Riešenia 4. sady domácich úloh

Úloha 1

(1,5 bodu) Dokážte, že relácia

$$R = \left\{(a, b) \in \mathbb{Z}^+ \times \mathbb{Z}^+;\ \exists k \in \mathbb{Z}\colon \frac{a}{b} = 2^k\right\}$$

je reláciou ekvivalencie a popíšte rozklad, ktorý indukuje.

Reflexívnosť. Relácia R je reflexívna, lebo pre každé $a \in \mathbb{Z}^+$ platí $a/a = 2^0$ a $0 \in \mathbb{Z}$.

Symetrickosť. Relácia R je symetrická, lebo pre každé a, $b \in \mathbb{Z}^+$ platí

$$aRB \Rightarrow \exists k \in \mathbb{Z}\colon \frac{a}{b} = 2^k \Rightarrow \frac{a}{b} = 2^k \Rightarrow \frac{b}{a} = 2^{-k} \Rightarrow bRa,$$

keďže $-k \in \mathbb{Z}$.

Tranzitívnosť. Relácia R je tranzitívna, lebo pre každé a, b, $c \in \mathbb{Z}^+$ platí

$$\begin{gathered} aRb \wedge bRc \\ \Downarrow \\ \frac{a}{b} = 2^k \wedge \frac{b}{c} = 2^\ell, \quad \text{pre nejaké } k, \ell \in \mathbb{Z} \\ \Downarrow \\ \frac{a}{c} = \frac{a}{b} \cdot \frac{b}{c} = 2^k \cdot 2^\ell = 2^{k+\ell} \\ \Downarrow \\ aRc, \end{gathered}$$

keďže $k + \ell \in \mathbb{Z}$.

Teda relácia R je reláciou ekvivalencie na $\mathbb{Z}^+$.

Rozklad. Najprv určíme, ako vyzerá trieda $R[n]$ obsahujúca $n \in \mathbb{Z}^+$. Tá obsahuje všetky také čísla a, pre ktoré platí $aRb \Leftrightarrow \exists k \in \mathbb{Z}\colon a/n = 2^k \Leftrightarrow \exists k \in \mathbb{Z}\colon a = n \cdot 2^k$. Teda číslo n je v triede so všetkými kladnými celými číslami v tvare $n \cdot 2^k$ pre nejaké celé číslo k.

Poďme vypisovať triedy postupne:

$$\begin{aligned} R[1] &= \{1, 2, 4, 8, 16, \dots\} = \{2^k;\ k \in \mathbb{N}\}, \\ R[3] &= \{3, 6, 12, 24, \dots\} = \{3 \cdot 2^k;\ k \in \mathbb{N}\}, \\ R[5] &= \{5, 10, 20, 40, \dots\} = \{5 \cdot 2^k;\ k \in \mathbb{N}\}, \\ R[7] &= \{7, 14, 28, 56, \dots\} = \{7 \cdot 2^k;\ k \in \mathbb{N}\}, \\ R[9] &= \{9, 18, 36, 72, \dots\} = \{9 \cdot 2^k;\ k \in \mathbb{N}\}, \\ &\vdots \end{aligned}$$

Novú triedu vytvárame len vtedy, keď treba. Preto sme vynechali $R[2]$, keďže $2 \in R[1]$, triedu $R[6]$ sme vynechali, lebo $6 \in R[3]$ a podobne ďalej. Takto si všimneme, že novú triedu treba vytvoriť pre každé nepárne číslo. Pre nepárne číslo n je číslo $n \cdot 2^k$ kladná celé práve vtedy, keď $k \in \mathbb{N} = \{0, 1, 2, \dots\}$. Rozklad indukovaný reláciou R teda pozostáva z tried

$$R[n] = \{n \cdot 2^k;\ k \in \mathbb{N}\} \qquad \text{pre každé nepárne prirodzené číslo } n.$$

Formálne zapísané, hľadaný rozklad je

$$\{\{n \cdot 2^k;\ k \in \mathbb{N}\};\ n \in \mathbb{N} \wedge 2 \mid n\}.$$

Z prednášky vieme, že uvažujeme triedu $R[n]$ pre každé $n \in \mathbb{Z}^+$, tak dostaneme rozklad $\mathbb{Z}^+$. Takto však máme viaceré triedy uvedení viackrát, čo nám nedáva úplnú informáciu o rozklade. Preto sme sa obmedzili na to, že uvažujeme len nepárne čísla n. Aby sme dokázali, že tento rozklad je správny, tak nám ostáva ukázať, že týmto obmedzením na nepárne čísla dostaneme každú triedu rozkladu práve raz.

