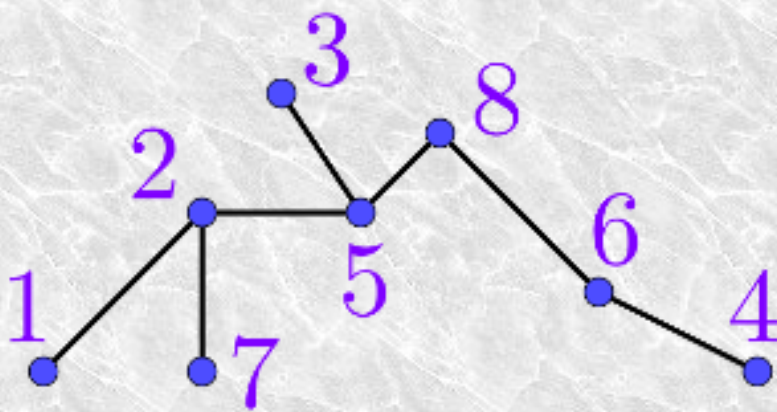




Prüferov kód : 2 5 6 8 2 5

12. Dijkstrov algoritmus, Prüferov kód

V tejto kapitole sa budeme zaoberať dvoma rôznymi problémami. Prvým je hľadanie najlacnejšej cesty Dijkstrovým algoritmom a druhým je Prüferov kód (časť 12.3).

12.1 Hľadanie najlacnejšej cesty Dijkstrovým algoritmom

V kapitole 10 boli definované ohodnotené grafy a popísané dva algoritmy hľadania minimálnej kostry ohodnoteného grafu. Problém hľadania minimálnej kostry daného grafu sa v aplikáciách vyskytuje pomerne často, ale ani zďaleka to nie je jediný problém súvisiaci s ohodnotenými grafmi. Iným častým problémom je napríklad hľadanie najlacnejšej cesty medzi dvoma danými vrcholmi, alebo najlacnejšej cesty z daného vrcholu do všetkých ostatných vrcholov.

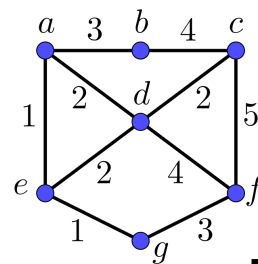
Už v príklade 10.1 sme hľadali „najkratšiu“, alebo lepšie povedané najlacnejšiu, cestu medzi dvoma vrcholmi. Teraz formálne definujeme pojmy *hodnota sledu* a *najlacnejšia cesta*.

Definícia 12.1.1 — Hodnota sledu. Hodnotou sledu v ohodnotenom grafe, nazývame súčet ohodnotení všetkých hrán daného sledu.

P Spomeňme si, že podľa definície 3.3.3 je každá cesta zároveň ťah a každý ťah je zároveň sled, takže predošlá definícia sa rovnako týka aj ťahov a ciest. V prevažnej väčšine praktických aplikácií nás budú zaujímať práve cesty.

■ Príklad 12.1

Hodnota sledu $s = (a, b, c, f, d, e)$ v grafe, ktorý vidíme na obrázku vpravo, je $3 + 4 + 5 + 4 + 2 = 18$.



Keď budeme hľadať sled s najmenšou hodnotou medzi dvoma danými vrcholmi grafu G , tak vždy pôjde o taký ohodnotený graf, ktorého všetky ohodnotenia sú kladné čísla. Pri záporných

ohodnoteniach hrán môže byť cena najlacnejšieho sledu zdola neohraničená, a preto záporné ohodnotenia hrán nebudeme pripúšťať. Takže vo všetkých úlohách, v ktorých pôjde o hľadanie sledu s najmenšou hodnotou, budeme uvažovať len grafy s kladnými ohodnoteniami.

Okrem toho, ak hľadáme sled s najmenšou hodnotou, tak vždy pôjde o cestu. Každá cesta je zároveň sled, v ktorom sa neopakujú hrany ani vrcholy. Triviálne sa dá dokázať tvrdenie, že ak má sled medzi vrcholmi u a v v ohodnotenom grafe G najmenšiu možnú hodnotu, tak potom sa v ňom žiadne hrany ani vrcholy neopakujú. Pozri cvičenie 12.1.

Definícia 12.1.2 — Najlacnejšia cesta. Cesta z vrcholu u do vrcholu v , ktorá má najmenšiu hodnotu, sa nazýva najlacnejšia cesta z u do v .

P V úlohách, v ktorých ohodnotenia hrán grafov budú zodpovedať vzdialenostiam alebo dĺžkam, napríklad grafy dopravnej siete, budeme niekedy hovoriť o *najkratšej* ceste, prípadne *najmenšej vzdialenosti* medzi vrcholmi, tak ako sme to robili v príklade 10.1. Inak budeme vždy hovoriť o najlacnejšej ceste a ceste s najmenšou hodnotou medzi dvoma vrcholmi.

■ **Príklad 12.2** V grafe z príkladu 12.1 má cesta s najmenšou hodnotou (najkratšia cesta) z vrcholu a do vrcholu c hodnotu (dĺžku) 4 a je to cesta (a, d, c) . ■

Úloha nájdenia najlacnejšej cesty medzi danými dvoma vrcholmi v ohodnotenom grafe sa veľmi často vyskytuje v rôznych aplikáciach. Existuje algoritmus, ktorý v ohodnotenom grafe nájde najlacnejšiu cestu medzi dvoma jeho vrcholmi. Jeho autorom je holandský matematik Edsger W. Dijkstra. Existuje viacero modifikácií Dijkstrovho algoritmu. Popis Dijkstrovho algoritmu nájdeme napríklad v [15], [7], alebo [18] a samozrejme nájdeme ho aj na webe, napríklad vo Wikipedii¹. Vo svojej základnej verzii Dijkstrov algoritmus nájde, pre daný ohodnotený graf G a v ňom pevne určený vrchol v , dĺžky najlacnejších ciest od vrcholu v ku všetkým ostatným vrcholom v grafe G . My tu uvedieme drobnú modifikáciu Dijkstrovho algoritmu, ktorá nám okrem dĺžok najlacnejších ciest od vrcholu v umožní tieto cesty aj jednoducho skonštruovať.

12.1.1 Popis Dijkstrovho algoritmu

Pri hľadaní najlacnejšej cesty sa stačí zaoberať obyčajnými grafmi, pretože slučky sa v najlacnejších cestách určite vyskytovať nebudú (pozri cvičenie 12.2). Preto ak by daný graf obsahoval slučky, môžeme ich vynechať. A ak by daný graf obsahoval násobné hrany medzi dvoma vrcholmi, tak stačí brať do úvahy len hranu s najmenšou hodnotou a všetky ostatné hrany medzi týmito dvoma vrcholmi môžeme vynechať.

Majme teda ohodnotený obyčajný graf $G = (V, E)$, v ktorom sú všetky ohodnotenia hrán kladné a pevne daný vrchol v v tomto grafe. Dijkstrov algoritmus nám určí najlacnejšiu cestu medzi každým vrcholom grafu G a vrcholom v . Algoritmus bude každému vrcholu $u \in V$ priradovať značky. Tieto značky budú najskôr dočasné a potom trvalé. V oboch prípadoch bude mať značka tvar $(L(u), \mathbb{P}(u))$. Hodnota $L(u)$ predstavuje pri dočasnej značke momentálne odhadovanú „najmenšiu“ hodnotu cesty medzi vrcholom u a vrcholom v . V prípade trvalej značky je to hodnota najlacnejšej cesty medzi vrcholom u a vrcholom v . Kým má niektorý vrchol len dočasnú značku, tak táto sa môže ešte meniť. Akonáhle je niektorému vrcholu priradená trvalá značka, táto sa už meniť nebude. Množina $\mathbb{P}(u)$ v značke vrcholu sa v pôvodnom Dijkstrovom algoritme nevyskytuje. Je to práve tá modifikácia, ktorá nám po skončení algoritmu umožní skonštruovať najlacnejšie cesty od každého vrcholu grafu G do vrcholu v . Množina $\mathbb{P}(u)$ bude v trvalej značke obsahovať všetkých susedov vrcholu u na cestách s najmenšou hodnotou vedúcich z vrcholu u do vrcholu v .

