



9. Prehľadávanie grafov

9.1 Úvod

V celej tejto kapitole budeme pracovať len so súvislými obyčajnými grafmi. Na konci kapitoly 3 bol v podkapitole 3.5 (strana 62) definovaný pojem *kostra* grafu. Kostra grafu sa v aplikáciách často využíva a potrebujeme ju všade tam, kde je potreba nájsť „najmenší“ súvislý faktor daného grafu. Pritom pojem „najmenší“ je tu chápaný v zmysle počtu hrán. Algoritmus hľadania kostry grafu, použitý v dôkaze vety 3.5.1, bol veľmi neefektívny a prakticky nepoužiteľný. Avšak jeho triviálnou modifikáciou dostaneme nasledujúci algoritmus hľadania kostry daného grafu, ktorý je oveľa lepší/efektívnejší.

Konštrukcia kostry grafu

VSTUP: Súvislý obyčajný graf $G = (V, E)$.

VÝSTUP: $S = (V', E')$ – kostra grafu G .

INICIALIZÁCIA

$V' := V$;

$E' := \{\}$;

VYTVÁRANIE KOSTRY S

```
for { pre všetky hrany  $h \in E$  } do
  if { pridaním hrany  $h$  do  $E'$  nevznikne cyklus v  $S$  } then
     $E' := E' \cup \{h\}$ ;
  end
  if {  $|E'| = |V'| - 1$  } then
    STOP;
  end
end
end
```

Zistiť, či pridaním jednej konkrétnej hrany vznikne cyklus, je oveľa rýchlejšia operácia, než hľadanie všetkých cyklov v danom grafe. Preto je modifikovaný algoritmus, ktorý je uvedený na predošlej strane, oveľa efektívnejší než predtým uvedená alternatíva hľadania všetkých cyklov v danom grafe. Ale ani tento modifikovaný algoritmus hľadania kostry daného grafu ešte nie je optimálny. Ešte rýchlejšie a efektívnejšie sú dva prehľadávacie algoritmy, ktoré si uvedieme v nasledujúcich podkapitolách tejto kapitoly. Sú to algoritmy prehľadávania grafu *do šírky* a prehľadávania grafu *do hĺbky*.

9.2 Prehľadávanie grafu

Skôr, než uvedieme popis algoritmov prehľadávania grafu do šírky a do hĺbky, uvedieme si dátové štruktúry *fronta* a *zásobník*, ktoré nám popis oboch algoritmov zjednotia a zjednodušia. Pri prehľadávaní grafov budeme ich hrany ukladať práve do fronty, resp. do zásobníka.

9.2.1 Fronta (FIFO) a zásobník (LIFO)

Fronta (po anglicky *queue*) a *zásobník* (po anglicky *stack*) sú v informatike často používané dátové štruktúry, slúžiace na dočasné ukladanie dát. Tieto dve štruktúry sa odlišujú spôsobom prístupu k uloženým dátam.

Fronta – FIFO

Skratka FIFO znamená „*First-In-First-Out*“. Čiže dáta, ktoré ako prvé do fronty uložíme, z neho aj ako prvé vyberieme. Takúto dátovú štruktúru si môžeme predstaviť ako „rúru“, do ktorej z jednej strany veci vkladáme a z druhej vyberáme. Takže to, čo sme do „rúry“ skôr vložili, z nej na druhej strane musíme aj skôr vybrať. FIFO fronta je napríklad výstupný žľab pre ľah lotérie, aký vidíme na obrázku 9.1. Frontou je tiež rad ľudí stojacich pri pokladni v obchode. Ľudia, ktorí poznajú operačné systémy UNIX-ového typu, sa už určite stretli s pojmom *pipe*, alebo po slovensky *rúra*. Pomocou rúr sa v UNIX-e dajú zrefazovať výstupy a vstupy príkazov, čo je častá prax. *Pipe* v UNIX-e je vlastne fronta (FIFO).



Obr. 9.1: Príklad fronty (FIFO)



Obr. 9.2: Príklad zásobníka (LIFO)

Zásobník – LIFO

Skratka LIFO znamená „*Last-In-First-Out*“. Čiže dáta, ktoré sme ako posledné do zásobníka vložili, z neho ako prvé vyberieme. Typický príklad LIFO zásobníka, je zásobník do pištole, aký vidíme na obrázku 9.2.

Frontu budeme v algoritme označovať písmenom *F* a zásobník písmenom *Z*. Pri oboch budeme používať dve funkcie. Prvá bude funkcia *F.vyber()*, resp. *Z.vyber()*, ktorá nám z fronty, resp. zásobníka, vyberie nasledujúci objekt. Druhá funkcia bude *F.vloz(x)*, resp. *Z.vloz(x)*, ktorá nám do fronty resp. zásobníka, vloží objekt *x*.

Označovanie hrán a ich vrcholov

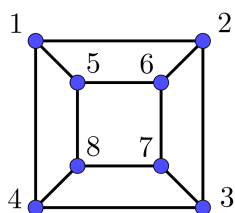
Pri prehľadávaní grafu, či už do šírky, alebo do hĺbky, budeme pracovať s neorientovanými grafmi. Čiže hrana medzi vrcholmi u a v bude dvojprvková množina (neusporiadaná dvojica) $\{u, v\}$. Samotné hrany budeme označovať písmenami $h, h' \dots$

Avšak pri prehľadávaní budeme konštruovať koreňový strom a jeho hranám dáme istú „orientáciu“. *Počiatočným vrcholom hrany* budeme nazývať ten jej vrchol, ktorý je bližšie ku koreňu stromu a *koncový vrchol hrany* bude ten jej vrchol, ktorý je ďalej od koreňa stromu. Z tohoto dôvodu zavedieme aj dve funkcie, ktoré v algoritmoch budeme používať. Funkcia *zaciatok* (h) nám vráti počiatočný vrchol hrany h a funkcia *koniec* (h) nám vráti koncový vrchol hrany h .

9.3 Prehľadávanie grafu do šírky

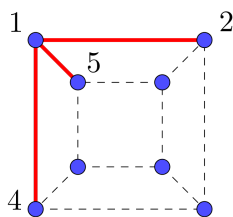
Kostra grafu je strom a pokiaľ si v strome vyberieme jeden konkrétny vrchol, tak dostávame koreňový strom. Algoritmus prehľadávania grafu do šírky, po anglicky „*Breadth-first search*“ (BFS), je založený na myšlienke prehľadávania vrcholov grafu v poradí, ktoré závisí od ich vzdialenosti od koreňa stromu. Preto je tento algoritmus vhodný na nájdenie najkratšej vzdialenosti medzi zvoleným vrcholom grafu (koreňom) a ľubovoľným iným vrcholom grafu. Všetky vrcholy na jednej úrovni koreňového stromu sa spracujú skôr, než prejdeme na ďalšiu úroveň. Čiže na začiatku máme len koreň stromu, nultú úroveň. V prvom kroku prehľadávania spracujeme všetky vrcholy prvej úrovne atď. Ukážeme si to na príkladoch.

■ Príklad 9.1

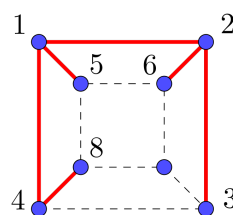
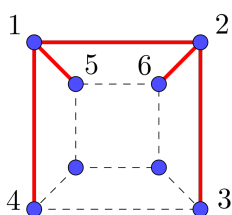


Na obrázku vľavo, je daný graf, ktorého vrcholy sú označené číslami 1 až 8. Nájdite jeho kostru prehľadávaním do šírky, pri lexikografickom usporiadaní vrcholov.

Riešenie: Začneme vrcholom 1 a v prvom kroku pridáme všetky s ním inciduujúce hrany a ich druhé koncové vrcholy, pri ktorých nám nevznikne cyklus. Dostaneme strom na nasledovnom obrázku

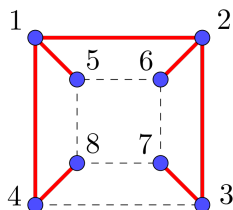


Týmto je zostrojená 1. úroveň koreňového stromu. K vrcholu 1 nám pribudli vrcholy 2, 4 a 5. Samozrejme pribudli aj hrany medzi týmito vrcholmi a koreňom stromu (vrcholom 1). V druhom kroku postupne prehľadáme vrcholy 2, 4 a 5 v uvedenom poradí a pridáme všetky s nimi inciduujúce hrany a ich druhé koncové vrcholy, pri ktorých nám nevznikne cyklus. Postup je zobrazený na nasledujúcich obrázkoch



Týmto je zostrojená 2. úroveň koreňového stromu. Prehľadaním vrcholu 2 nám pribudli vrcholy 3 a 6 a dve hrany medzi nimi a vrcholom 2. Prehľadaním vrcholu 4 nám pribudol vrchol 8 a

hrana medzi ním a vrcholom 4. Prehľadáním vrcholu 5 nám nepribudlo nič ďalšie, pretože ak by sme pridali do existujúceho stromu akúkoľvek s ním incidujúcu hranu, tak by nám vznikol cyklus. V ďalšom kroku ideme prehľadať vrcholy 3 a 6. Výsledok vidíme na ďalšom obrázku



Týmto je zostrojená 3. úroveň koreňového stromu. Prehľadáním vrcholu 3 nám pribudol vrchol 7 a hrana medzi ním a vrcholom 3. V tejto chvíli už koreňový strom obsahuje všetky vrcholy pôvodného grafu, takže je jeho kostrou. ■

Pseudokód algoritmu na prehľadávanie grafu do šírky môže vyzeráť napríklad takto

Prehľadávanie grafu do šírky

VSTUP: Súvislý obyčajný graf $G = (V, E)$ a pevne zvolený vrchol $v_1 \in V$.

VÝSTUP: $S = (V', E')$ – kostra grafu G .