1. Ak sú a, b rôzne nepárne čísla, tak neplatí aRb: bez ujmy na všeobecnosti nech $a > b$, potom $a/b = 2^k$ a $a/b > 1$, teda $k \geq 0$ – potom však $a = 2^k \cdot b$, teda a by bolo párne; spor. Preto a, b sú v iných triedach, čiže žiadnu triedu sme neuviedli zbytočne dvakrát.

2. Ukážeme, že každé kladné celé číslo a je vo vzťahu s nejakým nepárnym číslom. Nech n je súčin všetkých nepárnych prvočísel z prvočíselného rozkladu čísla a. Potom podiel a/n bude obsahovať len prvočísla 2, teda bude rovný 2^k pre nejaké celé k. Preto každú triedu dostaneme práve raz.

Úloha 2

(1,5 bodu) Na intervale $(1;\infty)$ definujeme reláciu $\sqsubseteq$ takú, že $x \sqsubseteq y \Leftrightarrow (5x < y \vee x = y)$ pre všetky $x, y \in (1;\infty)$. Rozhodnite, či ide o reláciu usporiadania na $(1;\infty)$. Ak áno, tak určte všetky jej minimálne, najmenšie, maximálne a najväčšie prvky. Všetky tvrdenia dokážte.

Reflexívnosť Relácia $\sqsubseteq$, lebo pre každé $a \in (1;\infty)$ platí

$$a = a \Rightarrow 5a < a \vee a = a \Rightarrow a \sqsubseteq a.$$

Antisymetrickosť Relácia $\sqsubseteq$ je antisymetrická, lebo pre každé $a, b \in (1;\infty)$ platí

$$\begin{gathered} a \sqsubseteq b \wedge b \sqsubseteq a \\ \Downarrow \\ (5a < b \vee a = b) \wedge (5b < a \vee a = b) \\ \Downarrow \\ a = b \vee (5a < b \wedge 5b < a). \end{gathered}$$

Keďže a, b sú kladné, tak platí $a < 5a$, $b < 5b$. Preto ak platí $5a < b \wedge 5b < a$, tak platí

$$a < 5a < b < 5b < a,$$

teda $a < a$, čo je spor. Preto je výroková forma $5a < b \wedge 5b < a$ vždy nepravdivá, teda musí platiť $a = b$.

Tranzitívnosť Relácia $\sqsubseteq$ je tranzitívna. Dokážeme, že pre každé $a, b, c \in (1; \infty)$ platí

$$(a \sqsubseteq b \wedge b \sqsubseteq c) \Rightarrow a \sqsubseteq c.$$

Pre $a = b$ zjavne platí $(a \sqsubseteq a \wedge a \sqsubseteq c) \Rightarrow a \sqsubseteq c$ a takisto pre $b = c$ platí $(a \sqsubseteq b \wedge b \sqsubseteq b) \Rightarrow a \sqsubseteq b$. Preto vo zvyšku dôkazu budeme predpokladať, že $a \neq b \wedge b \neq c$:

$$\begin{gathered} a \sqsubseteq b \wedge b \sqsubseteq c \\ \Downarrow \\ (5a < b \vee a = b) \wedge (5b < c \vee b = c) \\ \Downarrow \;\; a \neq b, b \neq c \\ 5a < b \wedge 5b < c \\ \Downarrow \;\; b < 5b, \text{lebo} 0 < b \\ 5a < c \\ \Downarrow \\ a \sqsubseteq c. \end{gathered}$$

Teda relácia $\sqsubseteq$ je usporiadaním na množine $(1; \infty)$. Pri určovaní význačných prvkov využijeme, že pre ostré usporiadanie $\sqsubset$ vytvorené z usporiadania $\sqsubseteq$ platí $x \sqsubset y \Leftrightarrow ((5x < y \vee x = y) \wedge x \neq y) \Leftrightarrow 5x < y)$.