¹https://en.wikipedia.org/wiki/Dijkstra's_algorithm

P V popise Dijkstrovho algoritmu budeme používať tieto označenia

- ▷ $G = (V, E)$ – ohodnotený obyčajný graf, ktorého všetky ohodnotenia hrán sú kladné,
- ▷ V – množina všetkých vrcholov grafu G ,
- ▷ $w(h)$ – ohodnotenie hrany h v grafe G ,
- ▷ $\mathbb{A}$ – množina „aktívnych“ vrcholov grafu G , čiže tých, ktoré ešte nemajú trvalú značku.

Dijkstrov algoritmus

VSTUP: Ohodnotený obyčajný graf $G = (V, E)$ a vrchol $v \in V$.

VÝSTUP: Pre každé $u \in V$ trvalá značka $(L(u), \mathbb{P}(u))$.

INICIALIZÁCIA

```

 $\mathbb{A} := V;$ 
for { každé  $u \in V$  } do
    |  $L(u) := \infty;$ 
    |  $\mathbb{P}(u) := \{u\};$ 
end
 $L(v) := 0;$ 
 $\mathbb{P}(v) := \{v\};$ 

```

PRIDEĽOVANIE ZNAČIEK VRCHOLOM

```

while { existuje  $x \in \mathbb{A}; L(x) < \infty$  } do
    |  $\delta := \min \{ L(u); u \in \mathbb{A} \};$ 
    |  $\mathbb{N} := \{ u \in \mathbb{A}; L(u) = \delta \};$ 
    |  $\mathbb{A} := \mathbb{A} \setminus \mathbb{N};$ 
    | for { pre každú hranu  $h = \{u, x\}$ , kde  $u \in \mathbb{N}$  a  $x \in \mathbb{A}$  } do
        | if {  $L(x) = L(u) + w(h)$  } then
            | |  $\mathbb{P}(x) := \mathbb{P}(x) \cup \{u\};$ 
        | end
        | if {  $L(x) > L(u) + w(h)$  } then
            | |  $L(x) := L(u) + w(h);$ 
            | |  $\mathbb{P}(x) := \{u\};$ 
        | end
    | end
end

```

V pseudokóde Dijkstrovho algoritmu uvedenom vyššie je podmienka ukončenia algoritmu pre každé $x \in \mathbb{A}$: $L(x) = \infty$. Znamená to, že buď $\mathbb{A} = \emptyset$, alebo sú všetky vrcholy množiny $\mathbb{A}$ nedosiahnuteľné z vrcholu v . V prípade súvislého grafu G môže nastať len prípad $\mathbb{A} = \emptyset$. Po skončení Dijkstrovho algoritmu má každý vrchol grafu G značku $(L(u), \mathbb{P}(u))$, kde $L(u)$ je hodnota najlacnejšej cesty medzi vrcholmi u a v a množina $\mathbb{P}(u)$ obsahuje zoznam všetkých susedov vrcholu u , na cestách s najmenšou hodnotou vedúcich z vrcholu u do vrcholu v . Uvedený algoritmus sa dá triviálne modifikovať aj pre orientované ohodnotené grafy.

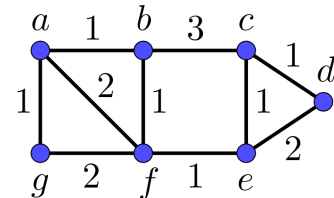
Dôkaz správnosti Dijkstrovho algoritmu sa robí matematickou indukciou a môžeme ho nájsť napríklad v knihe [15]. Slovný popis Dijkstrovho algoritmu je nasledovný:

1. Prvým krokom algoritmu je inicializácia.
 - Množina $\mathbb{A}$ bude obsahovať všetky vrcholy grafu $G = (V, E)$.
 - Pre každý vrchol $u \in V$ bude $\mathbb{P}(u) = \{u\}$.
 - Pre každý vrchol $u \in V$, $u \neq v$ bude $L(u) = \infty$.
 - Pre vrchol v bude $L(v) = 0$.
2. Podmienka ukončenia algoritmu je pre každé $x \in \mathbb{A}$: $L(x) = \infty$.
Čiže ak *existuje* $x \in \mathbb{A}$: $L(x) < \infty$, algoritmus pokračuje.
3. Druhým krokom algoritmu je voľba množiny $\mathbb{N}$ a úprava množiny $\mathbb{A}$.
 - Spomedzi vrcholov $x \in \mathbb{A}$ nájdeme minimálnu hodnotu $L(x)$ a označíme ju δ .
 - Do množiny $\mathbb{N}$ dáme všetky vrcholy $x \in \mathbb{A}$, pre ktoré $L(x) = \delta$.
 - Z množiny $\mathbb{A}$ vyhodíme vrcholy patriace do množiny $\mathbb{N}$.
4. Ďalším krokom algoritmu je úprava dočasných a priradenie trvalých značiek vrcholom.
 - Nájdeme všetky hrany $h = \{u, x\}$, ktorých koncový vrchol u patrí do množiny $\mathbb{N}$ a koncový vrchol x patrí do množiny $\mathbb{A}$. Budeme upravovať značky vrcholov x .
 - Ak platí $L(x) = L(u) + w(h)$, znamená to, že z vrcholu x sa cez vrchol u dostaneme do vrcholu v cestou rovnakej hodnoty, ako má doteraz najlacnejšia nájdená cesta. Takže hodnotu $L(x)$ meniť netreba. Ale do množiny $\mathbb{P}(x)$ pridáme vrchol u ako počiatočný vrchol alternatívnej najlacnejšej cesty do vrcholu v .
 - Ak platí $L(x) > L(u) + w(h)$, tak z vrcholu x do vrcholu v sa vieme dostať cez vrchol u lacnejšou cestou, než sme našli doteraz. Preto zmeníme hodnoty $L(x)$ aj $\mathbb{P}(x)$.
 - Návrat na test podmienky ukončenia algoritmu (2. bod).

Použitie Dijkstrovho algoritmu si teraz ilustrujeme na niekoľkých príkladoch.

■ Príklad 12.3

V ohodnotenom grafe, ktorý je na obrázku vpravo, nájdite hodnoty najlacnejších ciest z vrcholu a do všetkých ostatných vrcholov a všetky najlacnejšie cesty z vrcholu a do vrcholu d .



