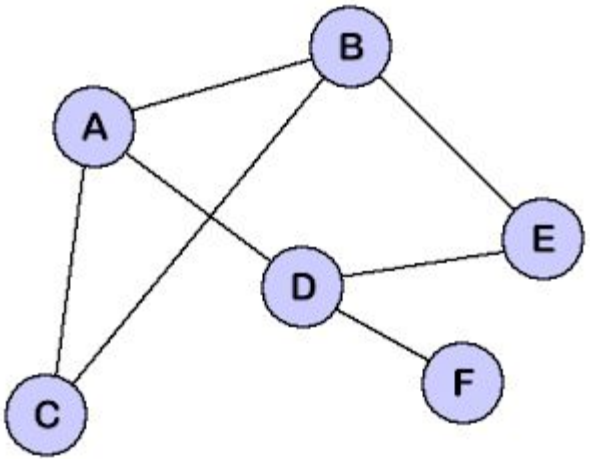
Grafy a Stromy

Dátové štruktúry a algoritmy 24/25 LS
Prednášky, garant: prof. Gabriel Juhás
Cvičenia: Milan Mladoniczky - milan.mladoniczky@paneurouni.com

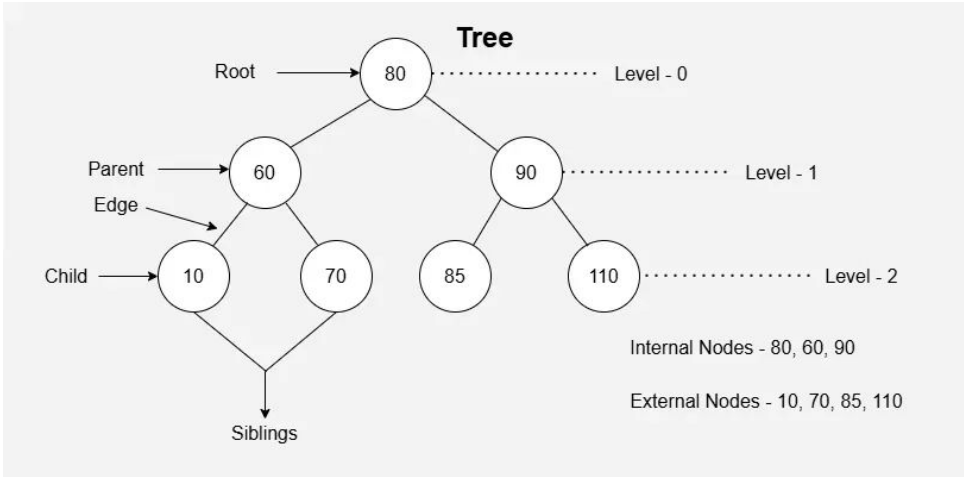
Graf

- Nelineárna dátová štruktúra pozostávajúca z hrán a vrcholov (uzlov).
- Kde vrchol môže byť spojený v 0-N inými uzlami hranami.
- Grafy môžu byť orientované, kedy hrana má definovaný smer (napr. pri analýze toku)
- Hrany môžu mať priradenú hodnotu, vtedy hovoríme o ováhovanom grafe.



Strom

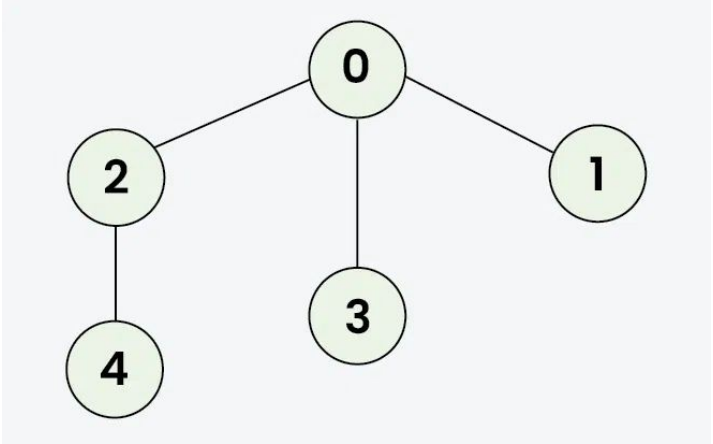
- Hierarchická štruktúra, kde medzi vrcholmi je jasne definovaný vzťah rodiča a dieťaťa.
- Špeciálny typ grafu
- Strom má jeden začínajúci vrchol -> root
- Poradie detí vrcholu môže byť určené algoritmom, vtedy hovoríme o zoradených stromoch.
- Ak je maximálny počet detí vrcholu 2 hovorí o binárnom strome.