INICIALIZÁCIA

for { pre všetky hrany $h \in E$, kde $\text{zaciatok}(h) = v_1$ } **do**

$F.\text{vloz}(h)$;

end

$E' := \{\}$;

$V' := \{v_1\}$;

VYTVÁRANIE KOSTRY S

while { $|V'| < |V|$ } **do**

$h := F.\text{vyber}()$;

if { $h \notin E'$ a pridaním hrany h do E' nevznikne cyklus v S } **then**

$E' := E' \cup \{h\}$;

$V' := V' \cup \{\text{koniec}(h)\}$;

for { pre všetky hrany $h' \in E$, kde $\text{zaciatok}(h') = \text{koniec}(h)$ } **do**

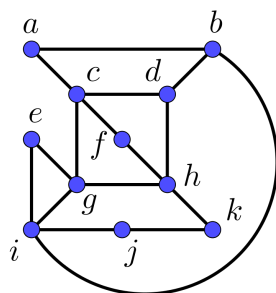
$F.\text{vloz}(h')$;

end

end

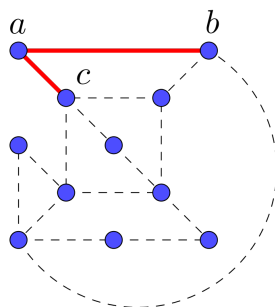
end

■ Príklad 9.2

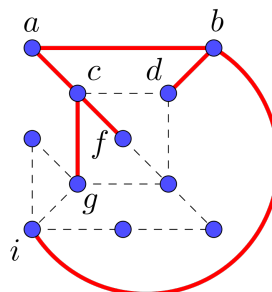
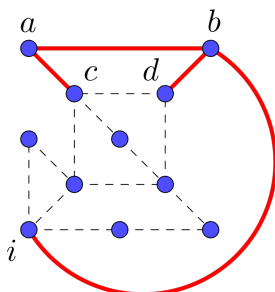


Na obrázku vľavo je daný graf, ktorého vrcholy sú označené písmenami a až k . Nájdite jeho kostru prehľadávaním do šírky, pri lexikografickom usporiadaní vrcholov.

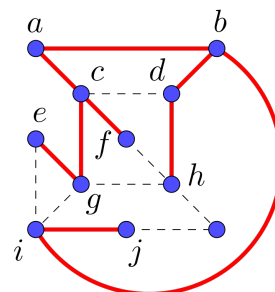
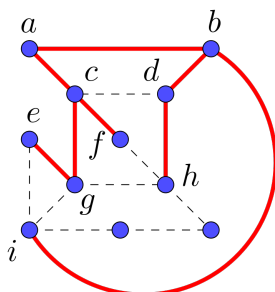
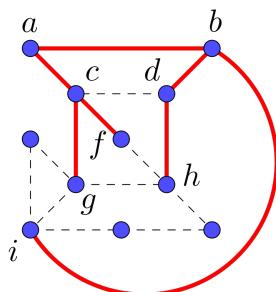
Riešenie: Začneme vrcholom a a v prvom kroku pridáme všetky s ním inciduujúce hrany a ich druhé koncové vrcholy, pri ktorých nám nevznikne cyklus. Dostaneme strom na nasledovnom obrázku



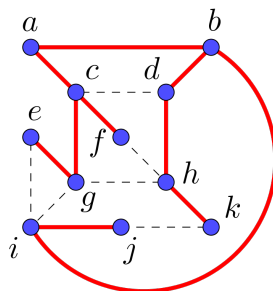
Týmto je zostrojená 1. úroveň koreňového stromu. Pribudli nám vrcholy b, c a 2 hrany medzi nimi a koreňom stromu. V druhom kroku prehľadáme vrcholy b a c v uvedenom poradí. Výsledok vidíme na ďalších obrázkoch



Týmto je zostrojená 2. úroveň koreňového stromu. Prehľadaním vrcholu b nám pribudli vrcholy d a i a prehľadaním vrcholu c nám pribudli vrcholy f a g . A samozrejme pribudli aj 4 zodpovedajúce hrany. V treťom kroku ideme prehľadať vrcholy d, f, g a i v uvedenom poradí. Dostávame



Týmto je zostrojená 3. úroveň koreňového stromu. Prehľadaním vrcholu d sme dostali vrchol h , prehľadaním vrcholu f nepribudol žiadny vrchol, prehľadaním vrcholu g pribudol vrchol e a prehľadaním vrcholu i pribudol vrchol j . Spolu s vrcholmi pribudli aj príslušné hrany. V poslednom štvrtom kroku prehľadáme vrcholy e, h a j v uvedenom poradí. Dostávame

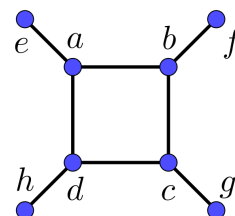


Týmto je zostrojená 4. úroveň koreňového stromu. Ako posledný pribudol vrchol k , prehľadáním vrcholu h a pribudla aj hrana medzi týmito dvoma vrcholmi. Koreňový strom už obsahuje všetky vrcholy daného grafu, takže je jeho kostrou. ■

Uvedieme ešte dva príklady prehľadávania grafu do šírky a okrem grafickej interpretácie si v nich zapíšeme postup prehľadávania grafu aj formou tabuľky.

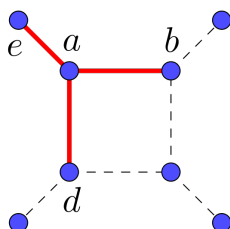
■ Príklad 9.3

Na obrázku vpravo je daný graf, ktorého vrcholy sú označené písmenami a až h . Nájdite jeho kostru prehľadávaním do šírky, pri lexikografickom usporiadaní vrcholov.



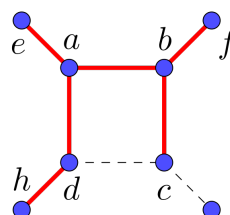
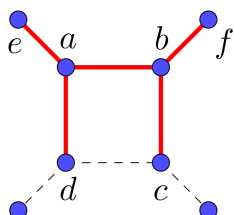
Riešenie: Tabuľkový zápis prehľadávania grafu bude mať 3 stĺpce. V prvom stĺpci tabuľky budú vrcholy, ktoré už patria do stromu S a ktorých incidentné hrany ideme prehľadávať. V druhom stĺpci tabuľky budú hrany, ktoré sú už súčasťou stromu S , čiže tie, ktoré sú na obrázku červené. A napokon, v treťom stĺpci tabuľky budú hrany, ktoré sú momentálne vo fronte a čakajú na spracovanie. Červenou farbou budú označené tie hrany vo fronte, ktorých pridaním do S nevznikne cyklus a budú preto do S pridané v nasledujúcom kroku.

Prehľadávanie začneme vrcholom a a v prvom kroku pridáme všetky s ním incidentné hrany a ich druhé koncové vrcholy, pri ktorých nám nevznikne cyklus. Dostaneme strom na nasledovnom obrázku



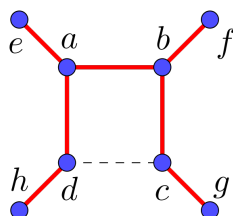
Krok	Prehľadávané vrcholy	Hrany v S	Obsah fronty
0.	a		$\{a,b\}, \{a,d\}, \{a,e\}$
1.	b,d,e	$\{a,b\}, \{a,d\}, \{a,e\}$	$\{b,a\}, \{b,c\}, \{b,f\}, \{d,a\}, \{d,c\}, \{d,h\}, \{e,a\}$

Týmto je zostrojená 1. úroveň koreňového stromu. Pribudli nám vrcholy b, d, e a hrany medzi nimi a vrcholom a . V druhom kroku prehľadáním týchto troch vrcholov dostávame



Krok	Prehľadávané vrcholy	Hrany v S	Obsah fronty
2.	c, f, h	$\{b, c\}, \{b, f\}, \{d, h\}$	$\{c, b\}, \{c, d\}, \{c, g\}, \{f, b\}, \{h, d\}$

Týmto je zostrojená 2. úroveň koreňového stromu. Pribudli nám vrcholy c, f, h a s nimi tri ďalšie hrany. V treťom kroku prehľadáme tieto tri vrcholy a dostaneme

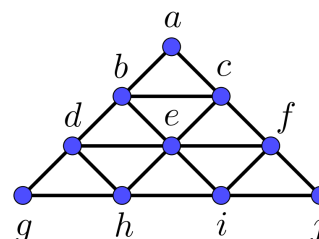


Krok	Prehľadávané vrcholy	Hrany v S	Obsah fronty
3.	g	$\{c, g\}$	

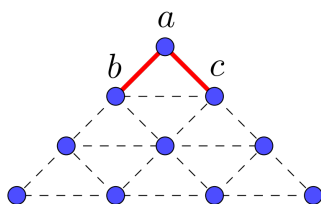
Týmto je zostrojená 3. úroveň koreňového stromu. V strome už máme všetky vrcholy pôvodného grafu, takže máme jeho kostru. ■

■ Príklad 9.4

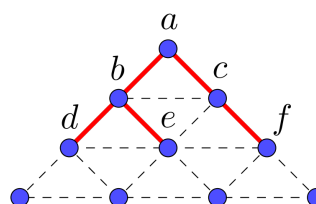
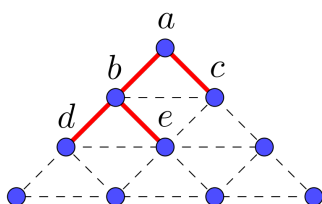
Na obrázku vpravo je daný graf, ktorého vrcholy sú označené písmenami a až j . Nájdite jeho kostru prehľadávaním do šírky pri lexikografickom usporiadaní vrcholov.



