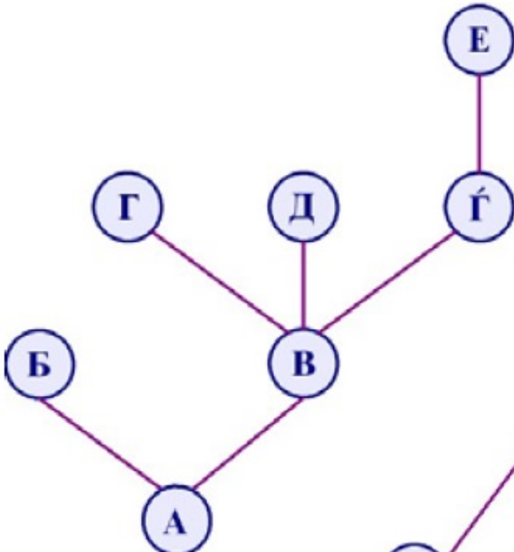
Дрва

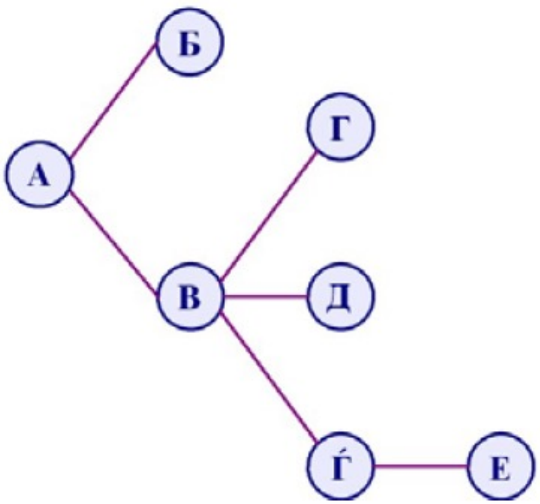
- Рекурзивна дефиниција на дрво:
- Дрво е конечно множество T со еден или повеќе елементи наречени јазли што ги задоволува следниве правила:
  - Постои еден јазел наречен корен на дрвотот
  - Останатите јазли (без коренот) се групирани во $k \geq 0$ дисјунктни множества $T1, T2, ..., Tk$, од кои секое е дрво. Овие дрва се нарекуваат поддрва на дрвото T

Дрва

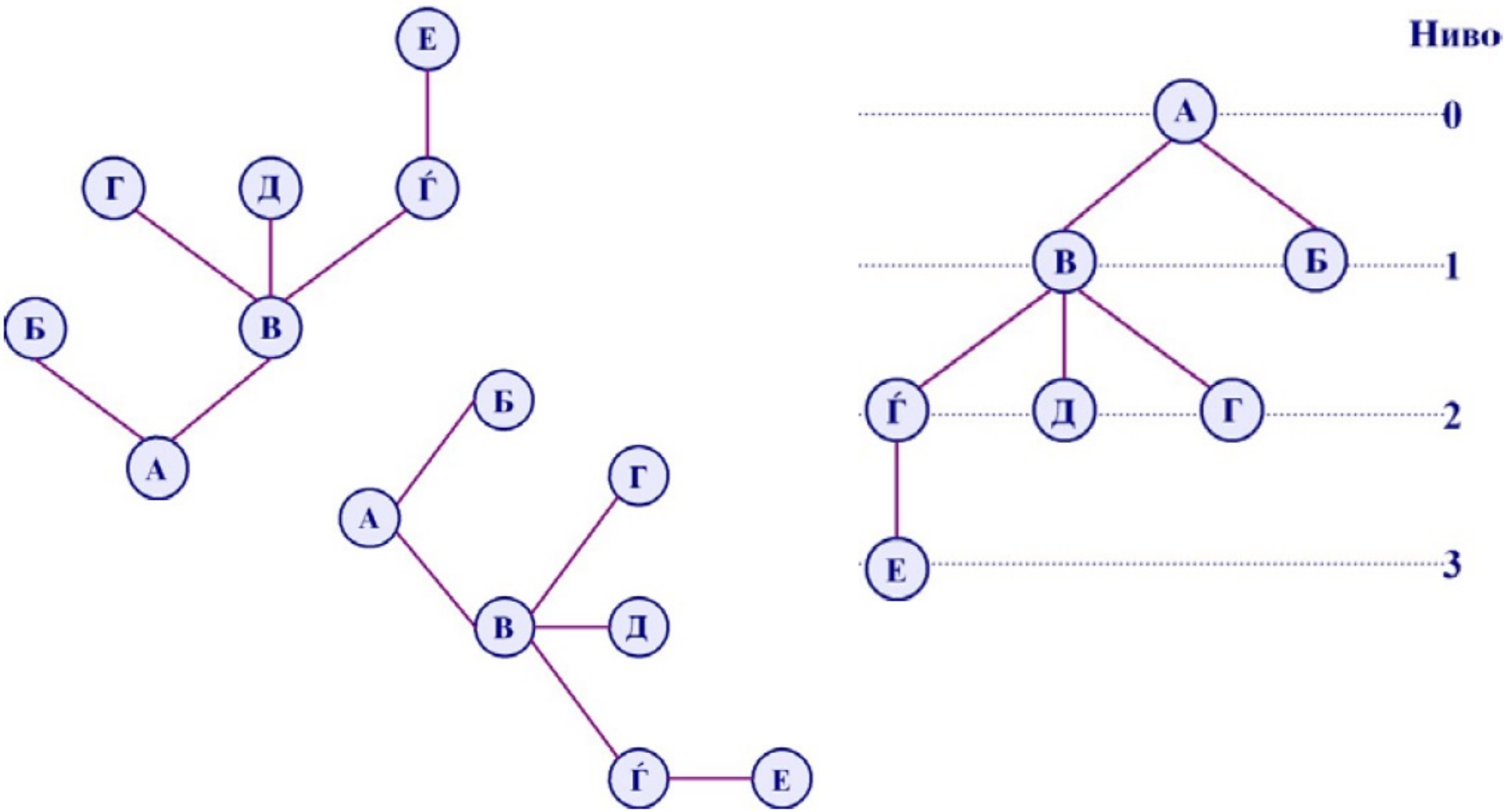


Дрва





Дрва



Дрва

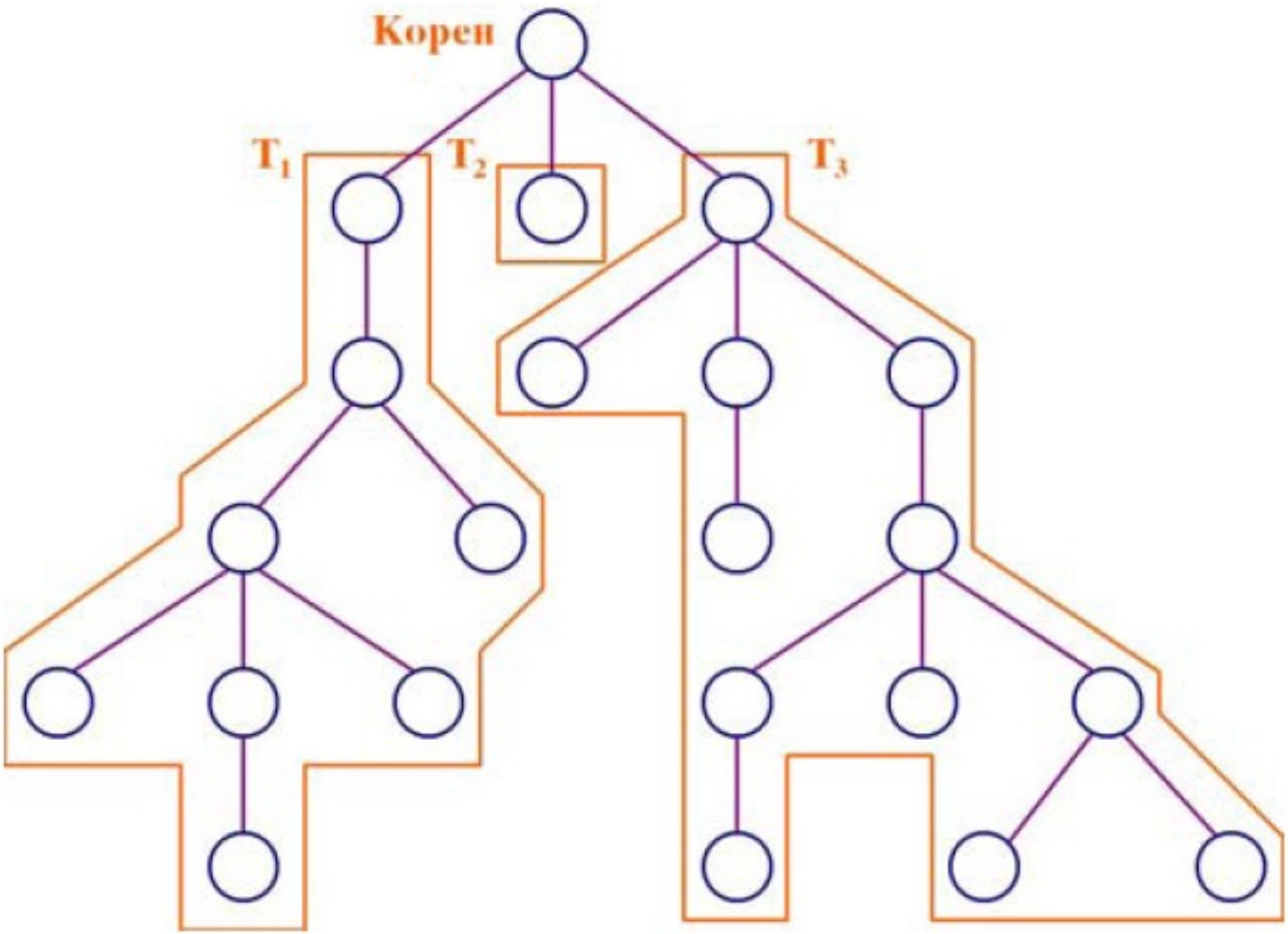
- секој внатрешен јазел во дрвото е корен на некое поддрво
- бројот на поддрва на еден јазел се нарекува степен на јазелот
  - Кога овој број е 0, јазелот се нарекува краен (терминален) јазел или лист
- сите јазли (освен коренот) имаат свој родител