Priebeh Dijkstrovho algoritmu budeme zapisovať do tabuľky. Pokiaľ sa v niektorom kroku značka vrcholu nebude meniť, tak do príslušného políčka zapíšeme —. Trvalé značky vrcholov budú označené modrou farbou a vrcholmi, ktoré už majú pridelenú trvalú značku, sa nemusíme viac zaoberať.

	a	b	c	d	e	f	g
1.	$(0, \{a\})$	$(\infty, \{b\})$	$(\infty, \{c\})$	$(\infty, \{d\})$	$(\infty, \{e\})$	$(\infty, \{f\})$	$(\infty, \{g\})$
2.	$(0, \{a\})$	$(1, \{a\})$	—	—	—	$(2, \{a\})$	$(1, \{a\})$
3.		$(1, \{a\})$	$(4, \{b\})$	—	—	$(2, \{a, b\})$	$(1, \{a\})$
4.			—	—	$(3, \{f\})$	$(2, \{a, b\})$	
5.			$(4, \{b, e\})$	$(5, \{e\})$	$(3, \{f\})$		
6.			$(4, \{b, e\})$	$(5, \{c, e\})$			
7.				$(5, \{c, e\})$			

Hodnoty najlacnejších ciest z vrcholu a do všetkých ostatných vrcholov grafu odčítame z trvalých značiek vrcholov. Napríklad najlacnejšia cesta do vrcholu c má hodnotu 4 a najlacnejšia cesta

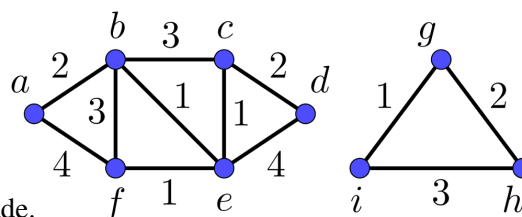
do vrcholu e má hodnotu 3. Ak chceme nájsť najlacnejšie cesty medzi vrcholom a a vrcholom d , tak začneme vo vrchole d , z jeho značky odčítame predposledný vrchol na najlacnejšej ceste, ideme do tohto vrcholu a opäť z jeho značky odčítame ďalší vrchol cesty atď. Takže najlacnejšie cesty medzi vrcholmi a a d sú

$$d \leftarrow \begin{cases} c \leftarrow \begin{cases} b \leftarrow a \\ e \leftarrow f \leftarrow \begin{cases} a \\ b \leftarrow a \end{cases} \end{cases} \\ e \leftarrow f \leftarrow \begin{cases} a \\ b \leftarrow a \end{cases} \end{cases}$$

Z obrázku vidíme, že existuje 5 najlacnejších ciest z vrcholu a do vrcholu d . Sú to cesty (a, b, c, d) , (a, f, e, c, d) , (a, b, f, e, c, d) , (a, f, e, d) a (a, b, f, e, d) . Všetkých päť ciest má hodnotu 5. ■

■ Príklad 12.4

V ohodnotenom grafe, ktorý je na obrázku vpravo, nájdite hodnoty najlacnejších ciest z vrcholu a do všetkých ostatných vrcholov a najlacnejšie cesty z vrcholu a do vrcholu d .



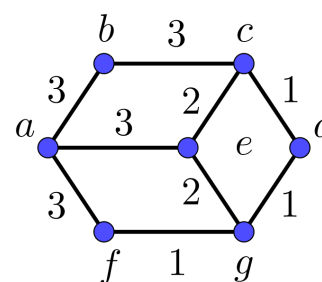
Postupovať budeme rovnako ako v predošlom príklade.

	a	b	c	d	e	f	g	h	i
1.	$(0, \{a\})$	$(\infty, \{b\})$	$(\infty, \{c\})$	$(\infty, \{d\})$	$(\infty, \{e\})$	$(\infty, \{f\})$	$(\infty, \{g\})$	$(\infty, \{h\})$	$(\infty, \{i\})$
2.	$(0, \{a\})$	$(2, \{a\})$	—	—	—	$(4, \{a\})$	—	—	—
3.		$(2, \{a\})$	$(5, \{b\})$	—	$(3, \{b\})$	—	—	—	—
4.			$(4, \{e\})$	$(7, \{e\})$	$(3, \{b\})$	$(4, \{a, e\})$	—	—	—
5.			$(4, \{e\})$	$(6, \{c\})$		$(4, \{a, e\})$	—	—	—
6.				$(6, \{c\})$			—	—	—

Algoritmus v 6. kroku skončil, pretože v aktívnej množine zostali už len vrcholy x so značkou $L(x) = \infty$. To znamená, že tieto vrcholy sú z vrcholu a nedosiahnuteľné. Najlacnejšia cesta z vrcholu a do vrcholu d je len jedna. Je to cesta $d \leftarrow c \leftarrow e \leftarrow b \leftarrow a$. ■

■ Príklad 12.5

V ohodnotenom grafe, ktorý je na obrázku vpravo, nájdite hodnoty najlacnejších ciest z vrcholu a do všetkých ostatných vrcholov a najlacnejšie cesty z vrcholu a do vrcholu d .



Postupovať budeme rovnako ako v predošlých príkladoch.

	a	b	c	d	e	f	g
1.	$(0, \{a\})$	$(\infty, \{b\})$	$(\infty, \{c\})$	$(\infty, \{d\})$	$(\infty, \{e\})$	$(\infty, \{f\})$	$(\infty, \{g\})$
2.	$(0, \{a\})$	$(3, \{a\})$	—	—	$(3, \{a\})$	$(3, \{a\})$	—
3.		$(3, \{a\})$	$(5, \{e\})$	—	$(3, \{a\})$	$(3, \{a\})$	$(4, \{f\})$
4.			—	$(5, \{g\})$			$(4, \{f\})$
5.			$(5, \{e\})$	$(5, \{g\})$			

Najlacnejšia cesta z vrcholu a do vrcholu d je len jedna. Je to cesta $d \leftarrow g \leftarrow f \leftarrow a$ a má dĺžku 5. ■

12.2 Problém obchodného cestujúceho

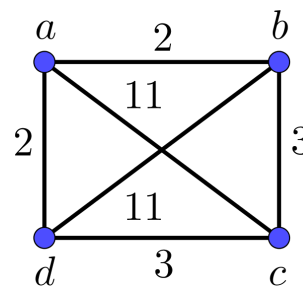
Problém obchodného cestujúceho je podobný problému hamiltonovského cyklu. Je to v podstate problém nájdenia najlacnejšieho hamiltonovského cyklu v ohodnotenom grafe. Rovnako ako problém existencie hamiltonovského cyklu v danom grafe, vznikol aj problém obchodného cestujúceho už v 19. storočí a súvisí s čiste praktickým problémom. Máme niekoľko miest, medzi nimi sú cesty. Tieto cesty sú rôznej dĺžky a kvality a obchodný cestujúci má prejsť všetkými mestami. Aby šetril čas a náklady na cestu, chce prejsť každým mestom práve raz.

Situáciu si môžeme opäť modelovať grafom. Mestám budú zodpovedať vrcholy grafu a cestám budú zodpovedať hrany grafu. Dĺžka ciest, časová náročnosť ciest, alebo cestovné náklady zodpovedajúce jednotlivým cestám budú vyjadrené ohodnotením hrán grafu. Tým sa problém obchodného cestujúceho transformuje na problém hľadania najlacnejšieho hamiltonovského cyklu v ohodnotenom grafe. Úlohou je nájsť najlacnejší, v zmysle dĺžky, dopravných nákladov, času, alebo ďalších kritérií vyjadrených ohodnotením hrán, hamiltonovský cyklus.

Keďže už samotný problém existencie hamiltonovského cyklu v neohodnotenom grafe je NP-úplný, je taký aj problém obchodného cestujúceho. To znamená, že vo všeobecnosti neexistuje jeho efektívne riešenie. Existuje však viacero heuristík, pomocou ktorých sa tento problém dá riešiť. Pri malom počte miest, hrán a vhodných ohodnoteniach, je riešenie pomerne jednoduché. Ukážeme si to na príklade.

■ Príklad 12.6

Problém obchodného cestujúceho budeme riešiť na grafe z obrázku vpravo. Cyklus $c = (a, b, c, d)$ je hamiltonovský cyklus v danom grafe a jeho hodnota je 10. Akýkoľvek cyklus obsahujúci hrany $\{a, c\}$, alebo $\{b, d\}$, ktoré majú dĺžku 11, by mal väčšiu hodnotu. Vidíme teda, že cyklus dĺžky 10 je najlacnejší možný. Preto bude cyklus c riešením problému obchodného cestujúceho v danom grafe.