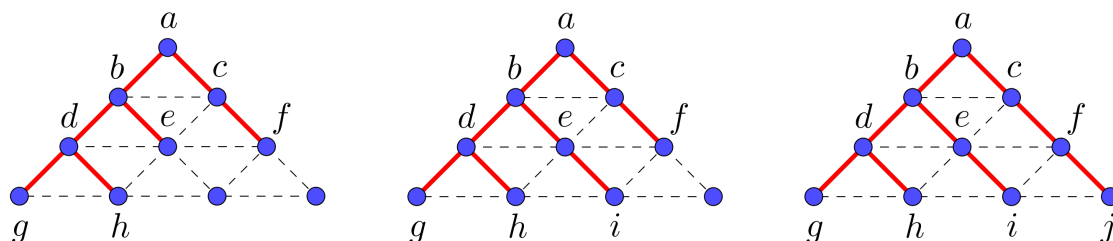
Riešenie: Úlohu najskôr vyriešime graficky a potom tabuľkou. Začneme vrcholom a a v prvom kroku pridáme všetky s ním incidujúce hrany a ich druhé koncové vrcholy, pri ktorých nám nevznikne cyklus. Dostaneme strom na nasledovnom obrázku



Týmto je zostrojená 1. úroveň koreňového stromu. Pribudli nám vrcholy b, c a hrany medzi nimi a vrcholom a . V druhom kroku prehľadáním týchto dvoch vrcholov dostávame



Týmto je zostrojená 2. úroveň koreňového stromu. Pribudli nám vrcholy d, e, f a s nimi tri ďalšie hrany. V treťom kroku prehľadáme tieto tri vrcholy a dostaneme



Týmto je zostrojená 3. úroveň koreňového stromu. V strome už máme všetky vrcholy pôvodného grafu, takže máme jeho kostru. Riešenie úlohy pomocou tabuľky vyzerá nasledovne.

Krok	Prehľadávané vrcholy	Hrany v S	Obsah fronty
0.	a		$\{a,b\}, \{a,c\}$
1.	b, c	$\{a,b\}, \{a,c\}$	$\{b,a\}, \{b,c\}, \{b,d\}, \{b,e\}, \{c,a\}, \{c,b\}, \{c,e\}, \{c,f\}$
2.	d, e, f	$\{b,d\}, \{b,e\}, \{c,f\}$	$\{d,b\}, \{d,e\}, \{d,g\}, \{d,h\}, \{e,b\}, \{e,c\}, \{e,d\}, \{e,f\}, \{e,h\}, \{e,i\}, \{f,c\}, \{f,e\}, \{f,i\}, \{f,j\}$
3.	g, h, i, j	$\{d,g\}, \{d,h\}, \{e,i\}, \{f,j\}$	

V ďalšej časti si ukážeme iný spôsob prehľadávania grafov.

9.4 Prehľadávanie grafu do hĺbky

Prehľadávanie grafu do hĺbky, po anglicky „*Depth-first search*“ (DFS), je založené na tom, že od koreňa stromu postupujeme v grafe tak hlboko, ako to je len možné. Keď sa dostaneme do vrcholu, z ktorého sa už nevieme dostať ďalej, tak sa vrátíme na najbližší predchádzajúci vrchol, z ktorého sa dá postupovať ďalej inou cestou atď., až pokiaľ nie sú prehľadané všetky vrcholy daného grafu. Spomenutý „krok späť“ sa nazýva *backtracking*.

Algoritmus prehľadávania grafu do hĺbky sa dá zapísať viacerými spôsobmi. Pseudokód algoritmu prehľadávania grafu do hĺbky, ktorý je uvedený na strane 187 hore, je úmyselne napísaný tak, aby bol takmer identický s pseudokódom prehľadávania grafu do šírky (strana 182). Jediný rozdiel medzi týmito dvoma algoritmami je v použitej dátovej štruktúre. Kým pri prehľadávaní grafu do šírky sme na ukladanie hrán používali frontu, pri prehľadávaní grafu do hĺbky budeme používať zásobník.

Prehľadávanie grafu do hĺbky si ilustrujeme na rovnakých štyroch príkladoch grafov, na akých sme si ilustrovali aj prehľadávanie grafu do šírky. Všimnime si rozdiel medzi postupom prehľadávania a výslednými kostrami grafu získanými jedným a druhým spôsobom prehľadávania.

Pri prehľadávaní grafu do hĺbky pribudne, na rozdiel od prehľadávania do šírky, do konštruovanej kostry v každom kroku vždy len jedna hrana. Okrem toho, ak chceme vrcholy grafu prehľadávať v lexikografickom usporiadaní tak, ako sme to robili v predošliých príkladoch, tak musíme hrany do zásobníka vkladať v opačnom lexikografickom usporiadaní. Toto poradie vkladania hrán do fronty, alebo zásobníka, však z hľadiska prehľadávania grafu nie je nijako podstatné. Je to len kozmetická záležitosť, ktorá slúži výlučne na to, aby výsledná kostra grafu bola jednoznačná. Bez určenia poradia prehľadávania vrcholov grafu totiž môžeme, oboma prehľadávaniami, dostať viacero rôznych kostier daného grafu.

Prehľadávanie grafu do hĺbky

VSTUP: Súvislý obyčajný graf $G = (V, E)$ a pevne zvolený vrchol $v_1 \in V$.

VÝSTUP: $S = (V', E')$ – kostra grafu G .

INICIALIZÁCIA

for { pre všetky hrany $h \in E$, kde $\text{zaciatok}(h) = v_1$ } **do**

$Z.\text{vloz}(h)$;

end

$E' := \{\}$;

$V' := \{v_1\}$;

VYTVÁRANIE KOSTRY S

while { $|V'| < |V|$ } **do**

$h := Z.\text{vyber}()$;

if { $h \notin E'$ a pridaním hrany h do E' nevznikne cyklus v S } **then**

$E' := E' \cup \{h\}$;

$V' := V' \cup \{\text{koniec}(h)\}$;

for { pre všetky hrany $h' \in E$, kde $\text{zaciatok}(h') = \text{koniec}(h)$ } **do**

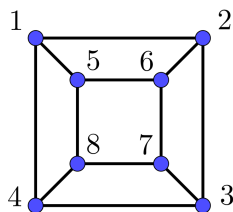
$Z.\text{vloz}(h')$;

end

end

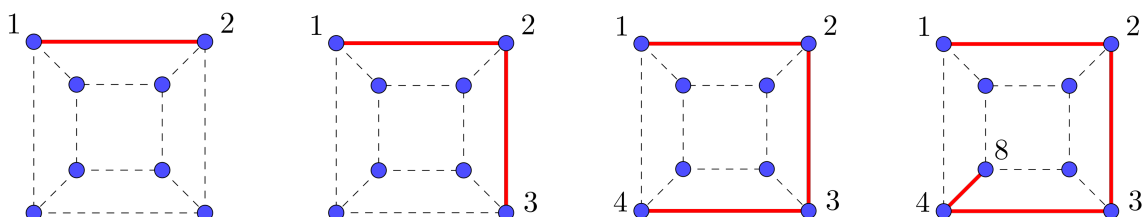
end

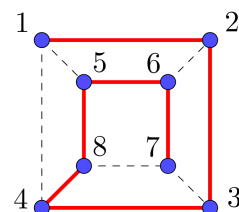
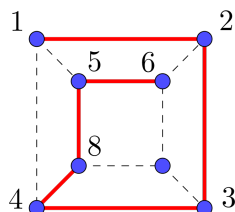
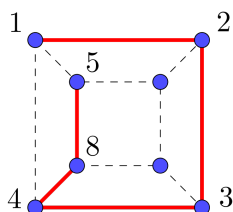
■ Príklad 9.5



Na obrázku vľavo je daný graf, ktorého vrcholy sú označené číslami 1 až 8. Nájdite jeho kostru prehľadávaním do hĺbky pri lexikografickom usporiadaní vrcholov.

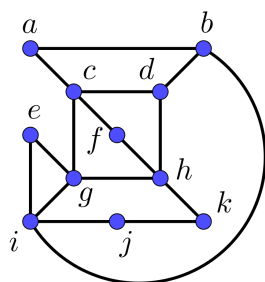
Riešenie: Začneme vrcholom 1, v prvom kroku pridáme vrchol 2, v druhom kroku pridáme vrchol 3, v treťom kroku pridáme vrchol 4, v štvrtom kroku pridáme vrchol 8, v piatom kroku pridáme vrchol 5, v šiestom kroku pridáme vrchol 6 a v poslednom, siedmom kroku, pridáme vrchol 7. V každom kroku samozrejme pridáme aj zodpovedajúcu hranu. Celý postup je zobrazený na nasledovnej sérii obrázkov





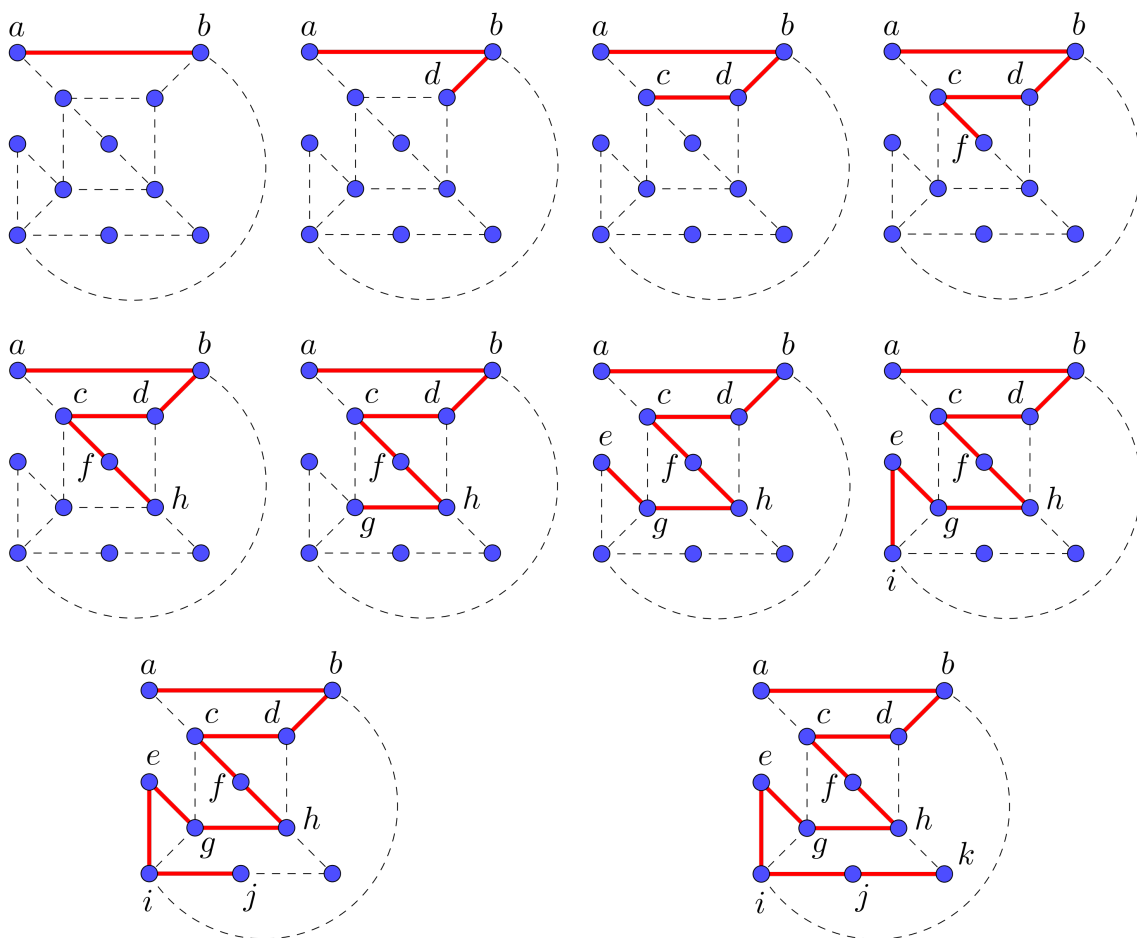
Ako vidíme z obrázkov, počas prehľadávania tohto grafu do hĺbky sme ani raz nemuseli použiť backtracking, t. j. spraviť „krok späť“. Z každého vrcholu sme vždy mohli pokračovať ďalej, až pokiaľ sme nedostali kostru grafu, ktorá má podobu cesty dĺžky 7. ■