Minimálne prvky sú všetky čísla z intervalu $(1; 5\rangle$:

- Každé číslo $x \in (1; 5\rangle$ je minimálny prvok. Pre spor predpokladajme, že existuje $a \neq x$, pre ktoré platí $a \sqsubset x$, teda $5a < x$. Potom $a < x/5 \leq 5/5 = 1$, teda $a < 1$, čo je spor. Teda od čísla x nemôže existovať ostro menšie, preto je to minimálny prvok.

- Žiadne číslo $x > 5$ nie je minimálnym prvkom, lebo platí

$$\frac{5+x}{10} \sqsubset x \Leftrightarrow 5 \cdot \frac{5+x}{10} < x \Leftrightarrow 5 + x < 2x \Leftrightarrow 5 < x$$

  a taktiež $(5 + x)/10 > (5 + 5)/10 > 1$, teda ide o číslo z intervalu $(1; \infty)$. Tento bod vieme však zdôvodniť aj tak, že si zvolíme ľubovoľné reálne číslo $a \in (1, x/5)$ – keďže $x > 5$, tak $x/5 > 1$ a medzi ľubovoľnými dvoma reálnymi číslami existuje iné reálne číslo. Pre takto zvolené a platí $5a < x$, teda $a \sqsubset x$.

Najmenší prvok neexistuje, keďže existuje viac minimálnych prvkov (napr. 2 a 3).

Maximálny prvok neexistuje, nakoľko pre každé $x \in (1; \infty)$ platí $x \sqsubseteq 6x \Leftrightarrow 5x < 6x \Leftrightarrow 5 < 6$.

Najväčší prvok neexistuje, nakoľko neexistuje maximálny prvok.

Úloha 3

(2 body) O nasledovných množinách rozhodnite a dokážte, či sú spočítateľné:

a) Množina A, ktorá obsahuje všetky postupnosti $(a_n)_{n=0}^{\infty}$ prirodzených čísel také, že pre všetky celé $n \geq 0$ platí $a_{2n+1} \geq a_{2n}$ a zároveň $a_{2n+2} \leq a_{2n+1}$.

b) Množina B, ktorá obsahuje všetky postupnosti $(a_n)_{n=0}^{\infty}$ prirodzených čísel také, že $a_0 = 0$ a pre všetky celé $n \geq 2$ sa člen a_n nachádza v uzavretom intervale, ktorého krajné body sú a_{n-1} a a_{n-2} (teda $a_2 \in \langle a_0, a_1 \rangle$, $a_3 \in \langle a_2, a_1 \rangle$, $a_4 \in \langle a_2, a_3 \rangle$, $\ldots$).

Podúloha a)

Množina A je nespočítateľná. Ukážeme to diagonalizáciou. Čiže pre spor, nech A je spočítateľná, takže každej postupnosti z A vieme priradiť jednoznačné prirodzené číslo. To znamená, že máme bijekciu z $\mathbb{N}$ do A takú, že číslo 0 sa zobrazí do postupnosti a_0 s prvkami $a_{0,0}, a_{0,1}, a_{0,2}, \dots$, číslo 1 sa zobrazí do postupnosti a_1 s prvkami $a_{1,0}, a_{1,1}, a_{1,2}, \dots$ a tak ďalej. Chceli by sme vytvoriť takú postupnosť p, ktorá sa medzi nimi nenachádza.

$$
\begin{array}{ccccccccccc}
0 & \leftrightarrow & a_{0,0} & \leq & a_{0,1} & \geq & a_{0,2} & \leq & a_{0,3} & \geq \cdots \\
1 & \leftrightarrow & a_{1,0} & \leq & a_{1,1} & \geq & a_{1,2} & \leq & a_{1,3} & \geq \cdots \\
2 & \leftrightarrow & a_{2,0} & \leq & a_{2,1} & \geq & a_{2,2} & \leq & a_{2,3} & \geq \cdots \\
\vdots & & \vdots & & \vdots & & \vdots & & \vdots & \\
\hline\hline
x & \leftrightarrow & p_0 & \leq & p_1 & \geq & p_2 & \leq & p_3 & \geq \cdots
\end{array}
$$

Ak to však budeme robiť úplne bezhlavo, napríklad $p_i = a_{i,i} + 1$, výsledná postupnosť nemusí spĺňať podmienky nerovnosti, a teda ani patriť A. Aby sme však mohli povedať, že diagonalizáciou sme dosiahli spor, musíme nájsť postupnosť, ktorá **patrí do množiny** A a nepriradili sme jej žiadne číslo.