Funkcia	Graf	Strom
Definícia	Kolekcia uzlov (vrcholov) a hrán, kde hrany spájajú uzly.	Hierarchická dátová štruktúra pozostávajúca z uzlov spojených hranami s jediným koreňovým uzlom.
Štruktúra	Môže obsahovať cykly a nesúvisiace komponenty.	Nemá cykly; súvislá štruktúra s presne jednou cestou medzi akýmikoľvek dvoma uzlami.
Koreňový uzol	Nemá koreňový uzol; uzly môžu mať viacerých rodičov alebo žiadneho.	Má určený koreňový uzol, ktorý nemá rodiča.
Vzťah medzi uzlami	Vzťahy medzi uzlami sú ľubovoľné.	Vzťah rodič-dieťa; každý uzol (okrem koreňa) má presne jedného rodiča.
Hrany	Každý uzol môže mať ľubovoľný počet hrán.	Ak je n uzlov, potom je počet hrán n-1.
Zložitosť prehľadávania	Prehľadávanie môže byť zložité kvôli cyklom a nesúvislým komponentom.	Prehľadávanie je jednoduché a môže sa vykonať v lineárnom čase.
Použitie	Používa sa v rôznych scenároch, ako sú sociálne siete, mapy, optimalizácia sietí atď.	Bežne sa používa na reprezentáciu hierarchických dát, ako sú súborové systémy, organizačné diagramy, HTML DOM, XML dokumenty atď.
Príklady	Sociálne siete, cestné siete, počítačové siete.	Súborové systémy, rodokmene, HTML DOM štruktúra.

Prechádzanie do šírky

- Prechádzanie vrcholov hneď ako na nich algoritmus narazí.
- Až po prejdený jednej úrovne sa algoritmus posunie na ďalsiu.
- Základný algoritmu prechádzania grafu alebo stromu.



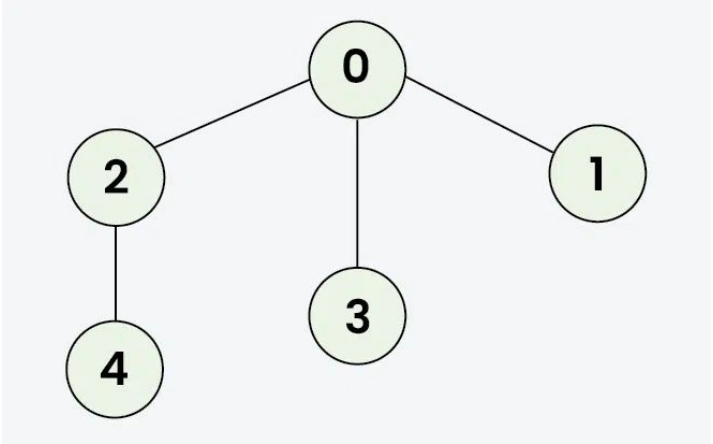

```
Root = 0

Začíname v roote // 0
Prvý dieťa roota -> 2 // 0,2
Druhé dieťa roota -> 3 // 0,2,3
Tretie dieťa roota -> 1 // 0,2,3,1
Posun na uzol ďalšej úrovne -> 2
Prvé dieťa uzla 2 -> 4 // 0,2,3,1,4

Koniec, prejdené všetky uzly
```

Prechádzanie do hĺbky

- Prechádzanie vrcholov tak, že sa hneď vnoríme a sa postupne "vynárame".
- Až po prejedení všetkých detí uzla sa posúvame na súrodenca.
- Základný algoritmu prechádzania grafu alebo stromu.




```
Root = 0

Začíname v roote // 0
Vnorenie do uzla 2 // 0,2
Vnorenie do uzla 4 // 0,2,4
Vynorenie do 2, vynorenie do root
Vnorenie do uzla 3 // 0,2,4,3
Vynorenie do root
Vnorenie do uzla 1 // 0,2,4,3,1

Koniec, prejdené všetky uzly
```

Rekurzia

- Funkcia vo svojom tela volá samú seba.
- V rekurzívnej funkcií je povinná terminujúca podmienka, ktorý ukončuje ďalšie rekurzívne vnáranie.

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


int fact(int n) {
    if (n == 0)
        return 1;
    return n * fact(n - 1);
}


int main() {
    cout <<"Factorial of 5 : "<< fact(5);
    return 0;
}
```

Poďme implementovať jednoduchý strom ->

https://dsa.interes.group/exercises/exercise-4