■ Príklad 9.6



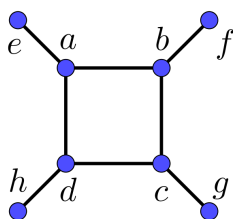
Na obrázku vľavo je daný graf, ktorého vrcholy sú označené písmenami *a* až *k*. Nájdite jeho kostru prehľadávaním do hĺbky pri lexikografickom usporiadaní vrcholov. ■

Riešenie: Celý postup riešenia je zobrazený na nasledovnej sérii desiatich obrázkov



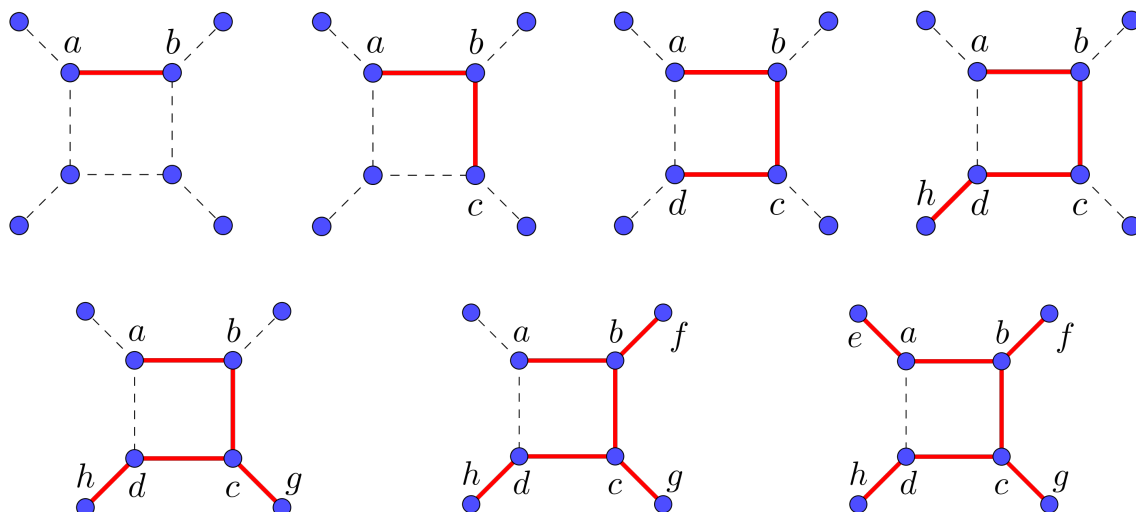
Pri ďalších dvoch príkladoch si okrem grafického riešenia zapíšeme riešenie aj pomocou tabuľky. ■

■ Príklad 9.7



Na obrázku vľavo je daný graf, ktorého vrcholy sú označené písmenami a až j . Nájdite jeho kostru prehľadávaním do hĺbky pri lexikografickom usporiadaní vrcholov.

Riešenie: prehľadávanie grafu do hĺbky znázorníme najskôr graficky na sérii obrázkov



V tomto príklade sme použili aj backtracking, a to dokonca trikrát. Najskôr sme sa z vrcholu h museli vrátiť späť až do vrcholu c , potom z vrcholu g sme sa museli vrátiť do vrcholu b a nakoniec sme sa z vrcholu f museli vrátiť do vrcholu a .

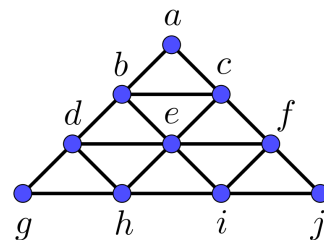
Teraz si riešenie znázorníme pomocou tabuľky. Toto bude podobné tomu, ktoré sme robili pri prehľadávaní grafu do šírky, v príkladoch 9.3 a 9.4. Tabuľkový zápis prehľadávania grafu bude mať 3 stĺpce. V prvom stĺpci tabuľky budú vrcholy, ktoré už patria do stromu S a ktorých incidentujúce hrany ideme prehľadávať. V druhom stĺpci tabuľky budú hrany, ktoré sú už súčasťou stromu S , čiže tie, ktoré sú na obrázku červené. A napokon, v treťom stĺpci tabuľky budú hrany, ktoré sú momentálne v zásobníku a čakajú na spracovanie. Červenou farbou bude v zásobníku vždy označená tá hrana, ktorá je najbližšia na rade, t. j. bola daná do zásobníka ako posledná.

Krok	Prehľadávaný vrchol	Hrany v S	Obsah zásobníka
0.	a		$\{a, e\}, \{a, d\}, \{a, b\}$
1.	b	$\{a, b\}$	$\{a, e\}, \{a, d\}, \{b, f\}, \{b, c\}, \{b, a\}$
2.	c	$\{b, c\}$	$\{a, e\}, \{a, d\}, \{b, f\}, \{c, g\}, \{c, d\}, \{c, b\}$
3.	d	$\{c, d\}$	$\{a, e\}, \{a, d\}, \{b, f\}, \{c, g\}, \{d, h\}, \{d, c\}, \{d, a\}$
4.	h	$\{d, h\}$	$\{a, e\}, \{a, d\}, \{b, f\}, \{c, g\}, \{h, d\}$
5.	g	$\{c, g\}$	$\{a, e\}, \{a, d\}, \{b, f\}, \{g, c\}$
6.	f	$\{b, f\}$	$\{a, e\}, \{a, d\}, \{f, b\}$
7.	e	$\{a, e\}$	$\{e, a\}$

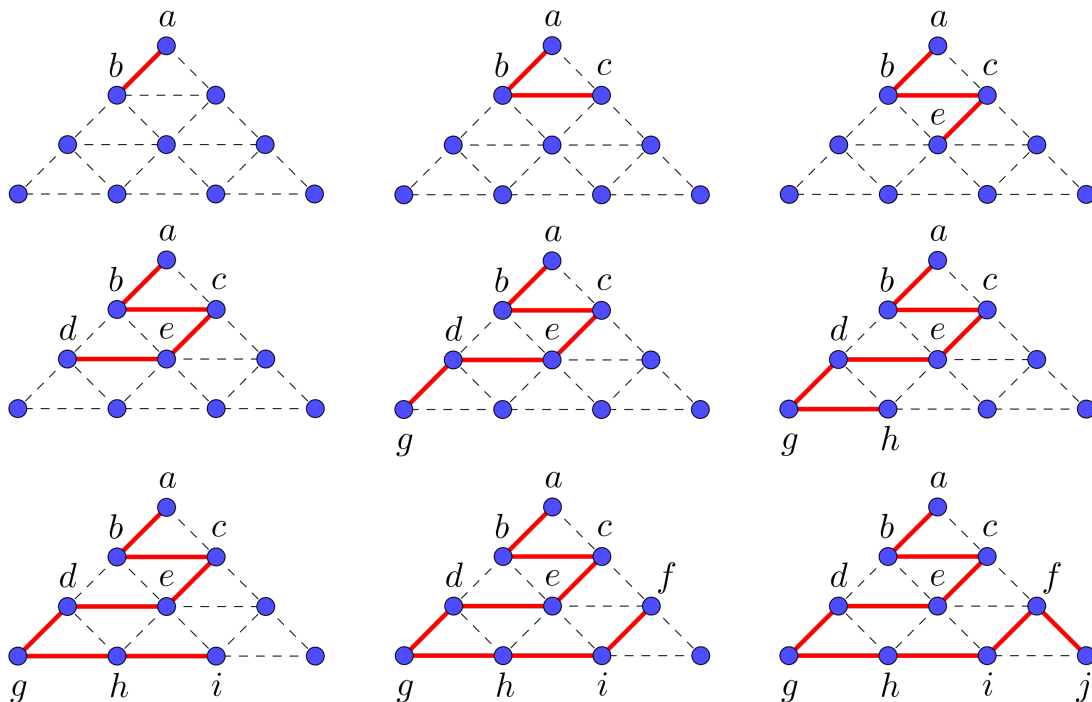
■

■ Príklad 9.8

Na obrázku vpravo je daný graf, ktorého vrcholy sú označené písmenami a až j . Nájdite jeho kostru prehľadávaním do hĺbky pri lexikografickom usporiadaní vrcholov.



Riešenie: prehľadávanie grafu do hĺbky si opäť znázorníme graficky na sérii obrázkov



Pri tomto grafe sme ani raz nemuseli použiť backtracking a nájdená kostra je cesta dĺžky 9. Teraz si priebeh prehľadávania grafu do hĺbky zapíšeme tabuľkou.