Ako nápad, ako braním „diagonálnych“ prvkov do postupnosti nepokaziť nerovnosti, nám môže prísť, že budeme brať každý druhý index v postupnosti p. Formálne, pre všetky i určíme $p_{2i+1} = a_{i,2i+1} + 1$. Tým určite dosiahneme, že naša postupnosť sa bude aspoň v jednom prvku líšiť od ľubovoľnej z očíslovaných postupností.

Ešte však nevieme, čo sa nachádza na prvkoch s párnymi indexami. Mohli by sme sa napríklad pozrieť na susedné prvky a dať tam menší z nich. Tým by sme zrejme splnili podmienky nerovností. Ak sa nám s tým však nechce babrať, môžeme na párne indexy dať aj samé nuly, lebo tie sú menšie alebo rovné ako čokoľvek iné.

Dostali sme tak postupnosť p, ktorá patrí do množiny A, no nemá priradené žiadne prirodzené číslo – totiž od postupnosti a_i sa líši na indexe $2i + 1$: $p_{2i+1} = a_{i,2i+1} + 1 \neq a_{i,2i+1}$. Tým sme došli k sporu, a teda A je nespočítateľná.

Výsledný spor možno nájsť aj takto. Keďže je postupnosť p v množine A, tak má nejaké číslo. Nech teda $p = a_i$. Pre jej člen na indexe $2i+1$ tak platí $p_{2i+1} = a_{i,2i+1}$. Avšak $p_{2i+1} = a_{i,2i+1}+1$, teda $a_{i,2i+1}+1 = a_{i,2i+1}$, čiže $0 = 1$, čo je spor.

Iné riešenie. Nech P je množina nekonečných binárnych postupností. Chceme ukázať, že z P existuje injekcia do A, čiže $|A| \geq |P| > |\mathbb{N}|$. Takouto injekciou môže byť napríklad $f((a_0, a_1, a_2, \dots)) = (a_0, 1, a_1, 1, a_2, 1, \dots)$. Funkcia f je injekciou, lebo keď si vezmeme dva možné obrazy, ktoré sa rovnajú, čiže $(a_0, 1, a_1, 1, a_2, 1, \dots) = (b_0, 1, b_1, 1, b_2, 1, \dots)$, tak $a_0 = b_0$, $a_1 = b_1, a_2 = b_2, \dots$, a teda aj postupnosti, ktoré boli vzormi týchto obrazov, sa museli rovnať.

Podúloha b)

Množina B však už spočítateľná je. Všimnime si, že po každom prvku sa nám hodnota $|a_{i+1} - a_i|$ nezmení alebo zmenší. Navyše, keďže je to prirodzené číslo, nemôže klesať donekonečna, preto od nejakého momentu bude platiť $|a_{i+1} - a_i| = k$ pre nejaké konštantné k, a teda táto postupnosť bude oscilovať medzi dvoma členmi.

V tejto úlohe máme teda dva stupne „nekonečnej“ voľnosti. Prvou je to, ako si zvolíme člen a_1, druhou je to, po koľkých členoch sa nám postupnosť zacyklí. Môžeme teda rozdeliť postupnosti v B podľa týchto dvoch informácií, čiže $B_{a_1,n}$ budú postupnosti, ktorých prvý člen je a_1 a zacyklia sa po n prvkoch. Uvedomme si, že mohutnosť každej z množín $B_{a_1,n}$ je konečná, pretože na prvých n miestach máme nanajvýš $(a_1 + 1)^n$ možností[1], ako postupnosť môže vyzerať a potom už všetky členy sú jednoznačne určené.

[1] Tento odhad je oveľa väčší, ako tých postupností v skutočnosti je, to nám ale nevadí, my sme chceli len ukázať, že ich je

No a nakoniec už použijeme podobnú myšlienku ako pri dvojiciach prirodzených čísel. Tieto množiny postupností si dáme do tabuľky. Teraz vieme robiť to, že pôjdeme po diagonálach, čiže najskôr vymenujeme