Дрва

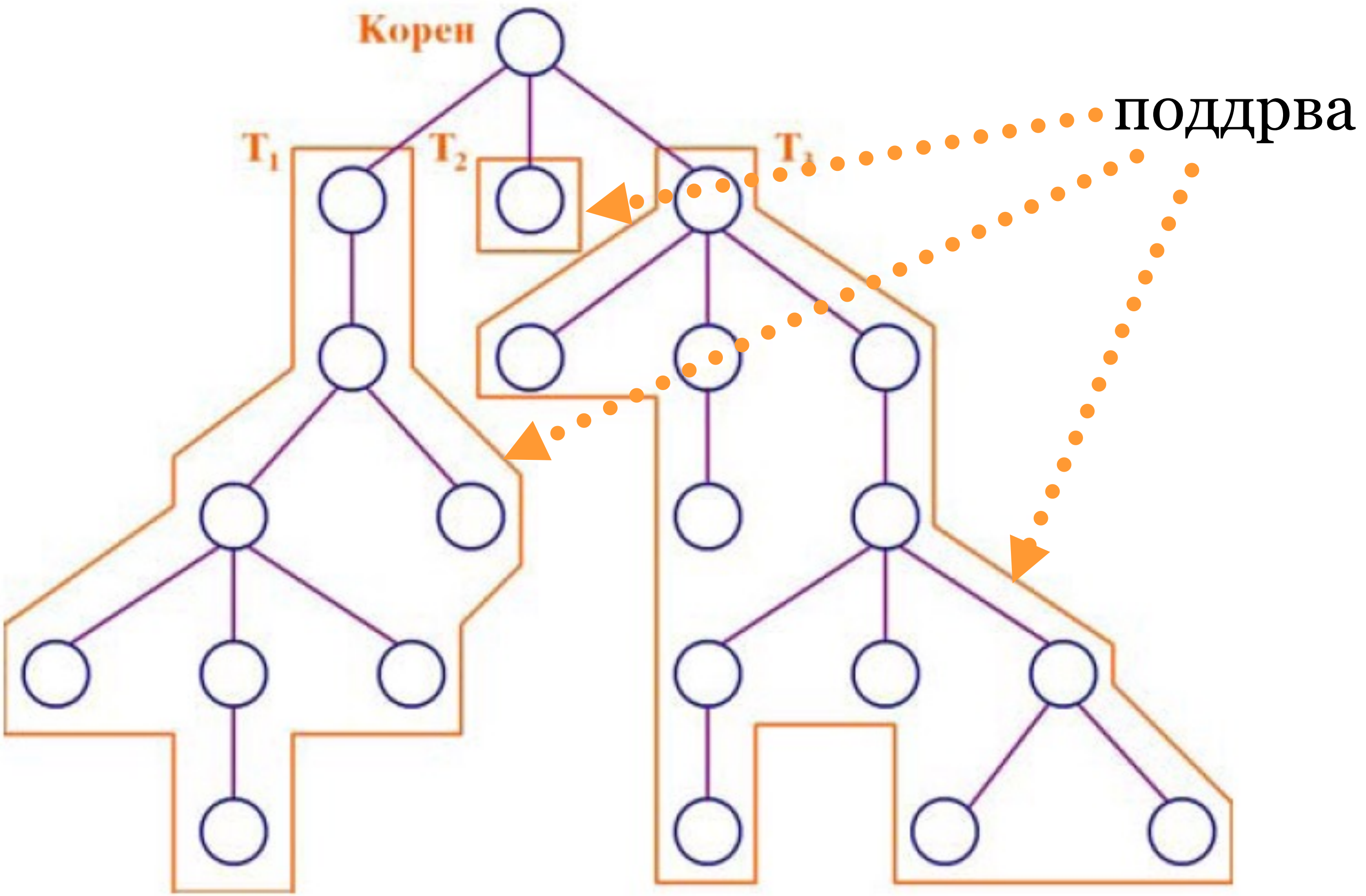
- Слични дрва се дрвата кои имаат иста структура, т.е. чии јазли и врски се соодветни (ако јазелот во едното дрво има две деца, и соодветниот јазел на другото дрво две деца, а и бројот на нивните деца е ист)

- Еквивалентни дрва се дрва кои се слични, но кои носат и иста информација во секој јазел.

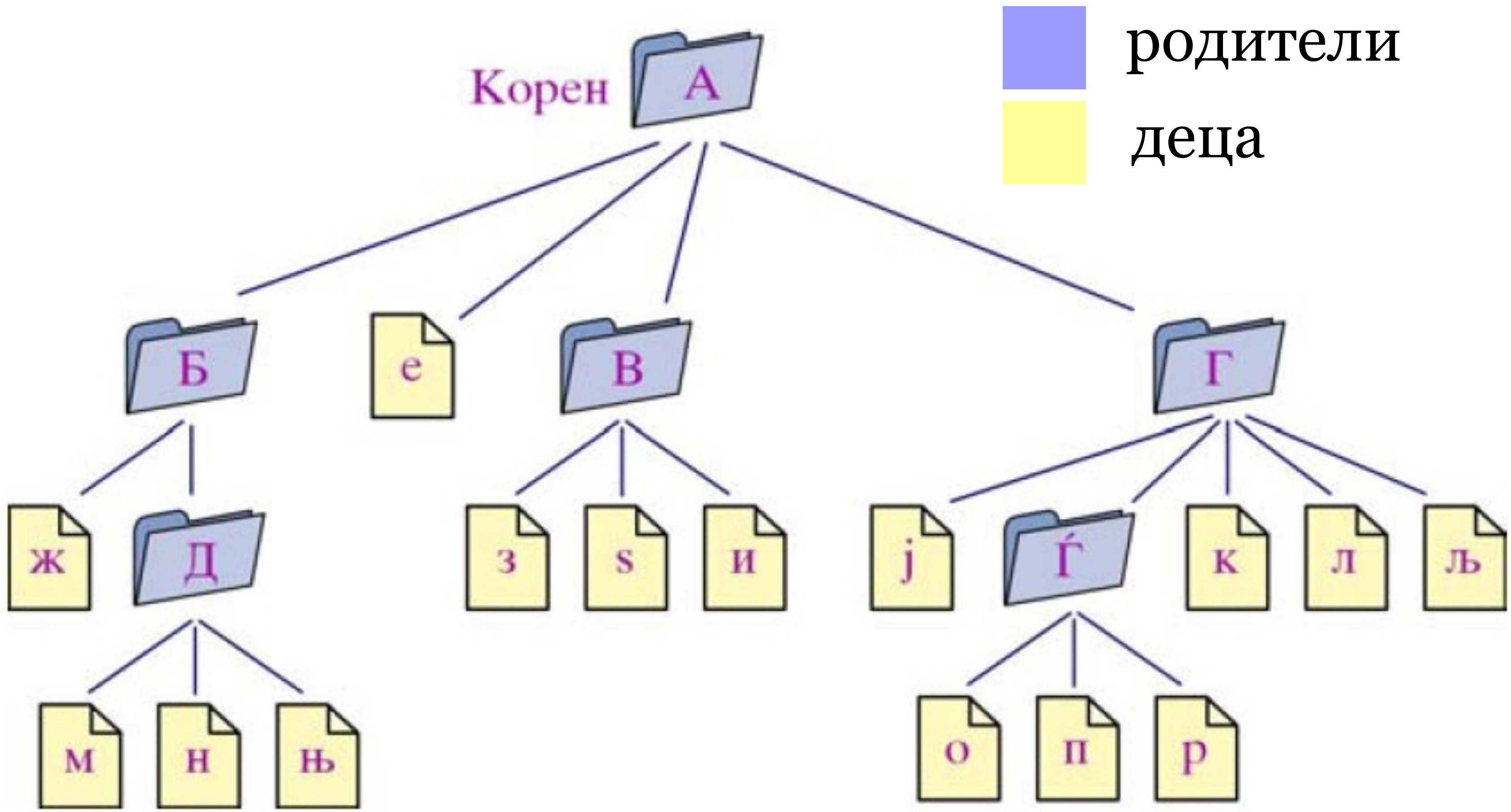
Дрва



Дрва



Дрва



Шума од дрва

- Множество (обично подредено) на различни (дисјунктни) дрва се нарекува **шума**
- Ако од едно дрво го избришеме коренот, се добива шума
- Ако пак во една шума додадеме само еден јазел и го поврземе со корените на дрвата, од шумата добиваме едно дрво

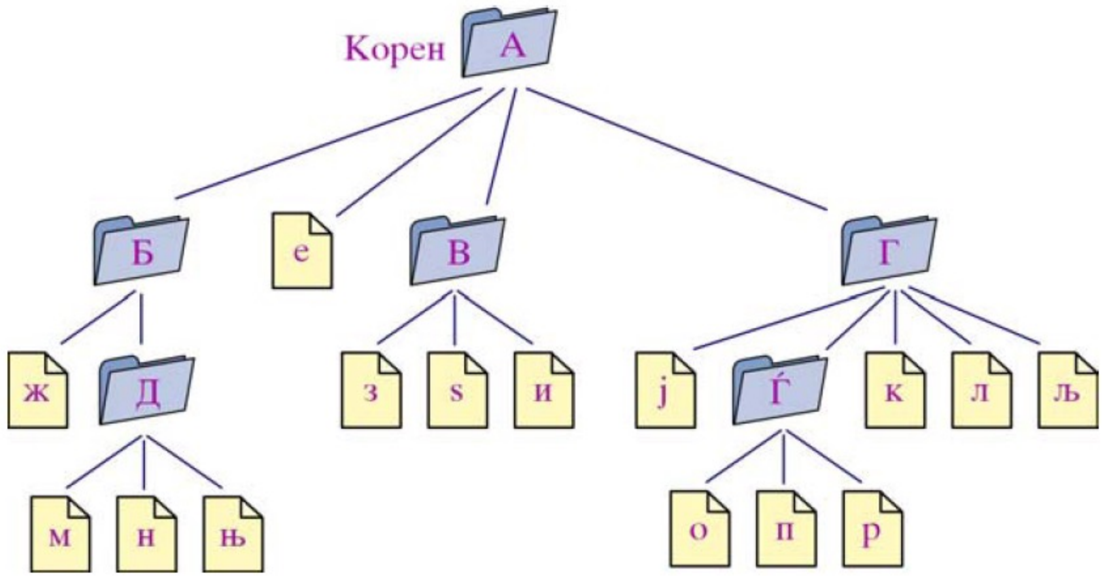
Патека во дрво

- Ако n_1, n_2, …, n_k е низа на јазли во дрво така да n_i е родител на n_{i+1}, $1 \leq i < n$, тогаш низата се нарекува **патека** од јазелот n_1 до n_k
- Должина на патека претставува број на врски меѓу два јазла, односно е за еден помала од бројот на јазли во патеката
- **Предок** и **наследник** на јазел

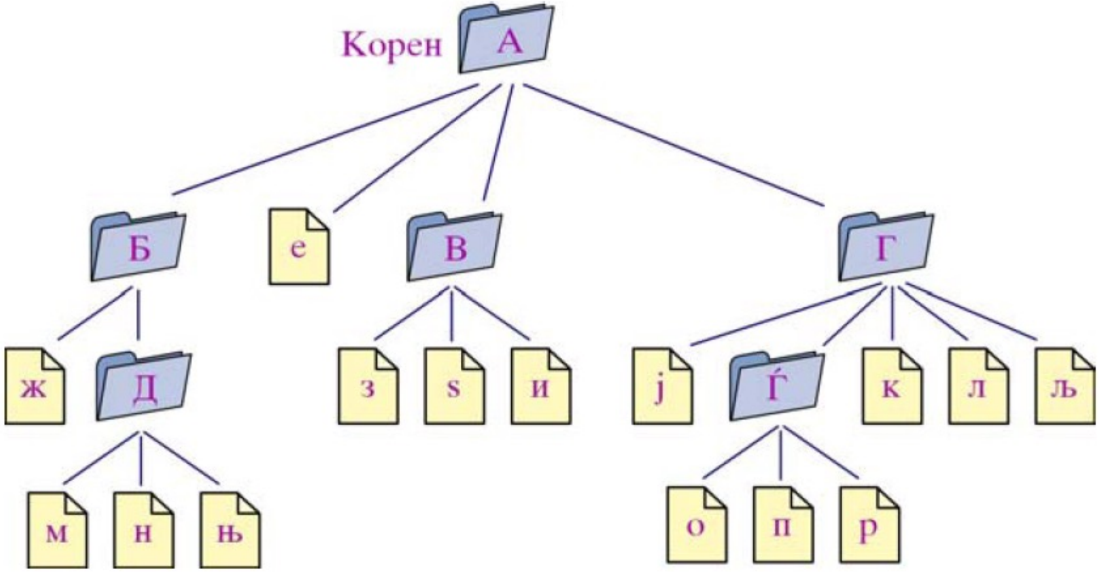
Дрва

- **Поддрво** на даден јазел во дрво е јазелот дете со сите свои наследници
- Бројот на поддрва на еден јазел се нарекува **степен на јазелот**
- **Висина** на јазел во стебло е должината на најдолгата патека од јазелот до листовите
- **Длабочина** на јазел е должината на единствената патека од коренот до јазелот