Krok	Prehľadávaný vrchol	Hrany v S	Obsah zásobníka
0.	a		$\{a, c\}, \{a, b\}$
1.	b	$\{a, b\}$	$\{a, c\}, \{b, e\}, \{b, d\}, \{b, c\}, \{b, a\}$
2.	c	$\{b, c\}$	$\{a, c\}, \{b, e\}, \{b, d\}, \{c, f\}, \{c, e\}, \{c, b\}, \{c, a\}$
3.	e	$\{c, e\}$	$\{a, c\}, \{b, e\}, \{b, d\}, \{c, f\}, \{e, i\}, \{e, h\}, \{e, f\}, \{e, d\}, \{e, c\}, \{e, b\}$
4.	d	$\{e, d\}$	$\{a, c\}, \{b, e\}, \{b, d\}, \{c, f\}, \{e, i\}, \{e, h\}, \{e, f\}, \{d, h\}, \{d, g\}, \{d, e\}, \{d, b\}$
5.	g	$\{d, g\}$	$\{a, c\}, \{b, e\}, \{b, d\}, \{c, f\}, \{e, i\}, \{e, h\}, \{e, f\}, \{d, h\}, \{g, h\}, \{g, d\}$
6.	h	$\{g, h\}$	$\{a, c\}, \{b, e\}, \{b, d\}, \{c, f\}, \{e, i\}, \{e, h\}, \{e, f\}, \{d, h\}, \{h, i\}, \{h, g\}, \{h, e\}, \{h, d\}$
7.	i	$\{h, i\}$	$\{a, c\}, \{b, e\}, \{b, d\}, \{c, f\}, \{e, i\}, \{e, h\}, \{e, f\}, \{d, h\}, \{i, j\}, \{i, h\}, \{i, f\}, \{i, e\}$
8.	f	$\{i, f\}$	$\{a, c\}, \{b, e\}, \{b, d\}, \{c, f\}, \{e, i\}, \{e, h\}, \{e, f\}, \{d, h\}, \{i, j\}, \{i, h\}, \{f, j\}, \{f, i\}, \{f, e\}, \{f, c\}$
9.	j	$\{f, j\}$	$\{a, c\}, \{b, e\}, \{b, d\}, \{c, f\}, \{e, i\}, \{e, h\}, \{e, f\}, \{d, h\}, \{i, j\}, \{i, h\}, \{j, i\}, \{j, f\}$

Všimnime si, že algoritmus skončil a v zásobníku ešte zostali hrany. Na tomto príklade vidíme, že pri popísanom algoritme sa nemusí zásobník vždy vyprázdniť tak, ako sa to stalo pri predošlom príklade. Závisí to od daného grafu. ■

9.5 Dôkaz korektnosti algoritmov prehľadávania do šírky a do hĺbky

Ako sme mohli vidieť v tabuľkových zápisoch algoritmov prehľadávania grafov do šírky (podkapitola 9.3) aj do hĺbky (podkapitola 9.4), nami použité algoritmy vôbec nie sú optimálne. Tejto skutočnosti sme si plne vedomí a uvedené pseudokódy algoritmov neboli písané s cieľom dosiahnuť optimálnu rýchlosť algoritmu. Naším cieľom pri písaní pseudokódov prehľadávacích algoritmov bolo napísať oba pseudokódy tak, aby

1. boli až na použitú dátovú štruktúru identické,
2. boli popisované algoritmy čo najzrozumiteľnejšie a čo najstručnejšie zapísané,
3. sa dal dôkaz ich korektnosti spraviť čo najjednoduchšie.

V nasledujúcej vete dokážeme, že algoritmy prehľadávania grafu do šírky aj do hĺbky sú korektné.

Veta 9.5.1 — O korektnosti algoritmov prehľadávania grafu do šírky a do hĺbky. Uvedené algoritmy prehľadávania grafu do šírky a do hĺbky sú korektné, čiže po ich skončení bude strom S kostrou daného grafu G .

Dôkaz: Oba algoritmy (str. 182 a 187) sú, až na použitú dátovú štruktúru (fronta/zásobník), identické. Preto dôkazy korektnosti oboch algoritmov budú rovnaké a budeme písať len o „prehľadávaní“ grafu, bez rozlišovania „do šírky“ a „do hĺbky“.

Algoritmom prehľadávania grafu postupne vzniká strom. V každom kroku pridávame k už existujúcemu stromu vrcholy a hrany tak, aby nevznikol cyklus. Okrem toho je z algoritmu zrejmé, že pre každý pridávaný vrchol existuje cesta od tohto vrcholu ku koreňu stromu. Preto medzi ľubovoľnými dvoma vrcholmi konštruovaného grafu existuje cesta. Takže konštruovaný graf je acyklický a súvislý, teda strom. Treba už len ukázať, že po skončení algoritmu strom S obsahuje všetky vrcholy daného grafu G , t. j. je jeho kostrou. To dokážeme sporom.

Algoritmus sa riadne ukončí vtedy, keď nastane situácia $|V'| = |V|$, čo je podmienka v cykle **while**. Ak je podmienka $|V'| = |V|$ splnená, tak strom S obsahuje všetky vrcholy daného grafu G , a preto je jeho kostrou. Problém môže nastať len v prípade, ak by algoritmus prehľadávania skončil predčasne. Takéto predčasné skončenie by mohlo byť spôsobené len na dvoch miestach algoritmu. Prvý možný dôvod predčasného ukončenia je ten, že fronta/zásobník sú už prázdne a zlyhá priradenie hrany $h := F.vyber()$, resp. $h := Z.vyber()$. Druhý možný dôvod predčasného ukončenia **while** cyklu je test **if**, kedy by sme už nedokázali pridať žiadnu ďalšiu hranu tak, aby nevznikol cyklus.

Takže predpokladajme, že skonštruovaný strom S nie je kostrou. Potom existuje nejaký vrchol x , ktorý nie je súčasťou vzniknutého stromu S . Daný graf G bol ale súvislý, takže v ňom musí existovať cesta medzi vrcholom x a koreňom stromu v_1 . Označme si túto cestu ($x = u_1, u_2, \dots, u_{i-1}, u_i, \dots, u_k = v_1$). Nech u_i je prvý vrchol na tejto ceste, ktorý je obsiahnutý aj v strome S . Je zrejmé, že $1 < i \leq k$. Keďže vrchol u_i je v strome S , musela byť v priebehu algoritmu aj hrana $\{u_{i-1}, u_i\}$ vložená do fronty/zásobníka, a teda musela byť v niektorom ďalšom kroku preskúmaná. Pridaním vrcholu u_{i-1} a hrany $\{u_{i-1}, u_i\}$ ku stromu S nemôže vzniknúť cyklus, a teda aj vrchol u_{i-1} musí byť súčasťou stromu S . To je však spor s tým, že vrchol u_i bol prvý vrchol na ceste ($x = u_1, u_2, \dots, u_{i-1}, u_i, \dots, u_k = v_1$), ktorý bol súčasťou stromu S . Preto všetky vrcholy pôvodného grafu G sú súčasťou S a strom S je kostra grafu G . Q.E.D.

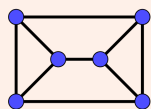
Každý, kto si beh algoritmov uvedených na stranách 182 a 187 skúsi simulovať na papieri, veľmi rýchlo príde na to, ako by sa dali uvedené algoritmy zoptimalizovať. Na túto tému sú aj cvičenia 9.10 a 9.11. Samozrejme, že pri riešení cvičení sa nemusí používať ťažkopádna forma algoritmov podľa uvedených pseudokódov a ich tabuľkový zápis prehľadávania použitý v príkladoch 9.3, 9.4, 9.7 a 9.8. My sme sa v týchto príkladoch otrocky pridržiavali v texte uvedených algoritmov. Tabuľkové zápisy oboch algoritmov sa malými modifikáciami pseudokódov dajú skrátiť, pri prehľadávaní do hĺbky dokonca výrazne skrátiť.

9.6 Cvičenia

Cvičenie 9.1 Pomocou prehľadávacích algoritmov dokažte, že všetky kostry daného grafu majú rovnaký počet hrán. ■

Cvičenie 9.2 Koľko navzájom neizomorfných kostier má graf K_6 ? ■

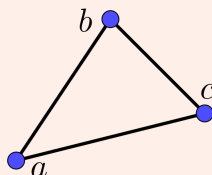
Cvičenie 9.3



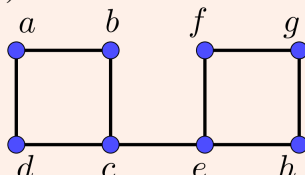
Nájdite aspoň tri rôzne kostry grafu zobrazenom na obrázku vľavo prehľadávaním do šírky a tri rôzne kostry prehľadávaním do hĺbky. ■

Cvičenie 9.4 Nájdite všetky kostry nasledujúcich grafov

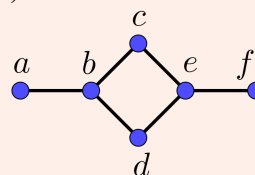
(a)



(b)



(c)



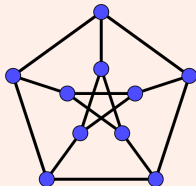
Cvičenie 9.5 Nech S je kostra grafu G . Dokážte, že ak do S pridáme ľubovoľnú hranu pri nezmenenej množine vrcholov, tak vznikne cyklus. ■

Cvičenie 9.6 Prehľadávaním do šírky aj do hĺbky nájdite kostru grafu Q_3 . Definíciu grafu Q_n nájdete v cvičení 5.21 (strana 118). ■

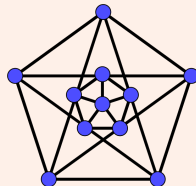
Cvičenie 9.7 Prehľadávaním do šírky aj do hĺbky nájdite kostru grafu $K_{3,7}$. ■

Cvičenie 9.8 Nájdite kostry nasledujúcich grafov, ich prehľadaním do šírky aj do hĺbky, vychádzajúc z toho istého vrcholu. ■

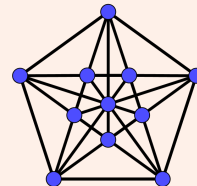
(a)