$B_{0,0}$	$B_{0,1}$	$B_{0,2}$	$\cdots$
$B_{1,0}$	$B_{1,1}$	$B_{1,2}$	$\cdots$
$B_{2,0}$	$B_{2,1}$	$B_{2,2}$	$\cdots$
$B_{3,0}$	$B_{3,1}$	$B_{3,2}$	$\cdots$
$\vdots$	$\vdots$	$\vdots$	$\ddots$

všetky prvky množiny $B_{0,0}$, potom všetky prvky množiny $B_{1,0}$, $B_{0,1}$, $B_{2,0}$, $B_{1,1}$, $B_{0,2}$, $B_{3,0}$, atď. Tento proces nikde nebude mať problém s tým, že by neprešiel na ďalšie políčko, lebo všetky množiny sú konečné, tým pádom sa ku všetkým množinám dostaneme. V takomto poradí budeme priraďovať prvkom čísla $0, 1, 2, \ldots,$ čím sme našli bijekciu medzi $B, \mathbb{N}$, a teda B je spočítateľná.

Poznámka. Predošlý odsek by sa dal stručnejšie odargumentovať tak, že máme $|\mathbb{N}\times\mathbb{N}| = |\mathbb{N}|$ (čiže spočítateľný počet) konečných množín, a teda aj ich zjednotenie bude spočítateľné.

Úloha 4

(1,5 bodu) V triede je niekoľko žiakov a niektoré dvojice sa kamarátia (vzťah kamarátenia sa je symetrický). Každý žiak má aspoň jedného kamaráta. Celá trieda prišla na MatFyz, kde si každý žiak vyberie práve jednu z prednášok: buď matematickú, alebo informatickú. Dokážete, že (v každej takejto triede) sa žiaci vedia rozdeliť na dve prednášky tak, aby každý žiak Z mal aspoň jedného kamaráta, ktorý sa zúčastnil inej prednášky ako samotný žiak Z.

Bonus. *(1,5 boda)* Napíšte program, ktorý pre zadané kamarátstvá v triede vypíše rozdelenie študentov dve prednášky. Zdôvodnite (v pdf súbore alebo komentároch), prečo váš program vypíše správne rozdelenie. V prípade, že program píšete podľa dôkazu z úlohy 4, tak stačí len stručne opísať súvis s dôkazom. Pri dobrom zdôvodnení nie je veľmi dôležitá efektívnosť programu.

Formát vstupu. V prvom riadku vstupu sú dve medzerou oddelené čísla n (počet študentov) a m (počet študentov, ktorí sa kamarátia). Študentov číslujeme $0, 1, \ldots, n-1$. Nasleduje m ďalších riadkov, z ktorých každý obsahuje dve medzerou oddelené čísla študentov, ktorí sa kamarátia. Je zaručené, že tieto kamarátstva spĺňajú podmienky z úlohy 4.

Formát výstupu. Program vypíše na výstup dva riadky, každý z nich bude obsahovať niekoľko medzerou oddelených čísel študentov. V prvom riadku budú študenti, ktorí sa zúčastnia matematickej prednášky, v druhom riadku účastníci fyzikálnej. Program nesme vypísať nič iné (ani veci typu „Zadajte n, m:“).

Príklad vstupu	Príklad výstupu
5 6 0 1 0 2 0 3 1 2 2 3 3 4	0 1 4 3 2

Úlohu si preložíme do teórie grafov a v tomto jazyku budeme písať aj riešenia (v úlohe to však potrebné nebolo). Máme dokázať, že v každom graf s minimálnym stupňom aspoň 1 vieme rozložiť vrcholy na dve skupiny (množiny) tak, aby každý vrchol mal suseda v inej skupine.

konečne veľa.

Ukážeme viac riešení. K väčšine z nich aj uvedieme program, ktorý bude priamočiaro kopírovať uvedené myšlienky. Tieto programy aj ilustrujú, ako možno istý typ matematického dôkazu prepísať do počítačového programu. Táto úloha bola aj na to zvolená. Mnohé jej riešenia opisovali algoritmus, ako vrcholy rozdeliť do skupín. Netreba však zabúdať na to, že opis takéhoto algoritmu nie je úplný dôkaz. Dôležitou súčasťou je aj zdôvodniť, prečo náš algoritmus nájde rozdelenie na skupiny, aké bolo požadované v zadaní.

Riešenie cez prechod hrán

Budeme postupne prechádzať hranami a priraďovať vrcholy do skupín. Pre každú hranu uv vykonáme nasledovné:

(i) Ak žiaden z vrcholov u, v nie je priradený v skupine, tak vrchol u dáme do A_0 a vrchol v dáme do A_1.