Дрва

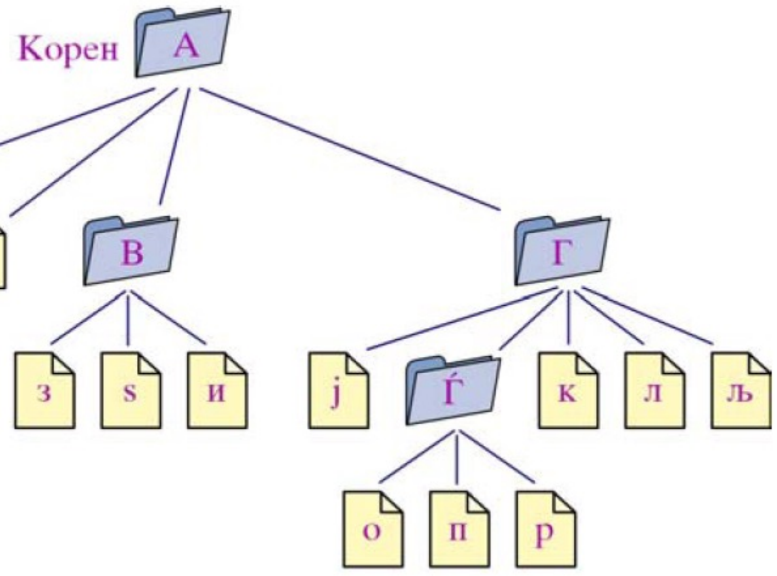


Дрва



патека од А до м: А Б Д м

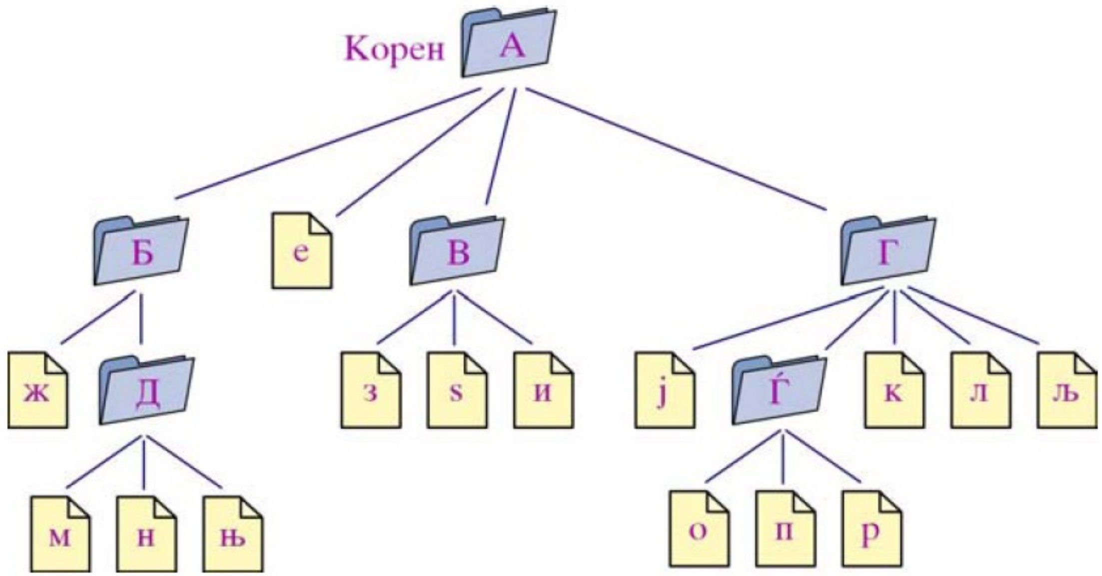
Дрва



патека од А до м: А Б Д м

наследници на Б: Д ж м н њ

Дрва

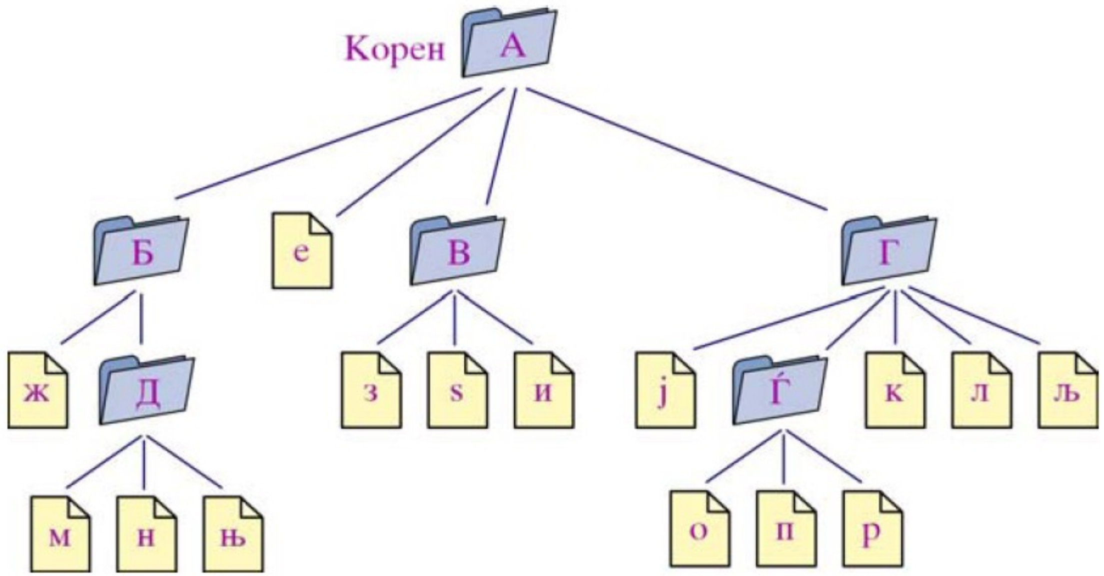


патека од А до м: А Б Д м

наследници на Б: Д ж м н њ

предци на м: Д Б А

Дрва



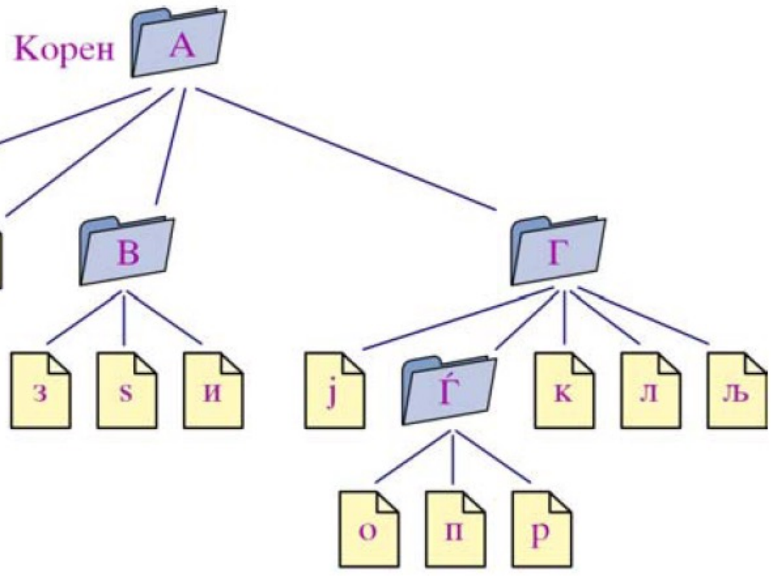
патека од А до м: А Б Д м

степен на А: 4

наследници на Б: Д ж м н њ

предци на м: Д Б А

Дрва



патека од А до м: А Б Д м

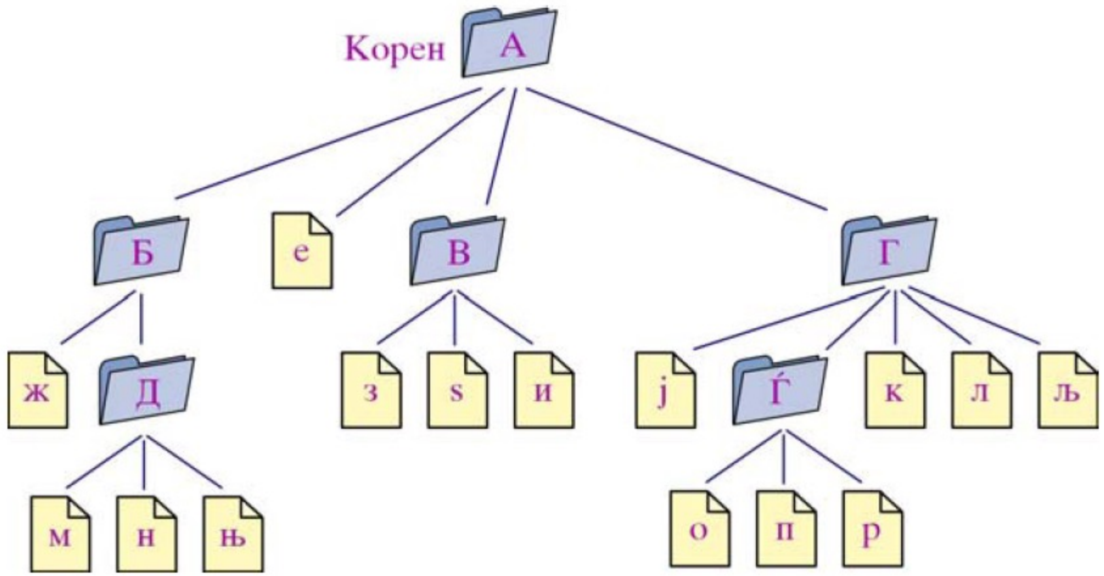
наследници на Б: Д ж м н њ

предци на м: Д Б А

степен на А: 4

степен на Г: 5

Дрва



патека од А до м: А Б Д м

наследници на Б: Д ж м н њ

предци на м: Д Б А

степен на А: 4

степен на Г: 5

степен на дрвото: 5

Репрезентација на дрвата

- Родителскиот јазел да содржи покажувачи кон своите деца јазли

Репрезентација на дрвата

- Родителскиот јазел да содржи покажувачи кон своите деца јазли

Проблем: Секој корен на подстеблата во едно дрво може да има произволен број на деца!

Репрезентација на дрвата

- Родителскиот јазел да содржи покажувачи кон своите деца јазли

Проблем: Секој корен на подстеблата во едно дрво може да има произволен број на деца!

Решение1: Предвидлив број на покажувачи во секој од јазлите на дрвото!

Репрезентација на дрвата

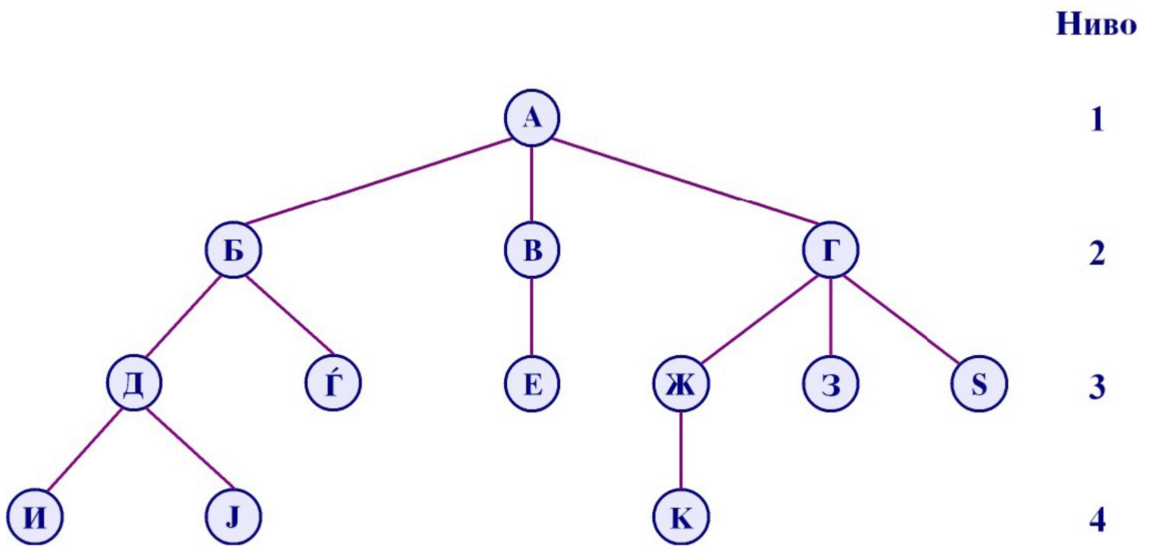
- Родителскиот јазел да содржи покажувачи кон своите деца јазли

Проблем: Секој корен на подстеблата во едно дрво може да има произволен број на деца!