12.3 Prüferov kód

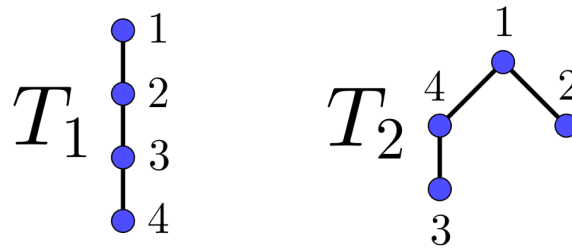
Prüferov kód je efektívny spôsob kódovania očíslovaných stromov. Pod *očíslovaným stromom* rozumieme strom, ktorého vrcholy sú označené číslami². Napríklad očíslované stromy T_1 a T_2 na nasledovnom obrázku



sú rôzne, aj keď ako grafy sú izomorfné. Očíslovaný strom si môžeme predstaviť ako koreňový strom, ktorého koreň je vždy vrchol 1. Stromy T_1 a T_2 sa potom dajú nakresliť aj tak, ako to vidíme na obrázku 12.1 (strana 235). Pri takejto interpretácii môžu mať aj dva izomorfné grafy úplne odlišnú podobu, ako to vidno z uvedeného príkladu.

Pomocou Prüferovho kódu vieme každý očíslovaný strom na n vrchoch, jednoznačne reprezentovať postupnosťou $(p_1, p_2, \dots, p_{n-2})$, čiže ako usporiadanú $(n-2)$ -tícu. Uvedieme si pseudokódy dvoch algoritmov. Pomocou prvého algoritmu budeme z daného očíslovaného stromu

²Niekedy sa namiesto čísel používajú aj písmená, alebo akékoľvek iné symboly s daným lineárnym usporiadaním. V anglickej terminológii sa očíslovaný strom označuje ako „labeled tree“ a v slovenskej terminológii sa občas používa aj pojem „označený strom“.



Obr. 12.1: Očíslované stromy

vytvárať Prüferov kód a pomocou druhého algoritmu budeme z daného Prüferovho kódu rekonštruovať pôvodný strom. V definícii 4.2.3 (strana 77) bol definovaný pojem *list* (stromu). V pseudokóde algoritmu na konštrukciu Prüferovho kódu budeme používať funkciu $\text{sused}(i)$, ktorá nám pre list so značkou i vráti značku jeho suseda. Keďže list je vrchol stromu, ktorého stupeň je 1, je táto funkcia dobre definovaná. Okrem toho budeme používať aj operáciu $\uplus$, ktorá nám pridá jeden prvok na koniec už vytvorenej postupnosti. Napríklad:

$$(2, 6, 1, 3, 4) \uplus 7 = (2, 6, 1, 3, 4, 7).$$

Konštrukcia Prüferovho kódu z daného očíslovaného stromu je veľmi jednoduchá. V každom kroku algoritmu nájdeme v strome list s najmenšou značkou, do kódu pridáme značku jeho suseda a tento list potom z pôvodného stromu odstránime aj s hranou, ktorá s ním incидуje. Toto opakujeme dovtedy, kým nám z pôvodného stromu nezostane len cesta dĺžky 1, t. j. len jedna hrana. Pseudokód algoritmu vytvárania Prüferovho kódu z očíslovaného stromu vyzerá takto

Prüferov kód z očíslovaného stromu

VSTUP: Očíslovaný strom $T = (V, E)$, kde $V = \{1, \dots, n\}$.

VÝSTUP: Postupnosť $P = (p_1, \dots, p_{n-2})$, kde každé $p_i \in \{1, \dots, n\}$.

INICIALIZÁCIA

$P := ()$;

VYTVÁRANIE PRÜFEROVHO KÓDU STROMU T

```

while  $\{ |P| < (n - 2) \}$  do
     $i := \min \{ x; x \text{ je list aktuálneho stromu } T \}$ ;
     $P := P \uplus \text{sused}(i)$ ;
    MODIFIKÁCIA STROMU  $T$ 
     $V := V \setminus \{i\}$ ;
     $E := E \setminus \{i, \text{sused}(i)\}$ ;

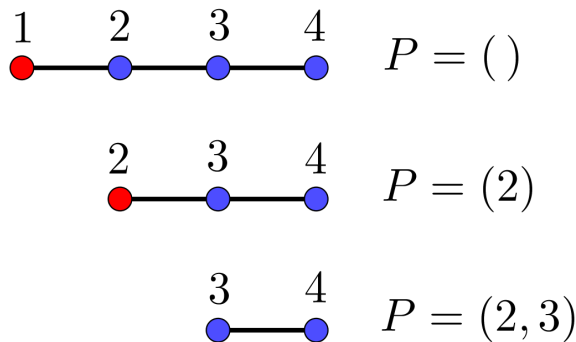
```

end

Tvorbu Prüferovho kódu z očíslovaného stromu si ilustrujeme na príkladoch.

■ **Príklad 12.7** Nájdite Prüferov kód očíslovaného stromu T_1 z obrázku 12.1 (strana 235).

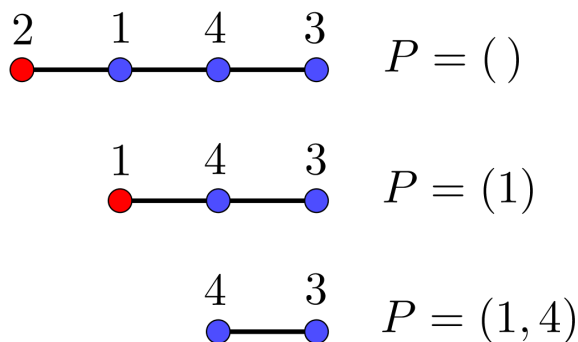
Riešenie: Strom T_1 má 4 vrcholy, takže bude mať dvojčiferný Prüferov kód. Postup jeho tvorby je zobrazený na nasledovnom obrázku



List s najmenšou značkou je 1, takže prvou číslicou kódu bude značka jeho suseda, čo je 2. Potom vrchol 1, aj so s ním incidujúcou hranou, odstránime a pokračujeme ďalej. V druhom kroku je list s najmenšou značkou 2, takže druhá číslica kódu bude značka jeho suseda, čo je 3. Vrchol 2, aj so s ním incidujúcou hranou, odstránime a keďže nám zostala už len jedna hrana, skončili sme. Prüferov kód grafu T_1 z obrázku 12.1 je $P = (2, 3)$. ■

■ **Príklad 12.8** Nájdite Prüferov kód očíslovaného stromu T_2 z obrázku 12.1 (strana 235).

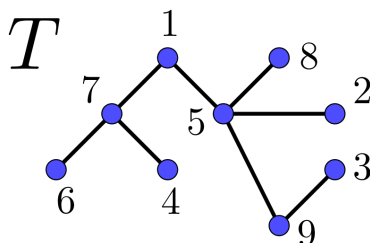
Riešenie: Strom T_2 má 4 vrcholy, takže bude mať dvojčiferný Prüferov kód. Postup jeho tvorby je zobrazený na nasledovnom obrázku