(ii) Nech práve jeden z vrcholov priradený v skupine, bez ujmy na všeobecnosti, nech to je u a nech je v skupine A_i. Potom vrchol v priradíme do opačnej skupiny (teda A_{1-i}).

(iii) Ak oba vrcholy u, v sú už v skupine, tak nič nerobíme.

Pri tomto prechádzaní vrcholmi zabezpečujeme dôležitú vlastnosť:

$$\text{Ak má vrchol priradenú skupinu, tak má suseda v druhej skupine.} \tag{*}$$

Skupiny priraďujeme len v prípadoch (i) a (ii), kde vlastnosť (*) zjavne dodržujeme. Keďže každý vrchol je incidentný s aspoň jednou hranou, objaví sa pri spracovávaní hrán, a teda dostane skupinu. Keďže každý vrchol priradíme do skupiny a dodržíme vlastnosť (*), tak tým získame požadované rozdelenie.

Poznámka. Ak by sme chceli byť veľmi formálny, tak túto ideu možno spísať aj matematickou indukciou. Mohli by sme v nej dokázať nasledovné tvrdenie: „Pre každé celé $m \geq 0$ platí: pre každý graf s m hranami platí, že jeho vrcholy stupňa aspoň 1 možno rozdeliť do dvoch skupín tak, aby každý vrchol mal suseda v inej skupine ako je on sám.“

Program Podľa toho riešenia vieme napísať aj veľmi jednoduchý a efektívny program, ktorým vyriešime bonus. Skupiny pre vrcholy si budeme pamätať v zozname `group`, kde -1 znamená nenastavenú skupinu, 0 matematiku a 1 informatiku.

```python
n, m = map(int, input().split())
# skupina pre kazdy vrchol: 0 alebo 1, resp. -1, ak este nie je nastavena
group = [-1] * n
for i in range(m):
    u, v = map(int, input().split())
    if group[u] == -1 and group[v] == -1:
        # pripad (i)
        group[u] = 0
        group[v] = 1
    elif group[u] == -1:
        # pripad (ii) - iba vrchol u nie je v skupine: dame mu opacnu skupinu od v
        group[u] = 1 - group[v]
    elif group[v] == -1:
        # pripad (ii) - iba vrchol v nie je v skupine: dame mu opacnu od u
        group[v] = 1 - group[u]

# Teraz len vypiseme skupiny
print(*(v for v in range(n) if group[v] == 0))
print(*(v for v in range(n) if group[v] == 1))
```

Riešenie cez striedavé predávanie susedov

Tvrdenie dokážeme pre každý komponent súvislosti zvlášť. Ďalej nájdeme teda uvažovať komponent súvislosti K nášho grafu. Dve skupiny, na ktoré budeme rozdeľovať vrcholy, budeme označovať M a I.

Začneme s ľubovoľným vrcholom, označíme si ho v a dáme ho do množiny M. Množinu jeho susedov označíme S_1 a tých dáme do množiny I. Potom sa pozrieme na všetkých susedov vrcholov z množiny S_1, ktorí ešte nie sú zradení a tých označíme S_2 a pridáme ich do M. Takto budeme pokračovať ďalej, dokým neminieme všetky vrcholy uvažovaného komponentu súvislosti.

Presne vyjadrené, na začiatok uvažujeme množinu $S_0 = \{v\}$ pre ľubovoľne vybraný vrchol komponentu K. Pre celé $i > 0$ definujeme množinu S_i ako množinu všetkých vrcholov, ktoré susedia s nejakým vrcholom z množiny S_{i-1} a nenachádzajú sa v žiadnej predošlých množín $S_0, S_1, \ldots, S_{i-2}$. (Alternatívne možno S_i definovať ako množinu vrcholov, ktorých najkratšia cesta z nich do vrcholu v má dĺžku presne i.)

Vrcholy potom rozdelíme takto: $M = S_0 \cup S_2 \cup S_4 \cup \ldots$, $I = S_1 \cup S_3 \cup S_5 \cup \ldots$, teda do M pridáme vrcholy z množín S_i s párnym indexom, do I s nepárnym.