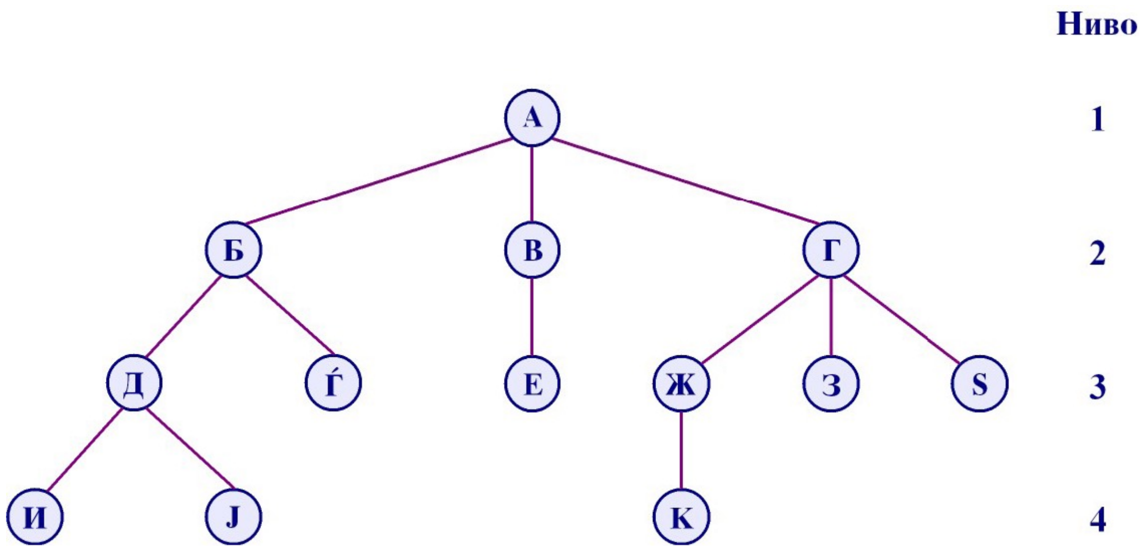
Решение1: Предвидлив број на покажувачи во секој од јазлите на дрвото!

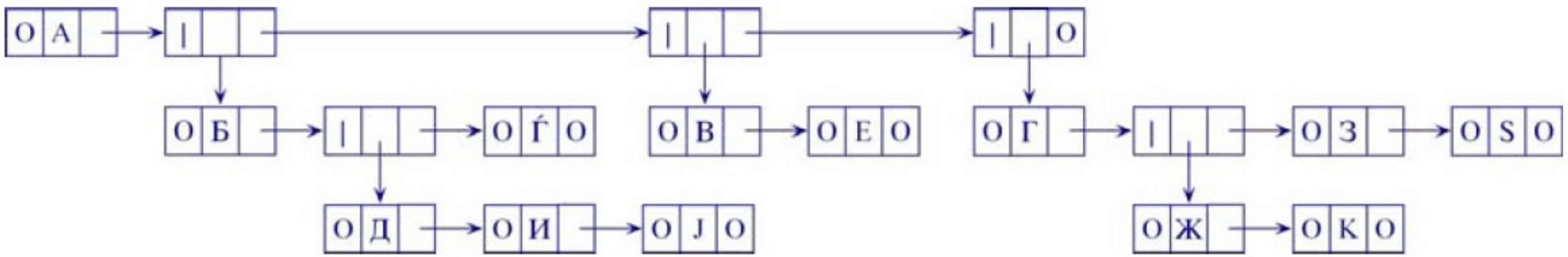
Решение2: Структура каде што бројот на покажувачи е најмногу два!

Репрезентација на дрва



Репрезентација на дрва



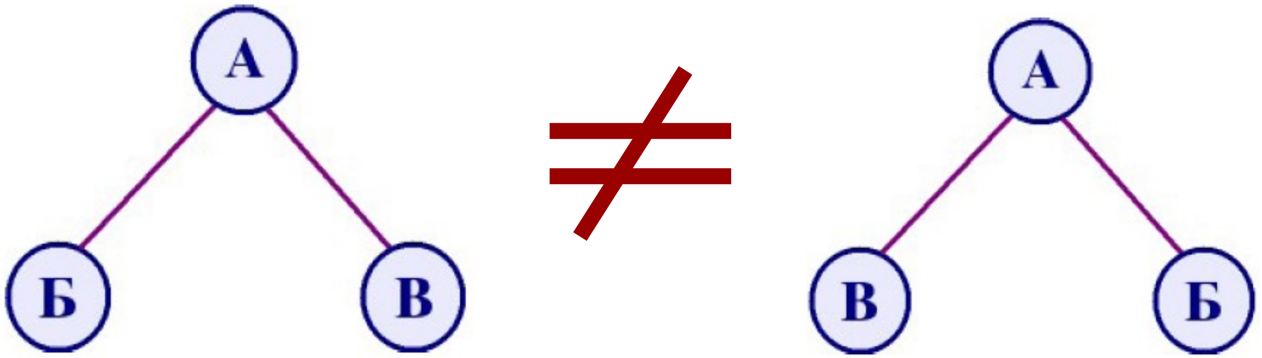


Подредени дрва

- Кога редоследот на поддрвата во едно дрво е важен, дрвото се нарекува **подредено дрво**

Подредени дрва

- Кога редоследот на поддрвата во едно дрво е важен, дрвото се нарекува **подредено дрво**



Бинарни дрва

- секој јазел може да има најмногу две поддрва
  - лево поддрво
  - десно поддрво
- степенот на дрвото изнесува два
- јазлите во бинарното стебло може да:
  - немаат наследници
  - имаат еден или
  - најмногу два наследници

Бинарни дрва

- Доколку се земе дека коренот на едно бинарно дрво е на ниво 1, а на секое наредно ниво бројот на јазли е поголем два пати, тогаш максималниот број на јазли на ниво *i* кај бинарните дрва е 2^{i-1}, $i>=1$
- Максималниот вкупен број *n* на јазли во бинарното дрво (број на јазли за максимално пополнето дрво) со длабочина *d* изнесува 2^d-1, $d>=1$. Тоа се добива од равенката:

$$n=\sum_{i=1}^{d}2^{i-1}=2^{d}-1$$

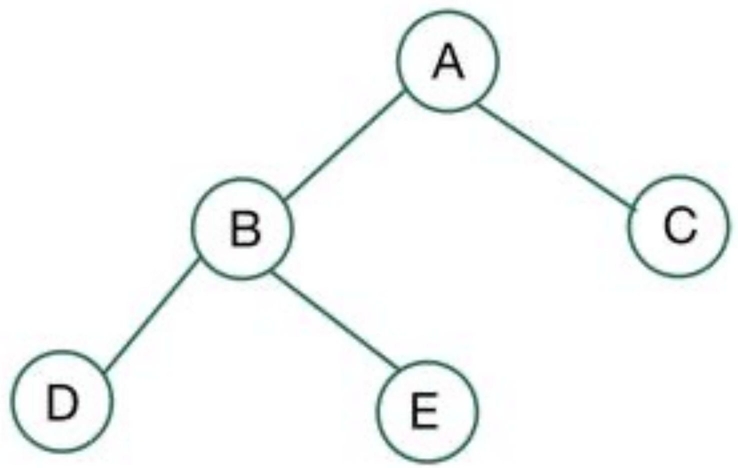
Бинарни дрва

- Од претходната равенка следи дека: $n \leq 2^d - 1$
- Од тука, ако е познат вкупниот број на јазли, за длабочината на дрвото се добива дека $d \geq \log_2(n+1)$
- Кога стеблото е максимално пополнето, тоа има најмала длабочина од сите можни бинарни дрва со вкупен број на јазли *n*, и таа длабочина изнесува

$$d = \lceil \log_2(n+1) \rceil$$

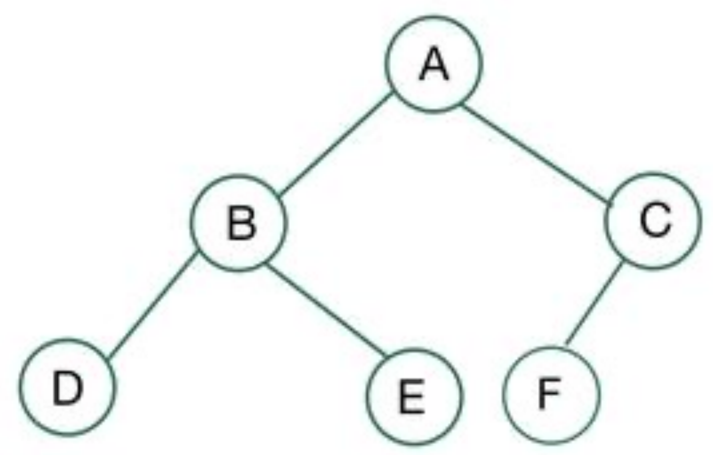
Полно (full) бинарно дрво

- Полно бинарно дрво е бинарно дрво во кое сите јазли имаат или 0 или 2 деца.
- Полно бинарно дрво е бинарно дрво во кое сите јазли, освен јазлите на листовите, имаат две деца.