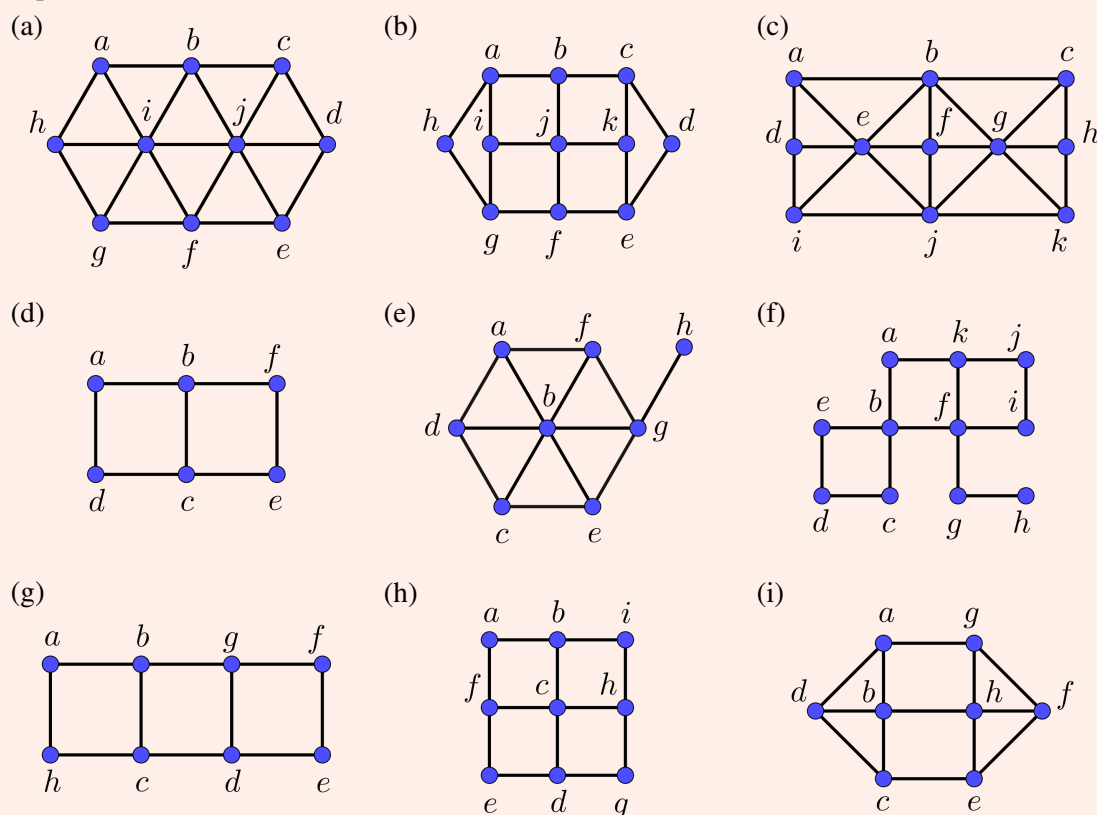
(b)



(c)



Cvičenie 9.9 Nájdite kostry nasledujúcich grafov, ich prehľadáním do šírky aj do hĺbky, vychádzajúc z toho istého vrcholu. V oboch prípadoch prehľadávajújte vrcholy grafov v lexikografickom usporiadaní.



Cvičenie 9.10 Modifikujte algoritmus na prehľadávanie grafu do šírky (strana 182) tak, aby v ňom nebolo treba použiť podmienku **if** a jediným testom v celom algoritme zostal len test v cykle **while**. ■

Cvičenie 9.11 Dá sa modifikovať algoritmus na prehľadávanie grafu do hĺbky (strana 187) tak, aby v ňom nebolo treba použiť podmienku **if** a jediným testom v celom algoritme zostal len test v cykle **while**? ■

Cvičenie 9.12 Prečo pridaním vrcholu u_{i-1} a hrany $\{u_{i-1}, u_i\}$, v dôkaze vety 9.5.1 (str. 191), do stromu S , nemôže vzniknúť cyklus? ■

Cvičenie 9.13 Naprogramujte prehľadávanie daného grafu do šírky v Pythone alebo v C.

Cvičenie 9.14 Naprogramujte prehľadávanie daného grafu do hĺbky v Pythone alebo v C.

Cvičenie 9.15 Modifikujte algoritmus na prehľadávanie grafu do šírky (strana 182) tak, aby sa do fronty ukladali vrcholy, nie hrany. (Pomôcka: každý vrchol ukladajú do fronty dostane značku identifikujúcu vrchol, z ktorého sme sa doň dostali.) ■

Cvičenie 9.16 Modifikujte algoritmus na prehľadávanie grafu do hĺbky (strana 187) tak, aby sa do zásobníka ukladali vrcholy, nie hrany. (Pomôcka: každý vrchol ukladaný do zásobníka dostane značku identifikujúcu vrchol, z ktorého sme sa doň dostali.) ■

Cvičenie 9.17 Modifikujte algoritmy na prehľadávanie grafu do šírky aj hĺbky (strany 182 a 187) tak, aby sme zistili, či je vstupný graf súvislý (v algoritmoch uvedených v skriptách je to ako predpoklad). ■

Cvičenie 9.18 Modifikujte algoritmus na prehľadávanie grafu do šírky (strana 182) tak, aby sme pomocou neho zistili vzdialenosť každého vrcholu v danom grafe od vrcholu v_1 , z ktorého sa začína prehľadávanie. ■

Cvičenie 9.19 Ukážte, že pri modifikovanom algoritme prehľadávania do hĺbky z cvičenia 9.16, sa v zásobníku nachádzajú len vrcholy tvoriace cestu medzi práve vyšetrovaným vrcholom po vrchol v_1 , z ktorého sa začína prehľadávanie. ■

Cvičenie 9.20 Ukážte, že každá hrana grafu G spája iba vrcholy, ktoré sú v strome získanom prehľadávaním do hĺbky vo vzťahu predok–potomok. ■