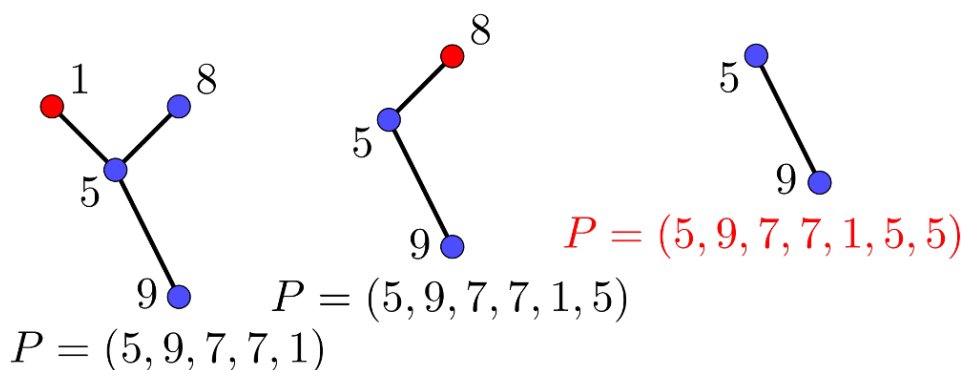
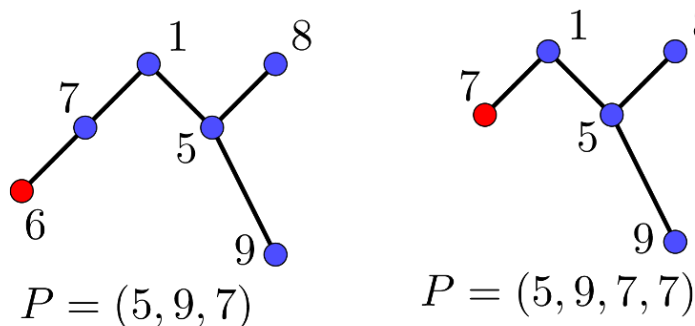
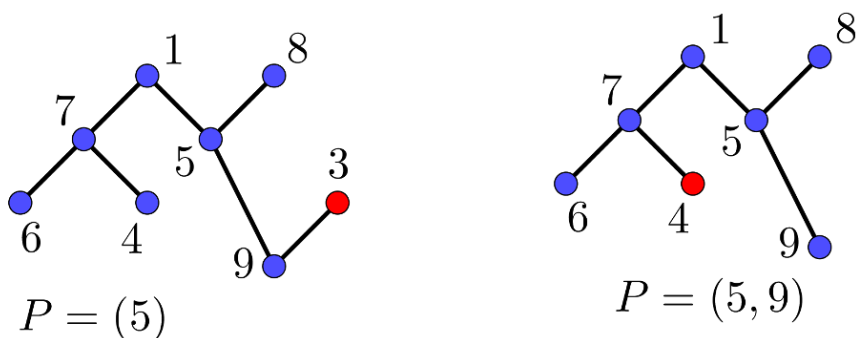
List s najmenšou značkou je 2, takže prvou číslicou kódu bude značka jeho suseda, čo je 1. Potom vrchol 2 aj so s ním incidujúcou hranou odstránime a pokračujeme ďalej. V druhom kroku je list s najmenšou značkou 1, takže druhá číslica kódu bude značka jeho suseda, čo je 4. Vrchol 1, aj so s ním incidujúcou hranou, odstránime, a keďže nám zostala už len jedna hrana, skončili sme. Prüferov kód grafu T_2 z obrázku 12.1 je $P = (1, 4)$. ■

Na predošlých dvoch príkladoch sme videli, že dva rôzne očíslované stromy, aj keď sú izomorfné, majú rôzne Prüferové kódy. Tvorbu Prüferovho kódu si ilustrujeme ešte na jednom príklade.

■ **Príklad 12.9** Nájdite Prüferov kód očíslovaného stromu T z nasledujúceho obrázku



Riešenie: Daný strom má 9 vrcholov, a preto jeho Prüferov kód bude usporiadaná sedmica. Postup riešenia máme zobrazený na nasledovnej sérii obrázkov. Séria sa začína stromom, ktorý už má odstránený vrchol 2 a s ním incidujúcu hranu. Listy stromu T budeme postupne odstraňovať v poradí



Obr. 12.2: Postup konštrukcie Prüferovho kódu stromu z príkladu 12.9

2, 3, 4, 6, 7, 1, 8. Samozrejme nie všetky uvedené vrcholy sú listy pôvodného stromu T . Vrcholy 7 a 1 sa stanú listami až po odstránení niektorých vrcholov pôvodného stromu. Výsledný Prüferov kód očíslovaného stromu T je usporiadaná sedmica $P = (5, 9, 7, 7, 1, 5, 5)$. ■

Ďalej si ukážeme, ako z $(n-2)$ číselného Prüferovho kódu zrekonštruujeme pôvodný n vrcholový očíslovaný strom. Najskôr si musíme uvedomiť dva fakty

1. Ak je Prüferov kód usporiadaná $(n-2)$ -tica, tak pôvodný strom má n vrcholov so značkami $\{1, \dots, n\}$.
2. Pre všetky $i \in \{1, \dots, n\}$, ktoré sa nenachádzajú v Prüferovom kóde, je i značka listu pôvodného stromu.

Pri rekonštrukcii očíslovaného stromu z Prüferovho kódu budeme postupovať presne opačným spôsobom, ako pri vytváraní Prüferovho kódu z očíslovaného stromu. Celý postup si najskôr ilustrujeme na príkladoch a až potom si zapíšeme pseudokód rekonštrukčného algoritmu. Budeme rekonštruovať očíslované stromy z príkladov 12.7, 12.8 a 12.9. Podrobný popis postupu rekonštrukcie uvedieme v texte príkladov.

■ **Príklad 12.10** Zrekonštruujte očíslovaný strom, ktorého Prüferov kód je $P = (2, 3)$.

Riešenie: Prüferov kód je usporiadaná dvojica, takže pôvodný strom má 4 vrcholy so značkami $\{1, 2, 3, 4\}$. Pri rekonštrukcii pôvodného očíslovaného stromu budeme používať pomocnú množinu M . Túto množinu budeme zapisovať ako usporiadanú k -ticu a jej prvky budeme písať vždy v lexikografickom usporiadaní. Na začiatku bude množina M obsahovať³ všetky listy pôvodného stromu, čiže $M = (1, \dots, n) \setminus P$, kde $P = (p_1, \dots, p_{n-2})$ je Prüferov kód. Okrem toho budeme pracovať s množinou L , do ktorej budeme postupne pridávať „listy“, ktoré sme už spojili hranou so s nimi susediacimi vrcholmi. Úvodzovky sú v predošlej vete použité preto, lebo nie všetky vrcholy z množiny L budú listami pôvodného očíslovaného stromu. Pozri poznámku v riešení príkladu 12.9. Na začiatku bude množina L prázdna. Postup rekonštrukcie budeme písať po jednotlivých krokoch. Vždy si uvedieme aktuálny stav množín P , M a L a následne obrázok aktuálne zrekonštruovaného stromu.

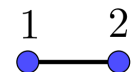
▷ **0. krok**

$$P = (2, 3) \quad L = () \quad M = (1, 2, 3, 4) \setminus (2, 3) = (1, 4)$$

List, ktorý bude spojený s vrcholom so značkou 2, čo je prvé číslo v Prüferovom kóde, musí byť list pôvodného stromu, ktorý mal najmenšiu značku. Čiže prvý vrchol z množiny M . V ďalšom kroku teda spojíme hranou vrcholy 1 a 2, vrchol 1 pridáme do množiny L , vrchol 2 vyhodíme z Prüferovho kódu P a novú množinu M dostaneme ako $M = (1, 2, 3, 4) \setminus (P \cup L)$. List, ktorý budeme spájať s ďalším vrcholom z Prüferovho kódu, musí byť vrchol s najmenšou značkou, ktorá sa nenachádza v aktuálnom Prüferovom kóde, ani medzi „listami“, ktoré sme už spojili s ich susedom.

▷ **1. krok**

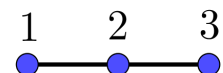
$$P = (3) \quad L = (1) \quad M = (1, 2, 3, 4) \setminus ((3) \cup (1)) = (2, 4)$$



S vrcholom 3, čo je momentálne prvé číslo v Prüferovom kóde, bude spojený vrchol so značkou 2, pretože je to najmenšia značka, ktorá sa nenachádza ani v aktuálnom Prüferovom kóde, ani medzi už použitými listami.