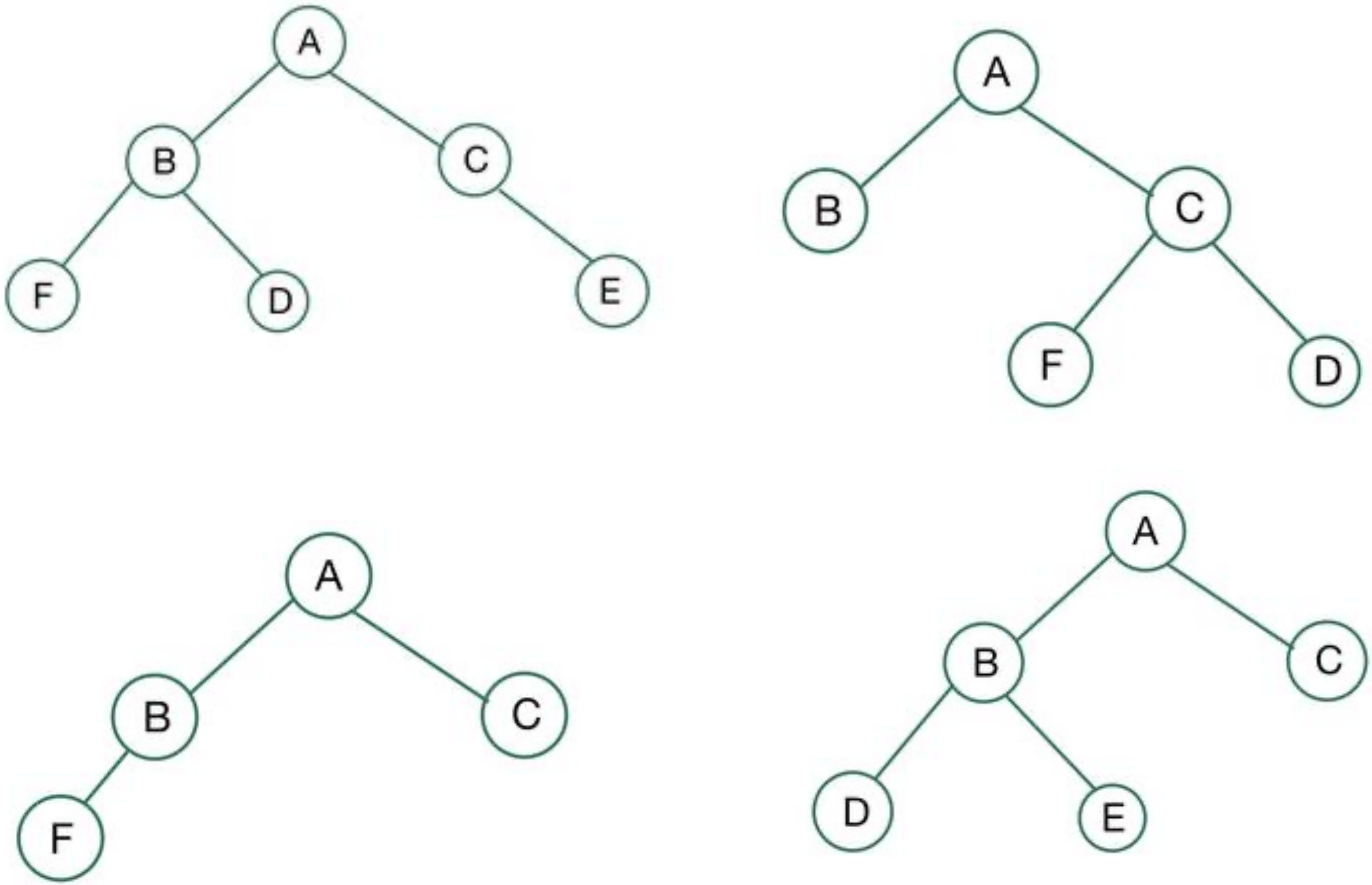


Комплетно (complete) бинарно дрво

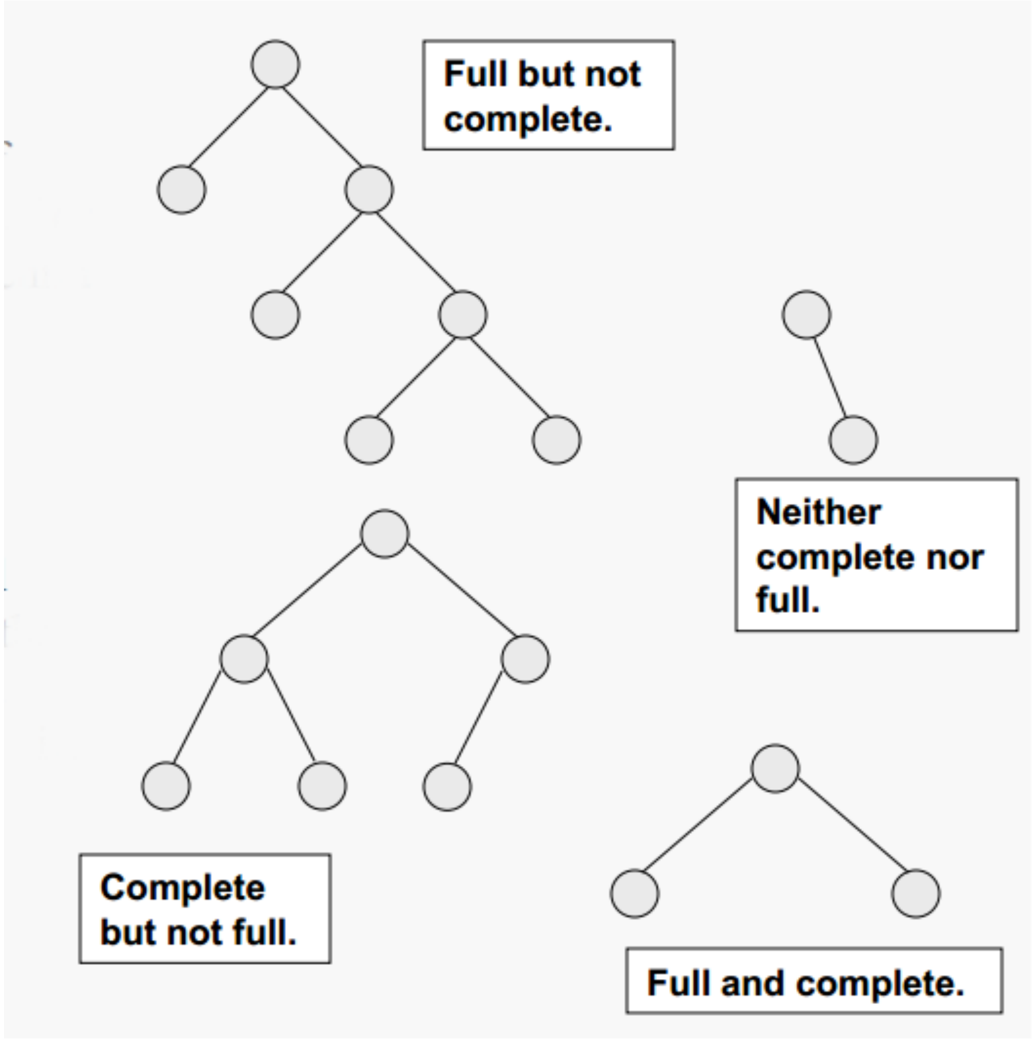
- Кога сите нивоа на бинарното дрво се целосно пополнети, освен последното ниво, кое може да содржи 1 или 2 деца и е исполнето од лево, се вели дека е комплетно бинарно дрво.



Какви се следните бинарни дрва?

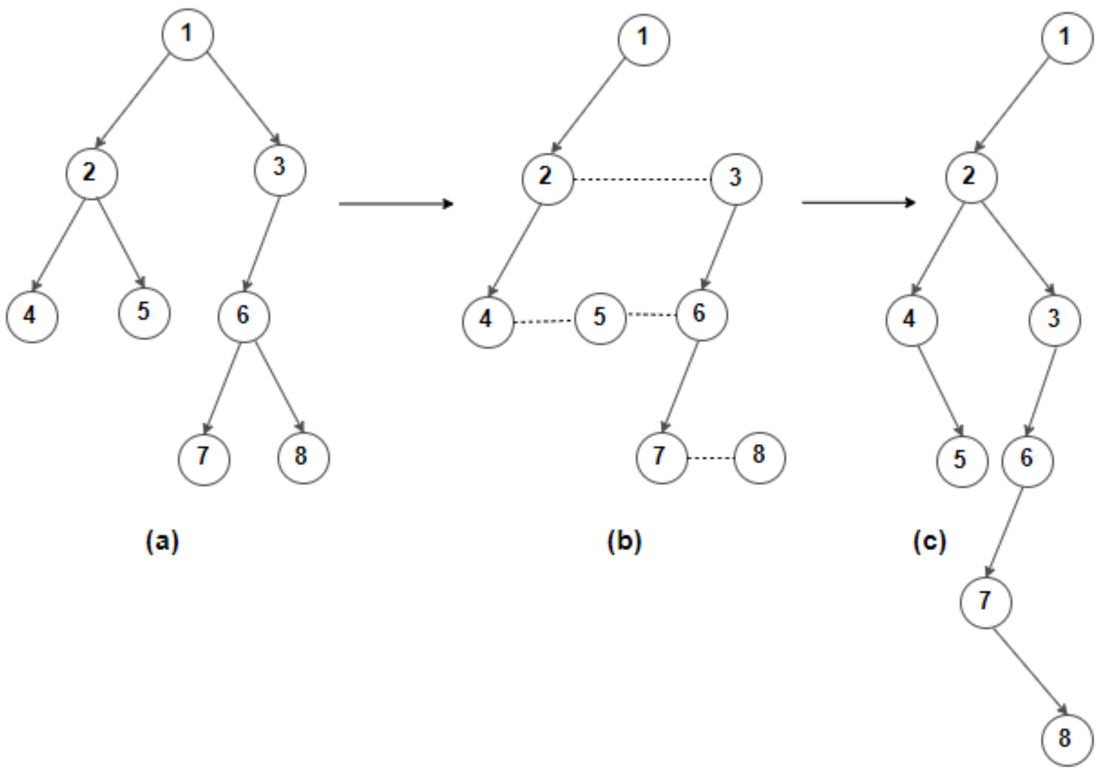


Примери

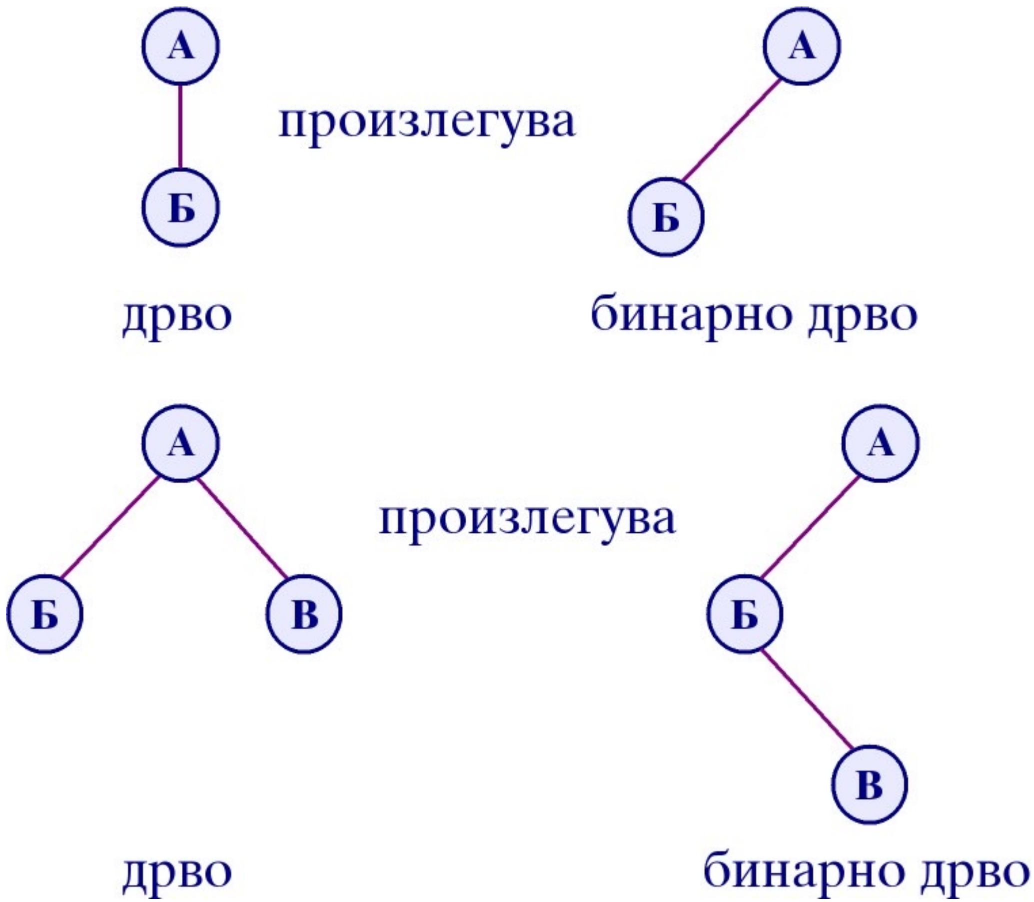

Full but not complete.
Neither complete nor full.
Complete but not full.
Full and complete.