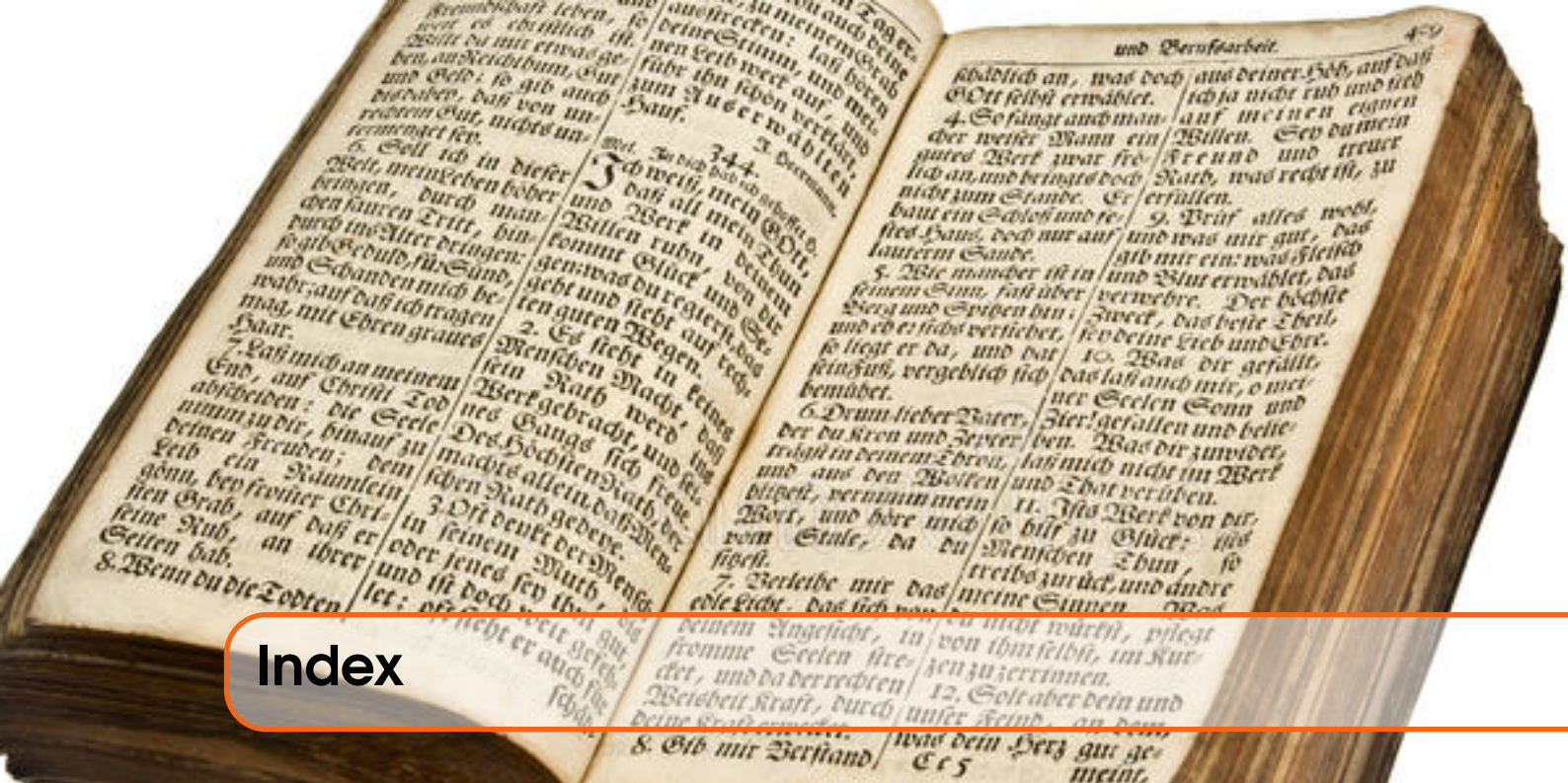
Použitá a odporúčaná literatúra

- [1] Martin Aigner: **Combinatorial Theory**, SPRINGER VERLAG, HEIDELBERG 1979.
- [2] Bohuslav Balcar, Petr Štěpánek: **Teorie množin**, ACADEMIA, PRAHA 1986.
- [3] Berežný, Draženská, Kravcová: **Zbierka úloh z diskkrétnej matematiky**, FEI TU, KOŠICE 2005. <http://web.tuke.sk/fei-km/sites/default/files/prilohy/10/Zbierka-DM.pdf>
- [4] J. A. Bondy, U. S. Murty: **Graph Theory With Applications**, NORTH-HOLLAND 1976.
- [5] Reinhard Diestel: **Graph Theory**, SPRINGER 2000.
- [6] Harary, Frank: **Graph theory**, ADDISON-WESLEY 1969.
- [7] Richard Johnsonbaugh: **Discrete Mathematics**, PEARSON 2017.
- [8] Josef Kaucký: **Kombinatorické identity**, VEDA, VYDAVATELSTVO SAV, BRATISLAVA 1975.
- [9] Martin Knor: **Úvod do matematickej logiky**, FIIT STU, BRATISLAVA 2016.
<http://www.math.sk/jmkollar/literatura/Knor-Uvod.do.matematickej.logiky.pdf>
- [10] Martin Knor: **Teória grafov**, SVF STU, BRATISLAVA 2008.
<http://www.svf.stuba.sk/docs/dokumenty/skripta/teoria-grafov.martin.knor.pdf>
- [11] Martin Knor, Jozef Kollár: **Matematika pre architektov**, STU BRATISLAVA, 2002.
- [12] Donald E. Knuth: **The Art of Computer Programming, Volume 1**, ADDISON-WESLEY, 1997.
- [13] Sergei K. Lando: **Lectures on Generating Functions**, AMERICAN MATHEMATICAL SOCIETY, STUDENT MATHEMATICAL LIBRARY 2003.
- [14] Milan Mareš: **Příběhy matematiky**, PISTORIUS & OLŠANSKÁ, PŘÍBRAM 2011.
- [15] J. Matoušek, J. Nešetřil: **Kapitoly z diskrétní matematiky**, KAROLINUM, PRAHA 2002.

- [16] М. В. Меньшиков, А. М. Ревякин, А. Н. Копылова, Ю. Н. Макаров, Б. С. Стечкин: **Комбинаторный Анализ – Задачи и упражнения**, ИЗДАТЕЛЬСТВО »НАУКА«, МОСКВА 1982.
- [17] В. А. Носов: **Комбинаторика и теория графов**, МОСКВА 1999.
<http://intsys.msu.ru/staff/vnosov/combgraph.htm> (PDF verzia)
- [18] Stanislav Palúch: **Algoritmická teória grafov**, ŽILINSKÁ UNIVERZITA, 2008.
<http://frcatel.fri.uniza.sk/users/paluch/grafy.pdf>
- [19] Ján Plesník: **Grafové algoritmy**, VEDA, BRATISLAVA 1983.
- [20] Franco P. Preparata, Raymond T. Yeh: **Úvod do teórie diskretných matematických štruktúr**, ALFA, BRATISLAVA 1982.
- [21] Edward M. Reingold, Jurg Nievergelt, Narsingh Deo: **Combinatorial Algorithms – Theory and Practice**, PRENTICE-HALL, NEW JERSEY 1977.
- [22] Beloslav Riečan a kol.: **Úlohy z matematiky pre 4. ročník gymnázia**, SPN, BRATISLAVA 1976.
- [23] Beloslav Riečan a kol.: **Matematika pre 4. ročník gymnázia**, SPN, BRATISLAVA 1987.
- [24] Beloslav Riečan a kol.: **Zbierka úloh z matematiky pre 4. ročník gymnázia**, SPN, BRATISLAVA 1991.
- [25] Karol Rován a kol.: **Zbierka riešených úloh z algebry pre SVŠ a odborné školy**, SPN, BRATISLAVA 1969.
- [26] К. А. Рыбников: **Введение в Комбинаторный Анализ**, ИЗДАТЕЛЬСТВО МОСКОВСКОГО УНИВЕРСИТЕТА, МОСКВА 1985.
- [27] Raymond M. Smullyan: **A Beginner's Guide to Mathematical Logic**, DOVER PUBLICATIONS, NEW YORK 2014.
- [28] František Vejsada, František Talafous: **Zbierka úloh z matematiky pre SVŠ a gymnázia**, SPN, BRATISLAVA 1973.
- [29] Naum J. Vilenkin: **Rozhovory o množinách**, SPN, BRATISLAVA 1972.
- [30] Herbert S. Wilf: **generatingfunctionology**, ACADEMIC PRESS, INC. 1994.
<https://www.math.upenn.edu/~wilf/DownldGF.html>
- [31] Niklaus Wirth: **Algoritmy a štruktúry údajov**, ALFA, BRATISLAVA 1989.
- [32] Štefan Znáť: **Kombinatorika a teória grafov**, PF UK, BRATISLAVA 1978.

IV

Index



Index

Symboly

∈ operátor „patrí do množiny“	11
∉ operátor „nepatří do množiny“	11
⊂ operátor vlastnej podmnožiny	12
⊆ operátor nevlastnej podmnožiny	12
⊔ operátor zjednotenia množín	12
∩ operátor prieniku množín	13
∖ operátor rozdielu množín	13
Ā komplement množiny A	14
¬ operátor logickej negácie	20
∨ operátor disjunkcie	20
∧ operátor konjunkcie	20
⇔ operátor ekvivalencie	20
⇒ operátor implikácie	20
∃ existenčný kvantifikátor	23
∀ všeobecný kvantifikátor	23
operátor „a delí b“ (a b)	40
Δ operátor symetrickej diferencie	41

A

acyklický graf	62
algoritmus	
Borůvkov	203
Dijkstrov	229–231
hľadania kostry grafy	179
Jarníkov (Primov)	197
konštrukcie očíslovaného stromu z Prüfe- rovho kódu	241

B

konštrukcie optimálneho Huffmanovho kódu	206
konštrukcie Prüferovho kódu z očíslova- ného stromu	235
konštrukcie prehľadávacieho binárneho stromu	215
Kruskalov	201, 202
pažravý	201
prehľadávania grafu do hĺbky	187
prehľadávania grafu do šírky	182
zostavovania identifikačného kódu binár- neho stromu	219
zostavovania identifikačného kódu kore- ňového stromu	222
antisymetrickosť (relácie)	125, 130
Appel Kenneth (1932–2013)	48
ASCII kód	204
atomická formula	20
axióma	31

backtracking	186
bijektivnosť	145
binárny strom	80
binomický koeficient	171
Borůvka Otakar (1899–1995)	196, 203
Bouton Charles Leonard (1869–1922)	224

C

Catalanovo číslo 174, 213
 Cayley Arthur (1821–1895) 16, 48, 241
 centrálny vrchol grafu 56
 centrum grafu 56
 cesta (v grafe) 55, 112
 cyklus (v grafe) 57, 112

Č

číslo

Catalanovo 174, 213
 celé $\mathbb{Z}$ 15
 iracionálne 15
 komplexné $\mathbb{C}$ 16
 kvaternion 16
 oktonion 16
 prirodzené $\mathbb{N}$ 14
 racionálne $\mathbb{Q}$ 15
 reálne $\mathbb{R}$ 15

D

dátová štruktúra

fronta 180
 zásobník 180

dôkaz

matematickou indukciou 37, 39
 nepriamy 35
 priamy 32
 sporom 36

de Moivre Abraham (1667–1754) 155

de Morgan Augustus (1806–1871) 48

definícia 32

Dijkstra Edsger Wybe (1930–2002) . 197, 230

Dijkstrov algoritmus 229, 231

disjunkcia ($\vee$) 21

disjunktné množiny 13

E

ekvivalencia ($\Leftrightarrow$) 22

ekvivalencia (relácie) 140

entropický kód 204

Euklides Alexandrijský (~3. stor. pred n. l.) 15

Euklidovský priestor 108

Euler Leonhard (1707–1783) 47, 84, 109, 156

F

faktor grafu 62

faktoriál 39

dvojný 172

Fano Roberto Mario (1917–2016) 204

Fibonacci Leonardo (~1170 – ~1240) ... 167

Fibonacciho postupnosť 167

formula

atomatická 20

prvotná 20

výroková 20, 22

formuly

ekvivalentné 22

fronta 180

funkcia 143

bijektívna 145

charakteristická funkcia množiny 149

definičný obor 143

injektívna 144

inverzná 145

koobor 143

obor hodnôt 143

parciálna 143

surjektívna 144

vytvárajúca

exponenciálna, 175

obyčajná, 160

zložená 146

G

Gauss Carl Friedrich (1777–1855) 38

geometrická postupnosť

súčet prvých n členov 39

Gonthier Georges 48

graf

acyklický 62

artikulácia 61

bipartitný 53

kompletný ($K_{m,n}$), 54, 112

centrum 56

cesta 55, 112

cesta (P_n) 112

cyklus 57, 112

hamiltonovský, 95

cyklus (C_n) 112

digraf 50

dĺžka ťahu 55

dĺžka cesty 55

graf

dĺžka cyklu	57
dĺžka sledu	54
dokonalé párovanie	89
Eulerova veta o rovinných grafoch	110
eulerovský	87
eulerovský ťah	85
otvorený, 85, 87	
uzavretý, 85, 86	
excentricita vrcholu	56
faktor	62
hodnota	196
hodnota hrany	195
hodnota sledu	229
homeomorfné grafy	112
hrana	49
ohodnotená, 195	
hranica oblasti	108
húsenica	113
hviezda (S_n)	113
incidencia (vrchol–hrana)	50
invariant	101
izomorfizmus	
binárnych stromov, 106	
koreňových stromov, 105	
izomorfizmus grafov	98, 99
izomorfné grafy	99
jadro	225
jednoduchý (obyčajný)	51
Knight graph	98
koleso (W_n)	113
komplementárny	53
kompletný (K_n)	52, 112
kompletný bipartitný ($K_{m,n}$)	54, 112
komponent súvislosti	57, 60
konečný	49
koreňový strom	77
dieťa vrcholu, 78	
koncový vrchol, 78	
list, 78	
otec vrcholu, 78	
podstrom, 78	
potomok vrcholu, 78	
predkovia vrcholu, 78	
súrodenci (vrcholy), 78	
úroveň vrcholu, 77	
vnútorný vrchol, 78	
výška, 77	