▷ **2. krok**

$$P = () \quad L = (1, 2) \quad M = (1, 2, 3, 4) \setminus (\emptyset \cup (1, 2)) = (3, 4)$$

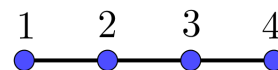


Teraz už máme Prüferov kód prázdny. Je ale zjavné, že z pôvodného stromu nám ešte chýba jedna hrana, čo je hrana, ktorá zostala po skončení tvorby Prüferovho kódu. Jeden vrchol tejto hrany musí byť posledné číslo, ktoré sa nachádzalo v Prüferovom kóde, čiže 3. A druhý vrchol tejto hrany je vrchol z množiny M , ktorý má najmenšiu značku rôznu od 3. Takže bude to vrchol 4.

³Nasledujúci zápis používa symbol odčítania množín, ale M aj P sú usporiadané množiny. Tejto drobnej nezrovnalosti sme si vedomí. Pri odčítaní budeme s M a P pracovať ako s množinami.

▷ **3. krok**

V poslednom kroku už len spojíme vrchol, ktorý bol ako posledný v Prüferovom kóde, s najmenším, od neho rôznym vrcholom v poslednej množine M .



■

Na predošlom príklade sme videli, že v poslednom kroku rekonštrukcie stromu pridávame hranu, ktorá nám ako posledná zostala pri tvorbe Prüferovho kódu. Jedným vrcholom tejto hrany musí byť posledný vrchol, uvedený v Prüferovom kóde. V ďalších príkladoch preto budeme zapisovať množinu P tak, že jej posledný prvok zopakujeme dvakrát, čo nám zjednotí popis rekonštruktívneho algoritmu pre všetky kroky, vrátane toho posledného. Čiže Prüferov kód P budeme zapisovať takto $P = (p_1, \dots, p_{n-2}, p_{n-2})$. Okrem toho pre väčšiu prehľadnosť už nebudeme písať ako sme dostali aktuálnu množinu M , čiže $M = (1, \dots, n) \setminus (P \cup L)$, ale len jej výslednú podobu.

■ **Príklad 12.11** Zrekonštruujte očíslovaný strom, ktorého Prüferov kód je $P = (1, 4)$.

Riešenie: Prüferov kód je usporiadaná dvojica, takže pôvodný strom má 4 vrcholy so značkami $\{1, 2, 3, 4\}$. Postup riešenia bude rovnaký ako v predošlom príklade a opäť ho budeme zapisovať po krokoch.

▷ **0. krok**

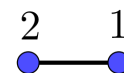
$$P = (1, 4, 4) \quad L = () \quad M = (2, 3)$$

V ďalšom kroku spojíme vrcholy 2 a 1.

▷ **1. krok**

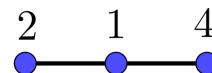
$$P = (4, 4) \quad L = (2) \quad M = (1, 3)$$

V ďalšom kroku spojíme vrcholy 1 a 4.

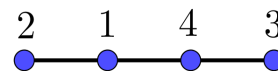
▷ **2. krok**

$$P = (4) \quad L = (2, 1) \quad M = (3)$$

V ďalšom kroku spojíme vrcholy 3 a 4.

▷ **3. krok**

Po pridaní poslednej hrany, spojením vrcholov 3 a 4, sme dostali pôvodný očíslovaný strom.



■

■ **Príklad 12.12** Zrekonštruujte očíslovaný strom, ktorého Prüferov kód je $P = (5, 9, 7, 7, 1, 5, 5)$.

Riešenie: Prüferov kód je usporiadaná sedmica, takže pôvodný strom má 9 vrcholov so značkami $\{1, 2, 3, 4, 5, 6, 7, 8, 9\}$. Na začiatku rekonštrukcie budeme mať množinu $P = (5, 9, 7, 7, 1, 5, 5)$, množina L bude prázdna a aktuálnu množinu M dostaneme vždy ako $M = (1, \dots, 9) \setminus (P \cup L)$. V každom kroku spojíme a , prvý vrchol množiny M , s vrcholom b , prvým vrcholom množiny P . Potom vrchol a pridáme do množiny L , vrchol b vyhodíme z množiny P a určíme si novú množinu M . Jednotlivé kroky budú vyzeráť takto

▷ **0. krok**

$$P = (5, 9, 7, 7, 1, 5, 5) \quad L = () \quad M = (2, 3, 4, 6, 8)$$

V ďalšom kroku spojíme vrcholy 2 a 5.

▷ **1. krok**

$$P = (9, 7, 7, 1, 5, 5, \mathbf{5}) \quad L = (2) \quad M = (3, 4, 6, 8)$$

V ďalšom kroku spojíme vrcholy 3 a 9.

▷ **2. krok**

$$P = (7, 7, 1, 5, 5, \mathbf{5}) \quad L = (2, 3) \quad M = (4, 6, 8, 9)$$

V ďalšom kroku spojíme vrcholy 4 a 7.

▷ **3. krok**

$$P = (7, 1, 5, 5, \mathbf{5}) \quad L = (2, 3, 4) \quad M = (6, 8, 9)$$

V ďalšom kroku spojíme vrcholy 6 a 7.

▷ **4. krok**

$$P = (1, 5, 5, \mathbf{5}) \quad L = (2, 3, 4, 6) \quad M = (7, 8, 9)$$

V ďalšom kroku spojíme vrcholy 7 a 1.

▷ **5. krok**

$$P = (5, 5, \mathbf{5}) \quad L = (2, 3, 4, 6, 7) \quad M = (1, 8, 9)$$

V ďalšom kroku spojíme vrcholy 1 a 5.

▷ **6. krok**

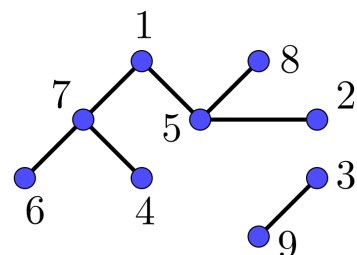
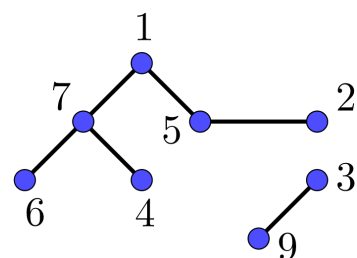
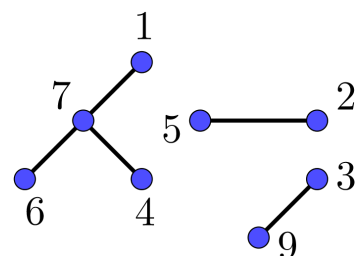
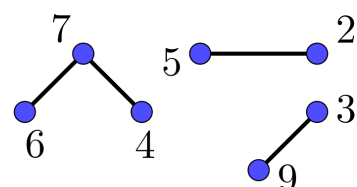
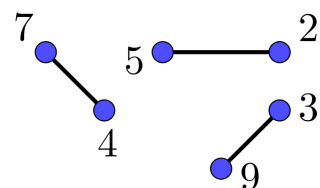
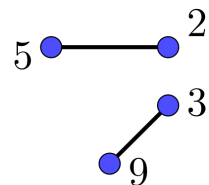
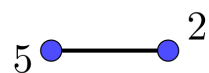
$$P = (5, \mathbf{5}) \quad L = (2, 3, 4, 6, 7, 1) \quad M = (8, 9)$$

V ďalšom kroku spojíme vrcholy 8 a 5.