Трансформација на дрво во бинарно дрво

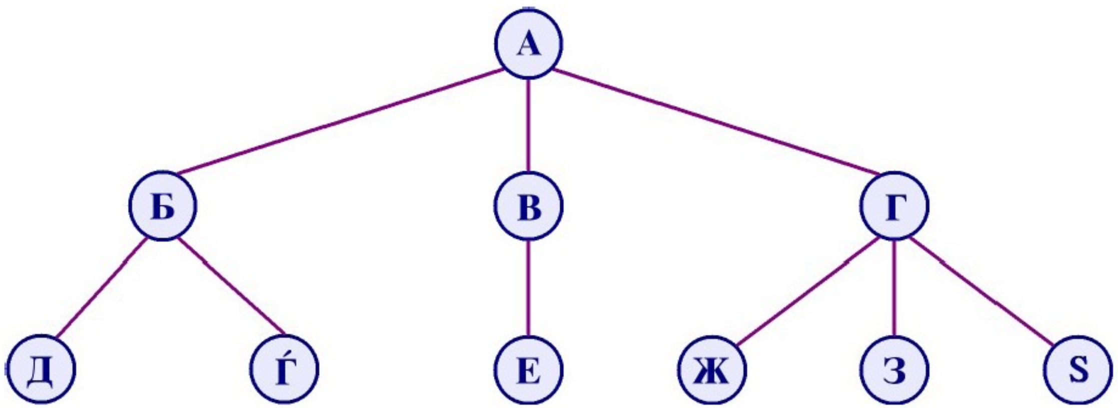
- Секое дрво може да се трансформира во бинарно дрво, така што еден од јазлите кои се наоѓаат на исто ниво ќе стане родител на сите останати.
- “десен брат – лево дете” трансформација



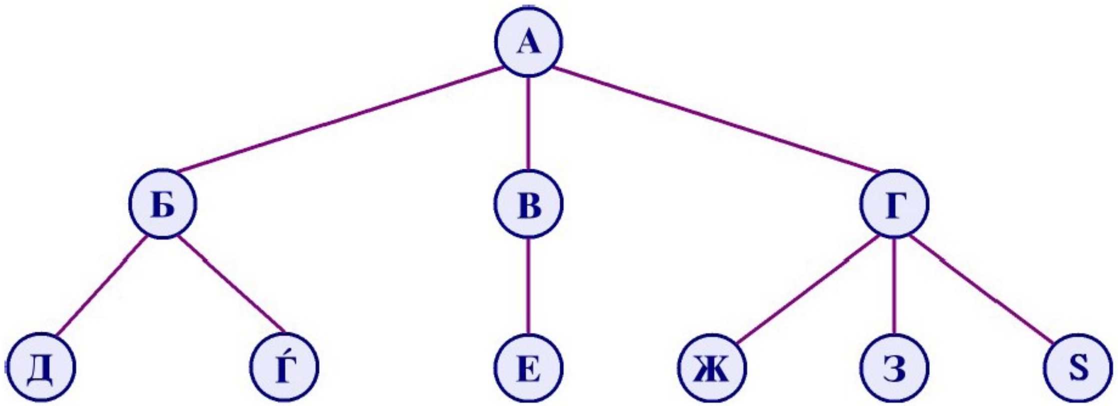
Репрезентација на едноставни дрва со бинарно дрво

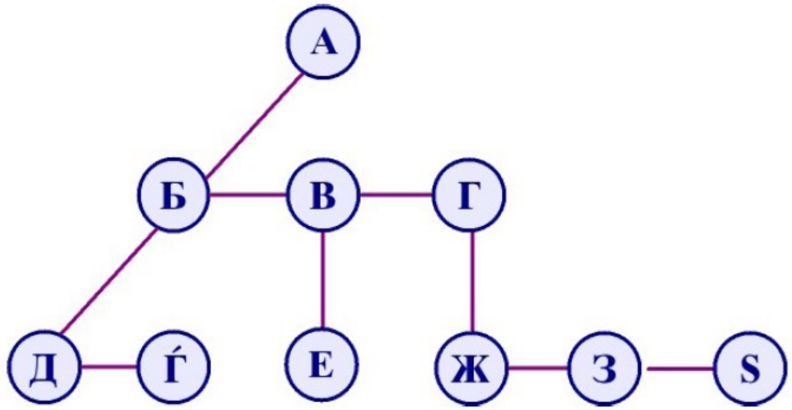


Трансформација на дрво во бинарно дрво - пример

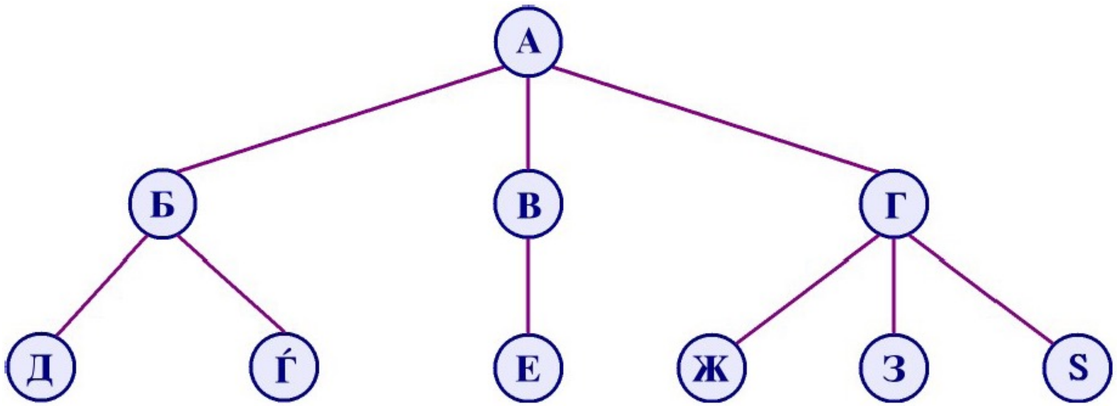


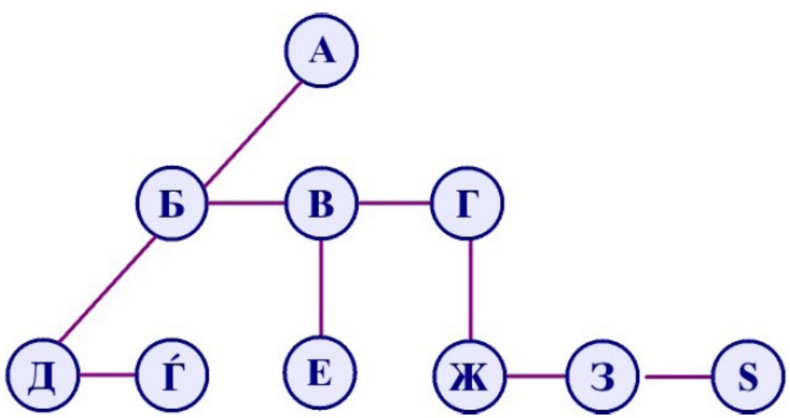
Трансформација на дрво во бинарно дрво - пример

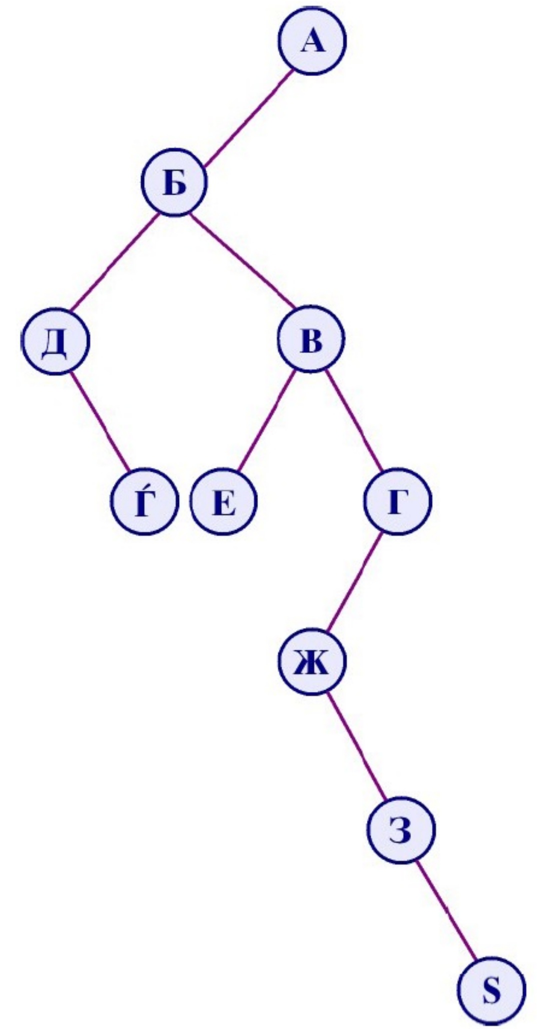




Трансформација на дрво во бинарно дрво - пример





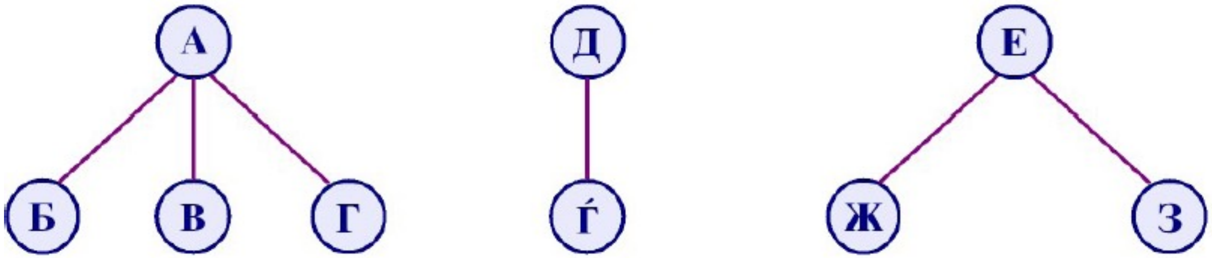


Трансформација на шума од дрва во бинарно дрво

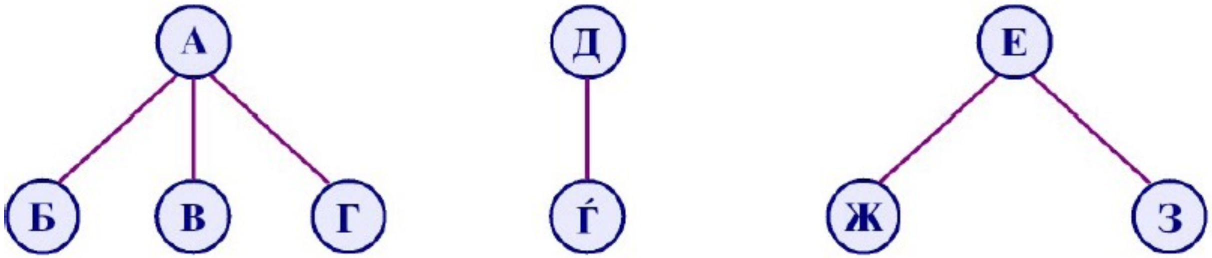
- Нека $T_1, \ldots, T_n$ е шума на дрва, тогаш бинарното дрво кое се добива со трансформацијата на оваа шума може да се означи со $B(T_1, \ldots, T_n)$ и за него важи:

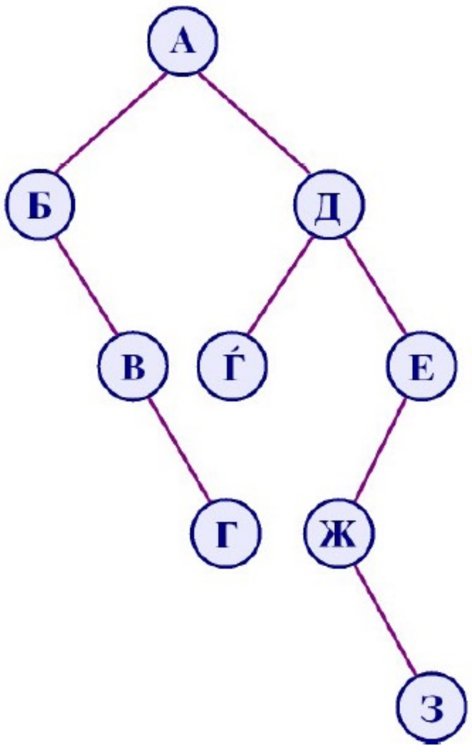
- $B(T_1, \ldots, T_n)$
  - е празно ако $n = 0$;
  - има корен еднаков со коренот (T_1);
  - има лево поддрво $B(T_{11}, T_{12}, \ldots, T_{1m})$ каде $T_{11}, \ldots, T_{1m}$ се поддрва на коренот (T_1);
  - има десно поддрво $B(T_2, \ldots, T_n)$

Трансформација на шума од дрва во бинарно дрво



Трансформација на шума од дрвa во бинарно дрво



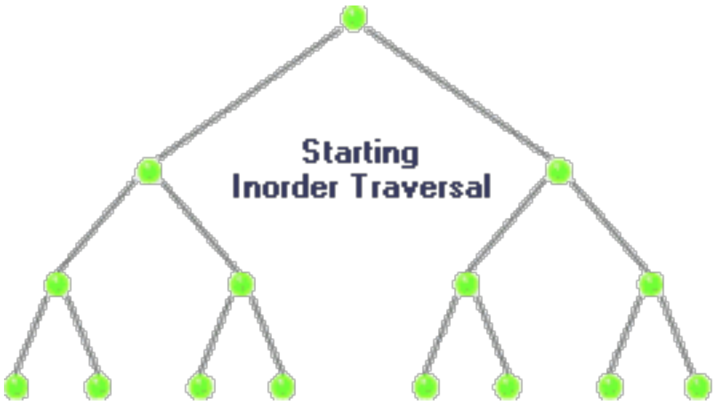


Изминување на бинарните дрва

- Постојат три основни начини на кои може да се изминат сите јазли во едно бинарно дрво:
  - inorder
  - preorder
  - postorder