Teraz zdôvodníme, že toto rozdelenie spĺňa podmienku zo zadania: Vrchol v má stupeň aspoň 1, teda má suseda v množine S_1, ktorý je v inej skupine. Každý ďalší vrchol je spojený cestou s vrcholom v (keďže sme v komponente súvislosti), preto sa objaví v nejakej množine S_i ($i > 0$, lebo uvažujeme vrchol rôzny od v). Tam sa však objavil ako sused nejakého vrchola z S_{i-1}, čo je jeho sused z inej skupiny.

Program. Graf si reprezeuntujeme ako zoznam susedov – v zozname si pamätáme pre každý vrchol zoznam jeho susedov. Hľadanie rozdelenia robíme podľa opísaného riešenia, ako je vysvetlené komentármi. Z toho vyplýva správnosť programu. Množinu S_i reprezentujeme ako zoznam `S[i]` a všetky tieto zoznamy budú v zozname `S`.

```
n, m = map(int, input().split())
graph = [[] for v in range(n)]
for i in range(m):
    u, v = map(int, input().split())
    graph[u].append(v)
    graph[v].append(u)

group = [-1] * n
# Tymto cyklom zabezpecime prechod cez vsetky komponenty. Vzdy ked najdeme vrchol bez skupiny, tak
# ho zvolime ako pociatocny vrchol pre spracovanie komponentu. Potom vsetky vrcholy jeho komponentu
# uz budu mat nastavenu skupinu a pri dalsom spracovavani ich preskocime.
for v in range(n):
    if group[-1] != -1:
        continue

    group[v] = 0
    S = [[0]] # teda S[0] = [0]
    i = 0
    while len(S[i]) > 0:
        S.append([]) # pridame zoznam S[i + 1]
        # Pre kazdy vrchol u v zozname S[i] si prejdeme jeho susedov
        for u in S[i]:
            for sused in graph[u]:
                if group[sused] == -1:
                    # Pokial sused este nema skupinu, pridame ho do opacnej skupiny
                    group[sused] = 1 - group[u]
                    # a pridame ho do zoznamu S[i + 1], aby sme ho mohli spracovat v dalsom kroku
                    S[i + 1].append(sused)
        # Kedze ku kazdemu vrcholu sa raz dostaneme, tak niekedy pri spracovavani S[i] budu vsetky
        # vrcholy v nejakej skupine, teda S[i + 1] bude prazdne
        i += 1
```

Dvojrozmerný zoznam `S` sme použili, aby sme lepšie ilustrovali značenie v dôkaze. V programe sme si však nepotrebovali pamätať všetky „vrstvy“ susedov. V praxi sa na takéto účely používa jednoduchý zoznam, resp. ešte lepšie `dequeue` z modulu `collections`, do ktorého pridávame susedov na koniec a vyberáme zo začiatku, keď ich spracovávame.

Dôkaz cez kostru

Tvrdenie dokážeme pre každý komponent súvislosti zvlášť. Uvažujme teda komponent súvislosti K nášho grafu. Keďže každý vrchol v K má stupeň aspoň 1, tak K má aspoň dva vrcholy. Keďže je K súvislý, tak má kostru. Kostra je strom, teda neobsahuje cykly, špeciálne, neobsahuje cykly nepárnej dĺžky. Preto je kostra bipartitný graf. Označme jej partície ako A a B. Keďže kostra je súvislá a má aspoň dva vrcholy, tak každý vrchol z množiny A má suseda, ktorý z definície bipartitného grafu musí byť v partícii B. Rovnako aj každý vrchol z B musí mať suseda v A. Tým sme ukázali, že rozdelenie vrcholov na množiny A a B je naše hľadané rozdelenie, pri ktorom má každý vrchol suseda v druhej skupine.

Dôkaz matematickou indukciou

Tvrdenie dokážeme matematickou indukciou podľa počtu vrcholov grafu G, ktorý si označíme n. Teda presnejšie dokazujeme výrok: „Pre každé celé $n \geq 2$ platí: V každom grafe s minimálnym stupňom aspoň 1 možno rozdeliť jeho vrcholy do dvoch skupín tak, aby každý vrchol mal suseda v inej skupine ako je on sám.“

Báza. Pre $n = 0$, teda prázdny graf, tvrdenie zjavne platí (všeobecne kvantifikujeme cez prázdnu množinu vrcholov).