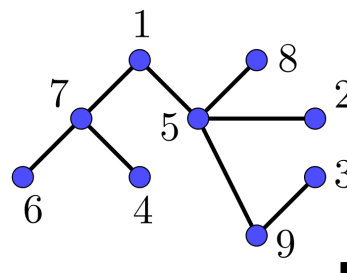
▷ **7. krok**

$$P = (\mathbf{5}) \quad L = (2, 3, 4, 6, 7, 1, 8) \quad M = (9)$$

V ďalšom, už poslednom, kroku spojíme vrcholy 9 a 5, čím dostaneme pôvodný očíslovaný strom. V poslednom kroku, keď už množina P je prázdna, nepotrebujeme ani určovať novú množinu M .



▷ 8. krok

$$P = () \quad L = (2, 3, 4, 6, 7, 1, 8, 9) \quad M = (5)$$


Pri popise pseudokódu konštrukcie očíslovaného stromu z Prüferovho kódu budeme používať funkciu *prvy* (P), ktorá nám z usporiadanej k -tice P vráti jej prvý prvok a zároveň tento prvok z P odstráni. Rovnakú funkciu sme používali aj v pseudokódoch Kruskalovho algoritmu (strana 202) a algoritmu na konštrukciu prehľadavacieho binárneho stromu (strana 215).

Korektnosť algoritmov konštrukcie očíslovaného stromu z Prüferovho kódu a naopak, sa dokazuje ľahko. Treba ukázať dve veci. Po prvé treba dokázať, že uvedeným algoritmom vždy vytvoríme očíslovaný strom a po druhé treba dokázať, že zo zrekonštruovaného stromu spätne dostaneme pôvodný Prüferov kód. Dôkaz korektnosti je uvedený napríklad v knihe [15] v časti 7.4.

Očíslovaný strom z Prüferovho kódu

VSTUP: Postupnosť $P = (p_1, \dots, p_{n-2})$, kde každé $p_i \in \{1, \dots, n\}$.

VÝSTUP: Očíslovaný strom $T = (V, E)$, kde $V = \{1, \dots, n\}$.

INICIALIZÁCIA

$$V := \{1, \dots, n\};$$

$$E := \{\};$$

$$P := (p_1, \dots, p_{n-2}, p_{n-2});$$

$$L := \{\};$$

$$M := V \setminus (P \cup L);$$

REKONŠTRUKCIA OČÍSLOVANÉHO STROMU T

```

while  $\{|P| > 0\}$  do
     $i := \min \{x; x \in M\};$ 
     $j := \text{prvy}(P);$ 
     $E := E \cup \{i, j\};$ 
     $L := L \cup \{i\};$ 
     $M := V \setminus (P \cup L);$ 
end

```

Ako sme mali možnosť vidieť z uvedených príkladov, konštrukcia Prüferovho kódu z očíslovaného stromu, rovnako ako rekonštrukcia očíslovaného stromu z Prüferovho kódu, je veľmi jednoduchá. Prüferov kód je pritom veľmi efektívny spôsob reprezentácie stromov. Autorom tohto kódu je nemecký matematik Heinz Prüfer, ktorý tento kód navrhol v roku 1918 v súvislosti s dôkazom Cayleyho⁴ vety. Znenie Cayleyho vety je nasledovné.

⁴Arthur Cayley bol slávny britský matematik z 19. storočia, zaoberajúci sa diskretnou matematikou, predovšetkým algebrou a teóriou grafov.

Veta 12.3.1 — Cayleyho veta. Počet rôznych kostier očíslovaného kompletného grafu na n vrcholoch je n^{n-2} . Alebo inak sformulované: Počet rôznych koreňových stromov s vrcholmi $V = \{1, \dots, n\}$ a koreňom 1 je n^{n-2} .

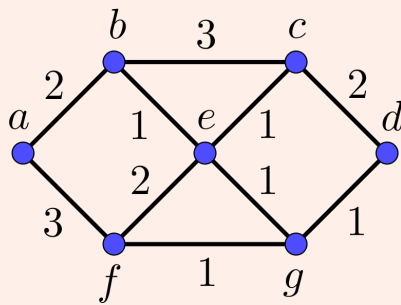
Dôkaz Cayleyho vety je s použitím Prüferovho kódu triviálny. Prenechávame ho ako cvičenie 12.8.

12.4 Cvičenia

Cvičenie 12.1 Majme ohodnotený graf G , ktorého všetky ohodnotnia hrán sú kladné. Nech s je najlacnejší sled medzi vrcholmi u a v v tomto grafe. Dokážte potom, že sled s je cesta, t. j. neopakujú sa v ňom ani hrany, ani vrcholy. ■

Cvičenie 12.2 Dokážte, že najlacnejší sled medzi dvoma vrcholmi daného grafu s kladnými ohodnoteniami hrán neobsahuje slučky. ■

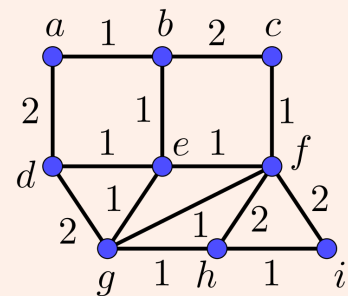
Cvičenie 12.3



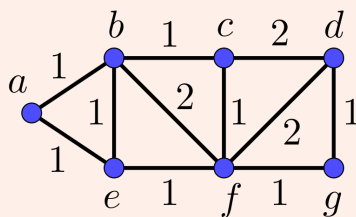
- Nájdite hodnoty najlacnejších ciest od vrcholu d ku všetkým ostatným vrcholom grafu.
- Nájdite všetky najlacnejšie cesty z vrcholu d do vrcholu a .

Cvičenie 12.4

- Nájdite hodnoty najlacnejších ciest od vrcholu d ku všetkým ostatným vrcholom grafu.
- Nájdite všetky najlacnejšie cesty z vrcholu a do vrcholu g .



Cvičenie 12.5

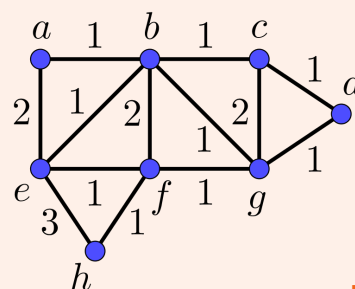


- Nájdite hodnoty najlacnejších ciest od vrcholu a ku všetkým ostatným vrcholom grafu.
- Nájdite všetky najlacnejšie cesty z vrcholu a do vrcholu d .

Cvičenie 12.6 Naprogramujte Dijkstrov algoritmus v Pythone alebo v C. ■

Cvičenie 12.7

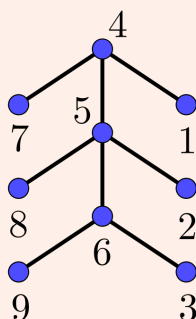
- (a) Nájdite hodnoty najlacnejších ciest od vrcholu a ku všetkým ostatným vrcholom grafu.
- (b) Nájdite všetky najlacnejšie cesty z vrcholu a do vrcholov d a h .

**Cvičenie 12.8** Dokážte Cayleyho vetu (strana 242).

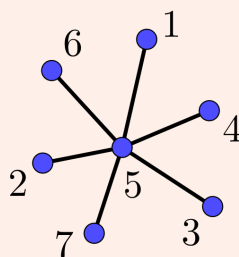
Návod: použite Prüferov kód.