Inorder изминување на бинарно дрво

- Изминете го левото поддрво, т.е. повикајте Inorder(лево дете->поддрво)
- Посетете го коренот.
- Изминете го десното поддрво, т.е. повикајте Inorder(десно дете-> поддрво)



Preorder изминување на бинарно дрво

- Посетете го коренот.
- Изминете го левото поддрво, т.е., повикајте Preorder(лево дете-> поддрво)
- Изминете го десното поддрво, т.е. повикајте Preorder(десно дете-> поддрво)

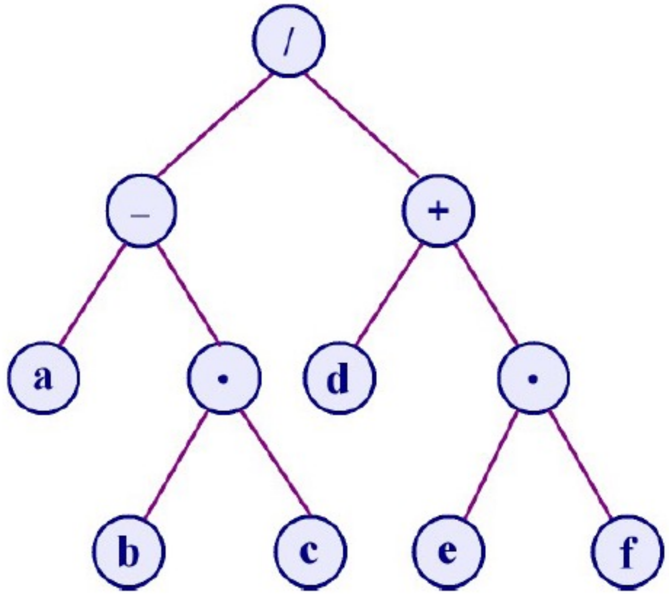
Postorder изминување на бинарно дрво

- Изминете го левото поддрво, т.е., повикајте го Postorder(лево дете -> поддрво)
- Изминето го десното поддрво, т.е., повикајте го Postorder(десно дете -> поддрво)
- Посетете го коренот

Изминување на бинарните дрва

- Постојат три основни начини на кои може да се изминат сите јазли во едно бинарно дрво:
  - inorder
  - preorder
  - postorder

Пример: Бинарно дрво за прикажување и пресметување на аритметички изрази

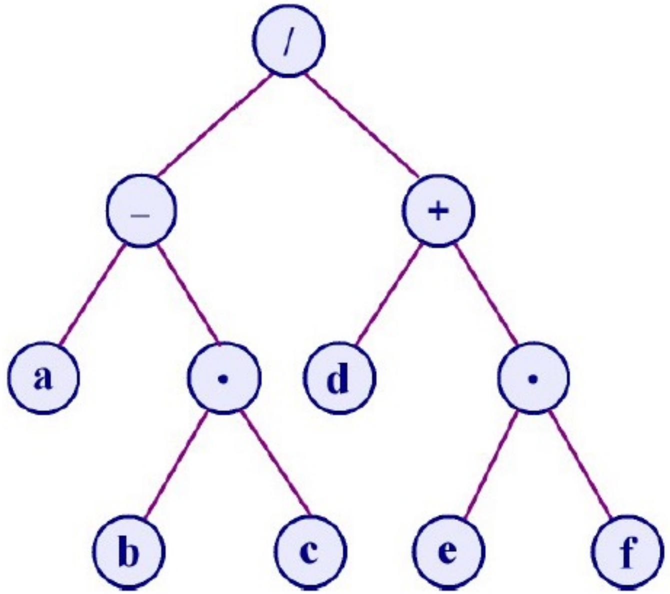


Изминување на бинарните дрва

- Постојат три основни начини на кои може да се изминат сите јазли во едно бинарно дрво:
  - inorder
  - preorder
  - postorder

a - b * c / d + e * f

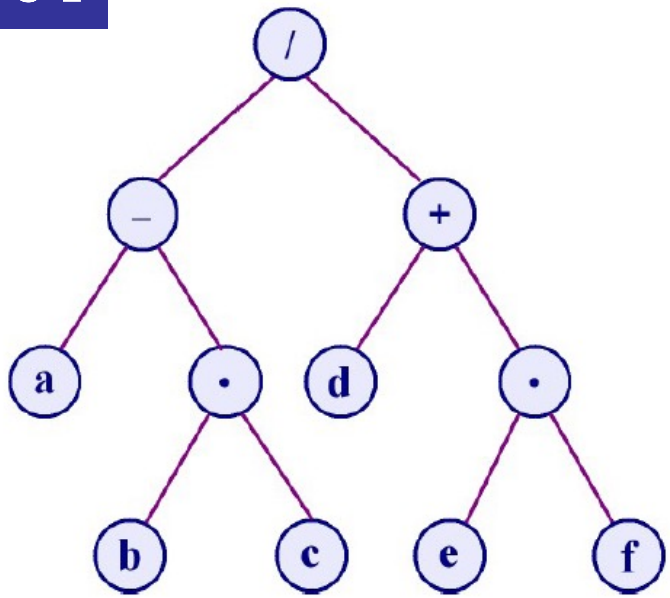
Пример: Бинарно дрво за прикажување и пресметување на аритметички изрази



Изминување на бинарните дрва

- Постојат три основни начини на кои може да се изминат сите јазли во едно бинарно дрво:
  - inorder   a - b * c / d + e * f
  - preorder   / - a * b c + d * e f
  - postorder

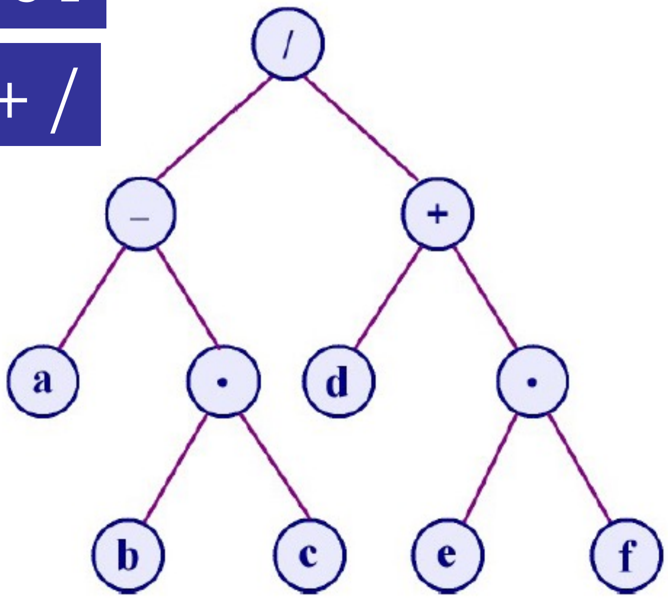
Пример: Бинарно дрво за прикажување и пресметување на аритметички изрази



Изминување на бинарните дрва

- Постојат три основни начини на кои може да се изминат сите јазли во едно бинарно дрво:
  - inorder a - b * c / d + e * f
  - preorder / - a * b c + d * e f
  - postorder a b c * - d e f * + /

Пример: Бинарно дрво за прикажување и пресметување на аритметички изрази



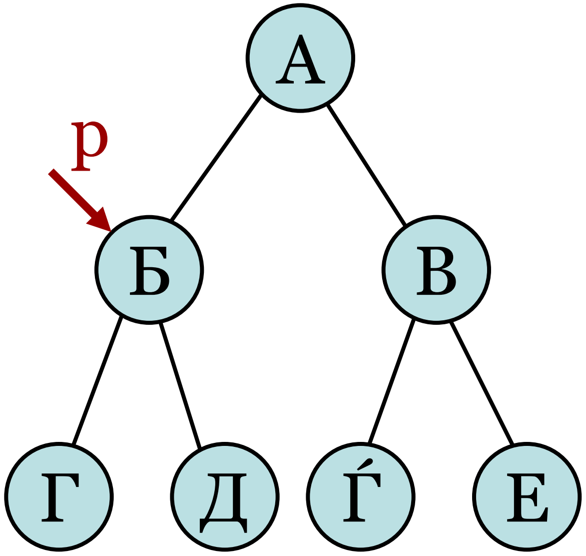
Изминување на бинарните дрва

Рекурзивните реализации на овие изминувања се тривијални!!

Основни манипулации со јазлите

- Додавање на јазел во бинарно дрво

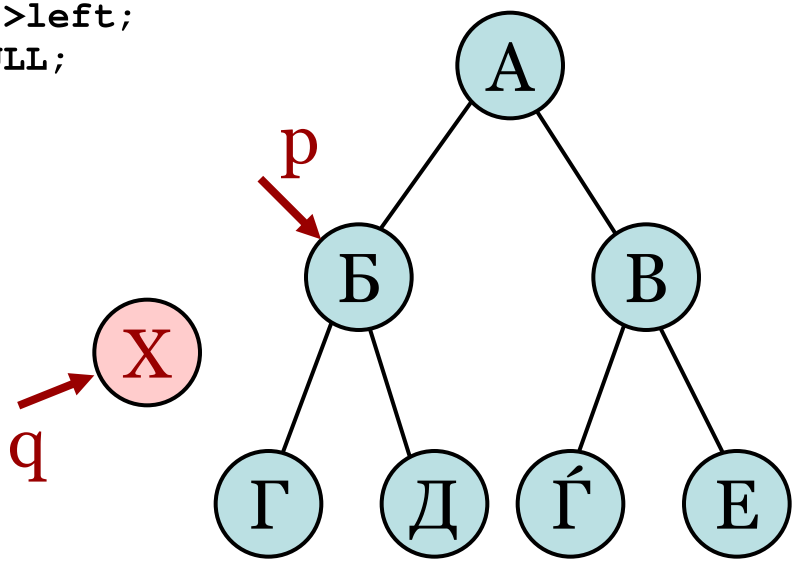
```
void vmetni_levo(nodep p, info_t x)
{
   nodep q=NEW(node);
   q->info=x;
   q->left= p->left;
   q->right=NULL;
   p->left=q;
}
```



Основни манипулации со јазлите

- Додавање на јазел во бинарно дрво

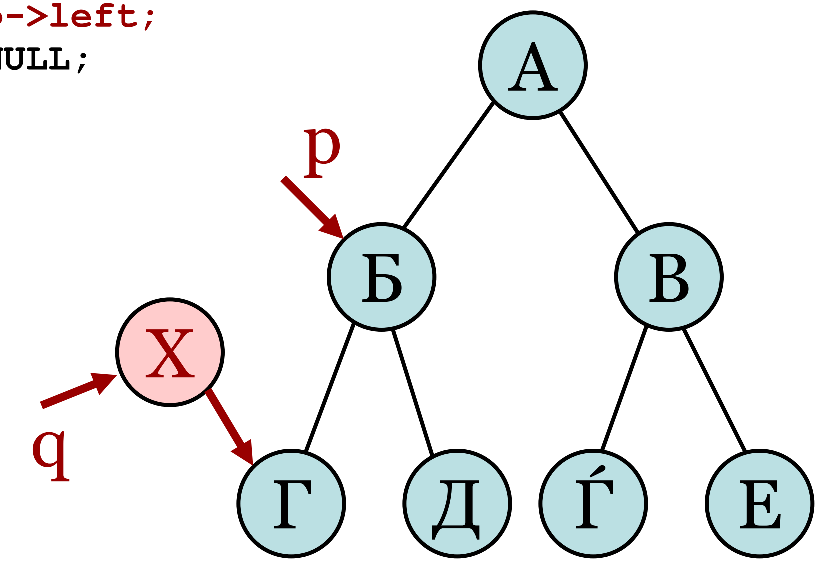
```
void vmetni_levo(nodep p, info_t x)
{
   nodep q=NEW(node);
   q->info=x;
   q->left= p->left;
   q->right=NULL;
   p->left=q;
}
```