graf

kostra	62
minimálna, 196	
Kuratowského veta	112
ľavý podstrom	213
les	77
list	77
matica incidencie	72
matica susednosti	69
minimálna kostra	196
množina vrcholov	
nezávislá, 225	
stabilná, 225	
most	61
najlacnejšia cesta	230
najlacnejšia kostra	196
násobné hrany	50
nekonečný	49
neorientovaný	49
nesúvislý	57
nezávislá množina vrcholov	225
oblasť	108
obyčajný (jednoduchý)	51
ohodnotený	195
ohodnotenie hrany	195
orientovaná hrana	49, 50
orientovaný	49, 50
otvorený sled	55
párovanie	89
maximálne, 89	
perfektné párovanie	89
podgraf	58
polomer	56
pravý podstrom	213
pravidelný	52
prázdny	49, 80
prehľadávanie do šírky	182
prehľadávanie do hĺbky	187
priemer	56
problém obchodného cestujúceho	234
rovinné nakreslenie	108
rovinný	108
sled	54
otvorený, 55	
uzavretý, 55	
slučka (hrana)	50
stabilná množina vrcholov	225

graf
 strom 73
 binárny, 80
 binárny úplný, 81
 binárny dokonalý, 82
 binárny prehľadavací, 214
 binárny vyvážený, 81
 koreň, 77, 105
 očíslovaný, 234
 prehľadavací binárny, 213
 stupeň vrcholu 51
 subdivízia hrany 111
 susedné oblasti 108
 susednosť (vrchol–vrchol) 50
 súvislý 57
 ťah 55
 uzavretý sled 55
 vrchol 49, 50
 centrálny, 56
 izolovaný, 50
 stupeň, 51
 vzdialenosť vrcholov 56
 Graves John Thomas (1806–1870) 16
 Guthrie Francis (1831–1899) 48

H

Haken Wolfgang (1928–...) 48
 Hamilton William R. (1805–1865) . 16, 48, 95
 harmonická postupnosť 166
 Herón Alexandrijský (10–75) 15
 Hippiasos (~530–~450 pred n. l.) 15
 hodnota grafu 196
 hrana
 koncový vrchol 181
 počiatočný vrchol 181
 Huffman David Albert (1925–1999) 204
 Huffmanov kód 80, 204, 206
 húsenica (graf) 113
 hviezda (graf) 113
 hypotéza 32

I

implikácia ($\Rightarrow$) 21
 indukčný
 krok 38
 predpoklad 38
 injektívnosť 144
 inverzná funkcia 145

inverzná relácia 127

J

jadro grafu 225
 Jarník Vojtěch (1897–1970) 196, 197
 Jarníkov (Primov) algoritmus 197
 jazyk 1. rádu 23
 jednotka
 imaginárna 16
 kvaternionová 16
 oktonionová 16

K

kartézsky súčin 19
 Kempe Alfred Bray (1849–1922) 48
 kód
 ASCII 204
 Huffmanov 80, 204, 206
 Prüferov 234
 koeficient
 binomický 171
 Newtonov, 171
 multinomický 177
 koleso (graf) 113
 komplementárna množina 14
 komponent súvislosti (grafu) 60
 konjunkcia ($\wedge$) 21
 kontradikcia 20
 koreň 77
 koreňový strom 77
 kostra grafu 62
 Kőnig Dénes (1884–1944) 48
 Kruskal Joseph B. (1928–2010) 196, 201
 Kruskalov algoritmus 202
 Kuratowski Kazimierz (1896–1980) 111
 kvantifikátor
 existenčný 23
 všeobecný 23

L

Laplace Pierre-Simon (1749–1827) 156
 lema 62
 les 77
 list 77
 logická
 hodnota 19
 premenná 19

logická	
spojka	19, 20
logický	
operátor	19
výrok	19
logika	
predikátová	23
výroková	22

M

matematická indukcia	37, 39
matica	
incidencie grafu	72
relácie	128
susednosti grafu	69
maximálne párovanie grafu	89
množina	11
charakteristická funkcia	149
kartézsky súčin množín	19
komplementárna	14
konečná	12
mohutnosť	12
nekonečná	12
podmnožina	12
potenčná	13
prázdna	11
rozklad	138
spočítateľná	37
systém množín	138
univerzálna	14
univerzum	13, 14
množiny	
disjunktné	13
priemik množín	13
rovnosť množín	11
rozdiel množín	13
symetrická diferencia množín	41
zjednotenie množín	12
mocninový rad	156
modulo (zvyšok po delení)	41
modulo $a \pmod{b}$	41
modus ponens	32
mohutnosť množiny	12
Moskovský papyrus	16
multinomická veta	177
multinomický koeficient	177

N

negácia ($\neg$)	20
Newton Isaac (1642–1727)	171
Newtonov binomický koeficient	171

P

P-kód (prefixový kód)	204
párovanie grafu	89
perfektné párovanie	89
podgraf grafu	58
podmnožina	
vlastná	12
postupnosť	
Fibonacciho	167
harmonická	166
potenčná množina	13
pravdivostná hodnota	19
pravdivostná tabuľka	20
prázdna množina	11
prázdny graf	80
prefixový kód	204
prehľadavací binárny strom	213, 214
prehľadávanie grafu	
do hĺbky	186, 187
do šírky	181, 182
prienik množín	13
Prim Robert Clay (1921–...)	197
princíp zapojenia-vypojenia	18
problém	
NP-úplný	96
obchodného cestujúceho	47, 234
štyroch farieb	48
Prüfer Ernst Paul Heinz (1896–1934)	241
Prüferov kód	234
prvky	
porovnateľné	137
neporovnateľné	137
prvotná formula	20
pseudokód	
Dijkstrovho algoritmu	231
hľadania kostry grafu	179
Jarníkovho algoritmu	197
konštrukcie očíslovaného stromu z Prüferovho kódu	241
konštrukcie optimálneho Huffmanovho kódu	206
konštrukcie prehľadávacieho binárneho stromu	215

konštrukcie Prüferovho kódu z očíslovaného stromu 235
 Kruskalovho algoritmu 202
 prehľadávania grafu do hĺbky 187
 prehľadávania grafu do šírky 182
 zostavovania identifikačného kódu binárneho stromu 219
 zostavovania identifikačného kódu koreňového stromu 222
 pytagorejci 15

R

rad
 formálny mocninový 156
 Maclaurinov 156
 mocninový 156
 Taylorov 156
 reflexívnosť (relácie) 124, 129
 relácia
 n -árna 123
 antisymetrická 125, 130
 binárna 123, 124
 čiasťočné usporiadanie 137
 ekvivalencie 140
 inverzná 127
 maticová reprezentácia 128
 na množine 124
 reflexívna 124, 129
 symetrická 125, 129
 tranzitívna 126
 zložená 128
 rovnosť množín 11
 rozdiel množín 13
 rozklad množiny 138
 triedy ekvivalencie 141
 triedy rozkladu 138

S

semifaktoriál 172
 skladanie funkcií 146
 skladanie relácií 128
 sled (v grafe) 54
 slučka (graf) 50
 spočítateľná množina 37
 Stirling James (1692–1770) 156
 strom (graf) 73
 vyvážený binárny 81
 subdivízia hrany 111

súčín
 kartézsky 19
 surjektívnosť 144
 sylogizmus 34
 symetrická diferencia množín (Δ) 41
 symetrickosť (relácie) 125, 129
 systém množín 138

T

tabuľka
 pravdivostná 20
 Tait Peter Guthrie (1831–1901) 48
 tautológia 20
 tautologicky ekvivalentné formuly 22
 tranzitívnosť (relácie) 126
 trieda
 rozkladu 138
 zvyšková 142
 trieda ekvivalencie 141

Ť

ťah (v grafe) 55

U

univerzum množín 13, 14
 usporiadaná dvojica 18
 usporiadanie
 čiasťočné 137
 neporovnateľnosť, 137
 porovnateľnosť, 137
 lineárne 138

Ú

úplný binárny strom 81

V

Vennove diagramy 16
 veta
 Cayleyho 242
 Eulerova o rovinných grafoch 110
 Kuratowského 112
 multinomická 177
 o centre stromu 224
 o cykloch 58
 o existencii cesty v grafe 56
 o existencii jadra grafu 226

veta	
o existencii kostry grafu	63
o explicitnom vyjadrení rekurentných postupností	170
o invariantnosti cyklu C_n	102
o invariantnosti stupňov vrcholov	101
o inverznej funkcii	145
o komponentoch grafu	61
o korektnosti prehľ. algoritmov	191
o matematickej indukcii	39
o matici susednosti	71
o matici zloženej relácie	131
o maticiach susednosti izomorfných grafov	100
o mohutnosti kartézskeho súčinu	19
o optimálnej lokalizácii v prehľadávacom binárnom strome	217
o otvorenom eulerovskom ťahu	87
o počte koncových vrcholov úplných binárnych stromov	81
o potenčnej množine	40
o relácii danej rozkladom množiny	139
o rozklade danom rel. ekvivalencie	141
o správnosti Jarníkovho algoritmu	199
o stromoch	76
o stupňoch vrcholov rovinného grafu	111
o súčte stupňov vrcholov grafu	52
o súvislosti výšky a počtu koncových vrcholov binárnych stromov	82
o tranzitívnosti relácie na množine	132
o uzavretom eulerovskom ťahu	86
o vrchoch stromov	76
pravidlo sylogizmu	34
základná algebry	16
základná aritmetiky	37
veta (matematická)	31
vrchol	
dieťa	78
koncový	50, 78
list	77
otec	78
počiatočný	50
potomok	78
predkovia	78
rodič	78
súrodenci	78
syn	78
vnútorný	78
výroková formula	20, 22

vytvárajúca funkcia	
exponenciálna	175
obyčajná	160
súčet	161
súčin	162
vyvážený binárny strom	81

W

Werner Benjamin	48
-----------------	----

Z

základná veta algebry	16
základná veta aritmetiky	37
zásobník	180
zjednotenie množín	12
zložená funkcia	146
zvyšková trieda	142
zvyšok po delení (modulo)	41