Cvičenie 12.9 Nájdite Prüferov kód pre nasledujúce očíslované stromy

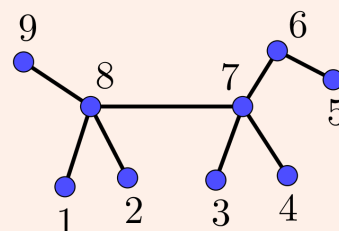
(a)



(b)



(c)

**Cvičenie 12.10** Pre nasledujúce Prüferové kódy zrekonštruujte pôvodné očíslované stromy(a) $(3, 3, 3, 3, 3)$ (d) $(4, 4, 4, 2, 2, 2, 3)$ (g) $(1, 1, 1, 2, 2, 3)$ (b) $(1, 2, 3, 4, 5)$ (e) $(5, 4, 3, 2, 1)$ (h) $(3, 2, 1, 4, 2, 4)$ (c) $(2, 2, 3, 3, 5)$ (f) $(1, 2, 2, 3, 3, 3)$ (i) $(5, 3, 2, 4, 2, 5, 2)$ **Cvičenie 12.11** Vrcholy cesty dĺžky 4 očísľujte číslami z množiny $\{1, \dots, 5\}$ tak, aby jej Prüferov kód obsahoval čo najmenej rôznych čísiel.**Cvičenie 12.12** V Prüferovom kóde nejakého stromu, ktorý je usporiadanou päticou čísiel, sa jedno číslo opakovalo trikrát a druhé dvakrát. Koľko má tento strom vrcholov, a aké sú ich stupne? Uveďte príklad takého stromu.**Cvičenie 12.13** Nájdite graf, v ktorom má každý vrchol stupeň aspoň 2 a ktorý má kostru získanú prehľadávaním do šírky izomorfnú

- (a) s cestou dĺžky n ,
- (b) s hviezdou na n vrcholoch (graf $K_{1,(n-1)}$),
- (c) so stromom, ktorého Prüferov kód je $(3, 3, 3, 1, 4, 4, 2)$,
- (d) so stromom, ktorého Prüferov kód je $(2, 1, 3, 1, 4)$.

Cvičenie 12.14 Nájdite graf, v ktorom má každý vrchol stupeň aspoň 2 a ktorý má kostru získanú prehľadávaním do hĺbky izomorfnú

- (a) s cestou dĺžky n ,
- (b) s hviezdou na n vrchoch – graf $K_{1,(n-1)}$,
- (c) so stromom, ktorého Prüferov kód je $(3, 3, 3, 1, 4, 4, 2)$,
- (d) so stromom, ktorého Prüferov kód je $(2, 1, 3, 1, 4)$.

Cvičenie 12.15 Nájdite graf, ktorého každá kostra získaná prehľadávaním do šírky, je izomorfná so stromom, ktorého Prüferov kód je $(2, 2, 1, 3, 3, 1, 4, 4)$.

Cvičenie 12.16 V Pythone alebo v C naprogramujte vytvorenie Prüferoveho kódu ku zadanému očíslovanému stromu.

Cvičenie 12.17 V Pythone alebo v C naprogramujte rekonštrukciu očíslovaného stromu zo zadaného Prüferoveho kódu. Strom môžete zapísať napríklad pomocou matice susedností.



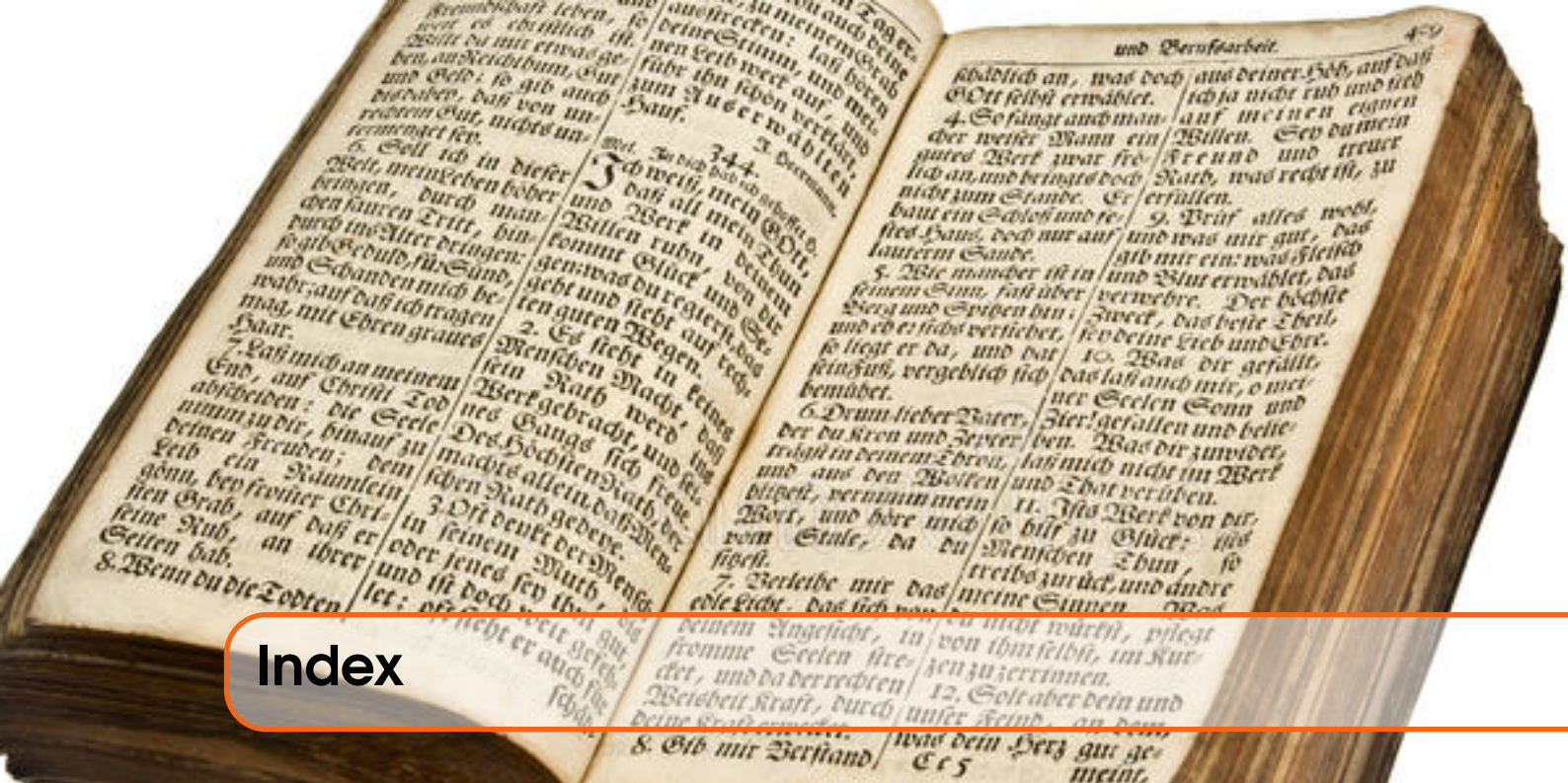
Použitá a odporúčaná literatúra

- [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

$\in$ operátor „patrí do množiny“	11
$\notin$ operátor „nepatrí do množiny“	11
$\subset$ operátor vlastnej podmnožiny	12
$\subseteq$ operátor nevlastnej podmnožiny	12
$\cup$ operátor zjednotenia množín	12
$\cap$ operátor prieniku množín	13
$\setminus$ operátor rozdielu množín	13
$\bar{A}$ komplement množiny A	14
$\neg$ operátor logickej negácie	20
$\vee$ operátor disjunkcie	20
$\wedge$ operátor konjunkcie	20
$\Leftrightarrow$ operátor ekvivalencie	20
$\Rightarrow$ operátor implikácie	20
$\exists$ existenčný kvantifikátor	23
$\forall$ všeobecný kvantifikátor	23
$ $ operátor „ a delí b “ ($a b$)	40
$\triangle$ 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 Prüferovho kódu	241

B

konštrukcie optimálneho Huffmanovho kódu	206
konštrukcie Prüferovho kódu z očíslovaného 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 binárneho stromu	219
zostavovania identifikačného kódu koreňového 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
prehľadávací 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