Основни манипулации со јазлите

- Додавање на јазел во бинарно дрво

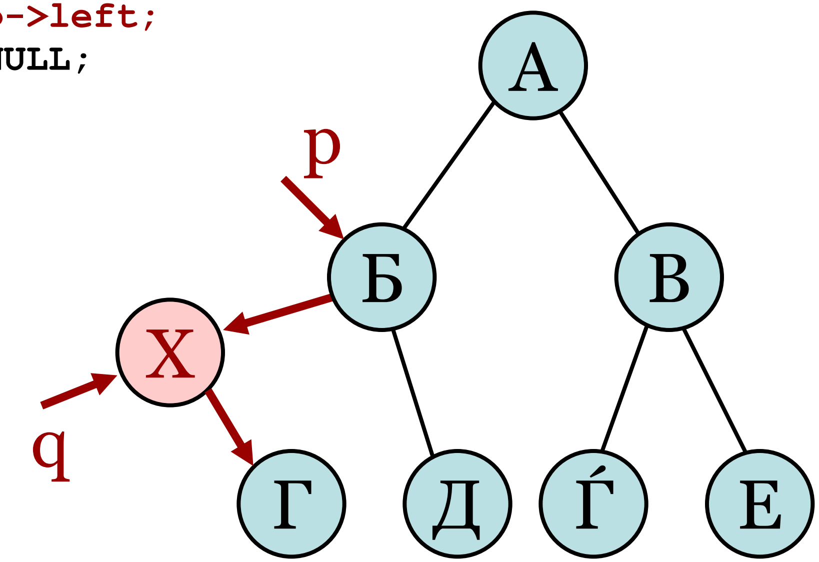
```
void vmetni_levo(nodep p, info_t x)
{
   nodep q=NEW(node);
   q->info=x;
   q->left= p->left;
   q->right=NULL;
   p->left=q;
}
```



Основни манипулации со јазлите

- Додавање на јазел во бинарно дрво

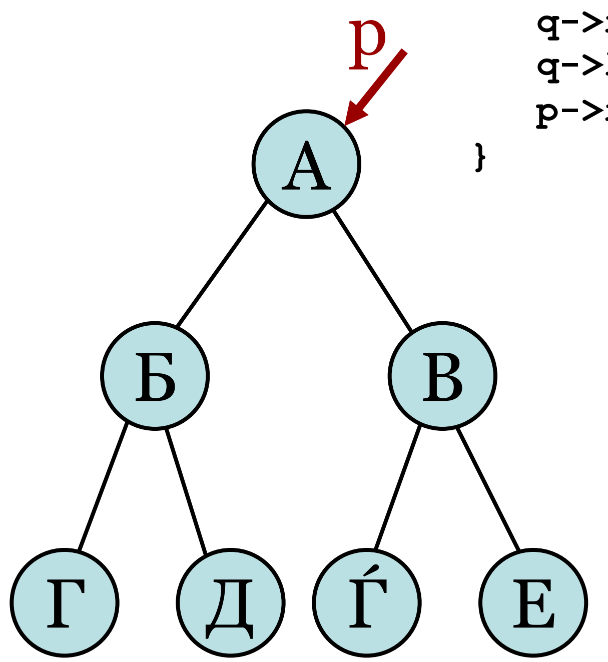
```
void vmetni_levo(nodep p, info_t x)
{
   nodep q=NEW(node);
   q->info=x;
   q->left= p->left;
   q->right=NULL;
   p->left=q;
}
```



Основни манипулации со јазлите

- Додавање на јазел во бинарно дрво

```
void vmetni_levo(nodep p, info_t x)
{
   nodep q=NEW(node);
   q->info=x;
   q->left= p->left;
   q->right=NULL;
   p->left=q;
}
```



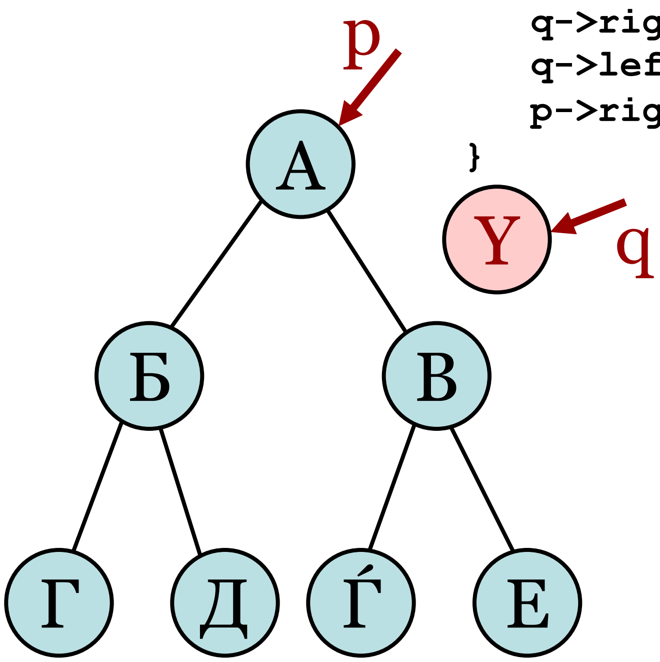

```
void vmetni_desno(nodep p, info_t y)
{
   nodep q=NEW(node);
   q->info=y;
   q->right = p->right;
   q->left=NULL;
   p->right=q;
}
```

Основни манипулации со јазлите

- Додавање на јазел во бинарно дрво

```
void vmetni_levo(nodep p, info_t x)
{
   nodep q=NEW(node);
   q->info=x;
   q->left= p->left;
   q->right=NULL;
   p->left=q;
}
```

```
void vmetni_desno(nodep p, info_t y)
{
   nodep q=NEW(node);
   q->info=y;
   q->right = p->right;
   q->left=NULL;
   p->right=q;
}
```

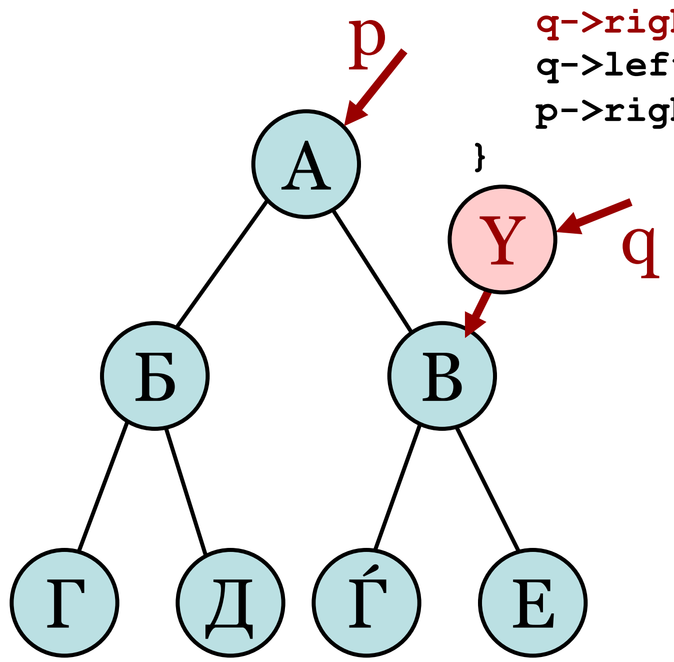


Основни манипулации со јазлите

- Додавање на јазел во бинарно дрво

```
void vmetni_levo(nodep p, info_t x)
{
   nodep q=NEW(node);
   q->info=x;
   q->left= p->left;
   q->right=NULL;
   p->left=q;
}
```

```
void vmetni_desno(nodep p, info_t y)
{
   nodep q=NEW(node);
   q->info=y;
   q->right = p->right;
   q->left=NULL;
   p->right=q;
}
```

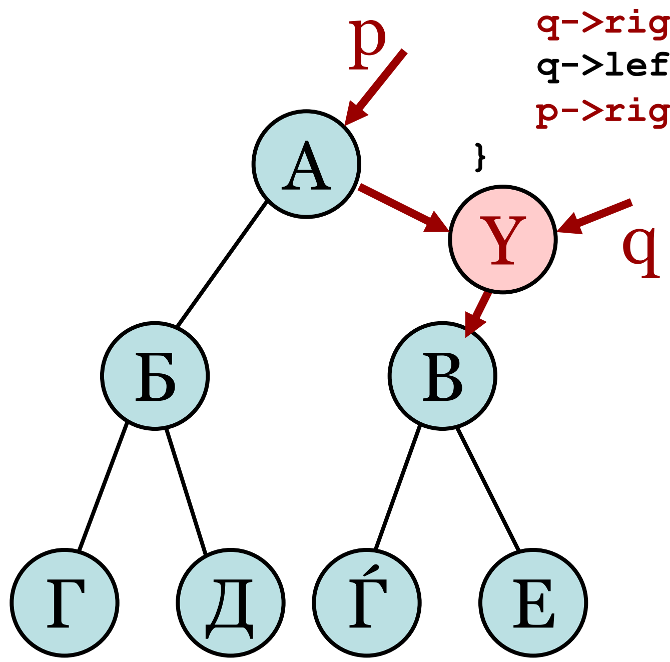


Основни манипулации со јазлите

- Додавање на јазел во бинарно дрво

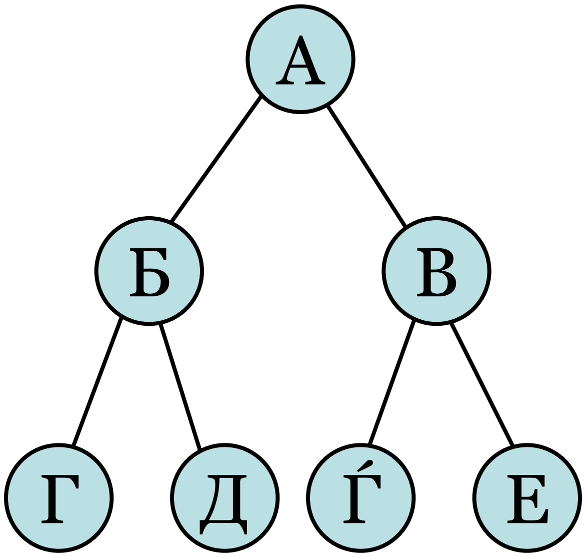
```
void vmetni_levo(nodep p, info_t x)
{
   nodep q=NEW(node);
   q->info=x;
   q->left= p->left;
   q->right=NULL;
   p->left=q;
}
```

```
void vmetni_desno(nodep p, info_t y)
{
   nodep q=NEW(node);
   q->info=y;
   q->right = p->right;
   q->left=NULL;
   p->right=q;
}
```



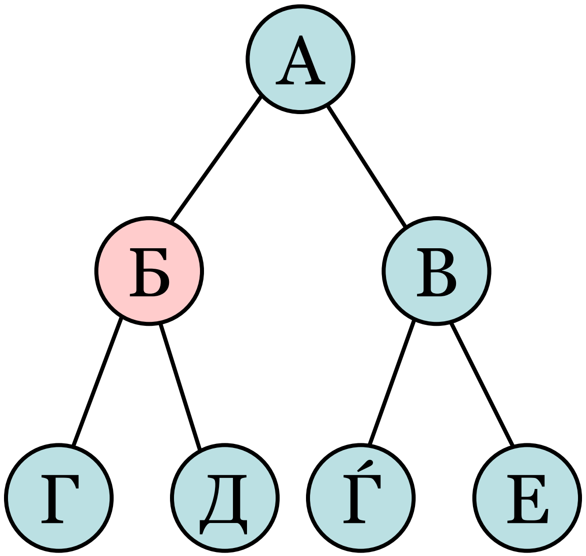
Основни манипулации со јазлите

- Бришење на јазел во бинарно дрво



Основни манипулации со јазлите

- Бришење на јазел во бинарно дрво



Проблем: Што ќе дојде на местото на избришаниот јазел за да имаме повторно структура на бинарно дрво?

Решение: избришаниот јазел треба да биде заменет со друг јазел (најчесто некој од неговото поддрво). Кога јазелот што се брише нема деца или пак има само едно дете, алгоритамот е тривијален.