Indukčný krok Uvažujme teraz $n > 0$. Predpokladajme, že každý graf G s menej ako n vrcholmi a $\delta(G) \geq 1$ možno rozdeliť podľa zadania. Tvrdenie dokážeme teraz pre n. Nech H je graf, ktorý má n vrcholov a $\delta(H) \geq 1$. Nech v je vrchol grafu G s minimálnym stupňom. Ak graf $H - v$ (ktorý má $n - 1$ vrcholov) má minimálny stupeň aspoň 1, tak podľa indukčného predpokladu možno rozdeliť jeho vrcholy do dvoch skupín tak, aby každý jeho vrchol mal suseda v inej ako jeho skupine. Vrchol v má v grafe $H - v$ aspoň jedného suseda u. Na základe toho vrchol v dáme do inej skupiny, ako v ktorej je vrchol u. Tým zaručíme, že aj vrchol v má suseda v inej skupine ako je on sám.

Nech teda graf $H - v$ má minimálny stupeň 0 a nech tento stupeň 0 má vrchol u. Odstránením vrchola v z grafu H sme z vrchola u odstránili najviac jednu hranu. Preto vrchol u musel mať v grafe H stupeň 1. To znamená, že minimálny stupeň grafu H musí byť 1, a teda aj stupeň vrchola v je 1. To znamená, že v grafe H je vrchol v spojený len s vrcholom u. Preto vrchol u je jediný, ktorý má stupeň 0 v grafe $H - v$. Preto graf $H - \{v, u\}$ má minimálny stupeň aspoň 1 a $n - 2 \leq n$ vrcholov, a teda podľa indukčného predpokladu možno jeho vrcholy požadovane rozdeliť. Vrcholy u, v rozdelíme do dvoch rôznych skupín. Takto dostávame opäť vyhovujúce rozdelenie vrcholov na dve skupiny.

Ukázali sme, že v oboch prípadoch vieme rozdeliť vrcholy grafu H a tým je dôkaz indukciou ukončený.

Program. Tento dôkaz prepíšeme do programu, ktorý bude riešiť bonusovú úlohu. Graf reprezentujeme tak, že pre každý jeho vrchol si v slovníku pamätáme množinu jeho susedov. Priradenie do skupín reprezentujeme tiež slovníkom, ktorý každému vrcholu grafu priradí 0 (matematika) alebo 1 (informatika). Program sme napísali tak, aby čo najvernejšie kopíroval úvahy, ktoré sme robili v našom dôkaze. Preto správnosť tohto programu vyplýva z uvedeného dôkazu indukciou. Upozorňujeme však, že tento program je napísaný po viacerých stránkach neefektívne.

```
from copy import deepcopy

n, m = map(int, input().split())
graph = {v: set() for v in range(n)}
for i in range(m):
```

```python
    u, v = map(int, input().split())
    graph[u].add(v)
    graph[v].add(u)

# Funkcia, ktora dostane graf s min. stupnom aspon 1
# a vrati dict, ktory pre kazdy vrchol obsahuje skupinu 0 alebo 1 tak,
# aby kazdy vrchol mal suseda v inej skupine ako je on sam.
def divide(H):
    # Ak mame prazdny graf, tak netreba rozdelovat ziadne vrcholy
    if len(H) == 0:
        return {}

    # V opacnom pripade najdeme vrchol s min. stupnom
    v = None
    for x in H:
        if v is None or len(H[x]) < len(H[v]):
            v = x

    # Odstranime z grafu H vrchol v
    G = deepcopy(H)
    for x in G[v]:
        G[x].remove(v)
    del G[v]

    u = next(iter(H[v])) # lubovolny sused vrchola v
    if len(H[v]) == 1 and len(G[u]) == 0:
        # Pripad, kedy v je spojeny iba s jednym vrcholom: u
        # Odstranime aj vrchol v
        del G[u]
        # rekurzivne sa zavolame na mensi graf (s o 2 menej vrcholmi) = pouzitie IP
        solution = divide(G)
        # rozdelime vrcholy u a v
        solution[u] = 0
        solution[v] = 1
    else:
        # V opacnom pripade mame graf G s min. stupnom aspon 1,
        # mozeme sa rekurzivne zavolat (pouzit IP)
        solution = divide(G)
        # vrchol v dame do opacnej skupiny ako jeho sused v
        solution[v] = 1 - solution[u]

    return solution

solution = divide(graph)
print(*(v for v in solution if solution[v] == 0))
print(*(v for v in solution if solution[v] == 1))
```