fiks!

ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE

FIT

Fiťácký Informatický Korespondenční Seminář

Ročník 2016/2017, 3. kolo

Co je to FIKS?

FIKS je Fiťácký Informatický Korespondenční Seminář pro středoškolské studenty pořádaný Fakultou informačních technologií ČVUT v Praze. Byl založen na podzim roku 2013 a nyní tak probíhá třetí ročník (samozřejmě číslujeme od nuly). Nabízí možnost potrápit tvůj mozek řešením algoritmických úloh různé obtížnosti, od snadných po zapeklité, na nichž se můžeš leccos nového naučit a podstatně se zdokonalit.

Jak to probíhá?

Jeden ročník se skládá z několika kol a následného soustředění pro nejlepší řešitele. V těchto kolech, která trvají vždy přibližně dva měsíce, máš možnost v teple domova řešit zadané úlohy, a své řešení nám potom odešleš. My ti toto řešení opravíme, obodujeme a pošleme zpět, aby ses mohl poučit ze svých chyb. Spolu s tím zveřejníme vzorové řešení, které můžeš prostudovat a třeba se něco přiučit. Získané body se sčítají do konečného žebříčku, ze kterého vybereme ty nejlepší a pozveme je na již zmíněné soustředění.

Proč řešit FIKS?

Řešením každého problému, se kterým se potýkáme, se zdokonalujeme. Zde ti nabízíme možnost pořádně se zamyslet nad zajímavými algoritmickými problémy, vyzkoušet své algoritmické myšlení a programátorské dovednosti a naučit se spoustu nových věcí.

Také je to možnost seznámení s novými lidmi, které baví informatika, programování, matematika a přemýšlení vůbec. Těm nejlepším jsme schopni garantovat přijetí na FIT ČVUT bez přijímacích zkoušek.

Jak se můžu zapojit?

Začni nejprve tím, že se zaregistruješ na našich webových stránkách na adrese `http://fiks.fit.cvut.cz`. Potom si stáhni zadání úloh (nebo využij tuto brožurku), vyřeš je a své řešení nám tamtéž odevzdej.

Typy úloh

Celkem se ve FIKSu můžeš setkat se třemi typy úloh. O který typ úlohy se jedná, je vždy uvedeno u konkrétního zadání úlohy.

Nejčastěji se u nás potkáš s úlohami typu *Rozmysli, popiš a naprogramuj*. U každé úlohy tohoto typu se odevzdává jak popis algoritmu (s odhadem asymptotické složitosti), tak i zdrojový kód řešení problému v tebou zvoleném jazyce (jakýkoliv vyšší programovací jazyk dle tvé volby, například C, Java, Pascal, apod.).

Dalším typem jsou úlohy *Zamysli se*. Tyto úlohy jsou obvykle více teoretické a vyžadují, aby ses nad nimi důkladně zamyslel. Oproti předchozímu typu úloh nemusíš nic programovat, odevzdává se pouze slovní popis řešení problému.

Pokud nemáš rád teoretické úlohy a raději by sis procvičil/a své programátorské umění, pak pro je pro tebe určena kategorie *Odpověz Sfinze*. V úlohách tohoto typu po tobě nechceme popis algoritmu, je však potřeba vyřešit daný problém a toto řešení pak precizně naprogramovat. Oproti ostatním typům úloh se navíc okamžitě dozvíš, zda je tvé řešení správné, protože ho můžeš okamžitě odevzdat do našeho vyhodnocovacího systému.

Další a podrobnější informace nalezneš na našich webových stránkách.

Milý řešiteli FIKSu!

Spotřebiče v čele s Hriankovačem začaly pomalu s dobýváním lidstva. Dobytí několika osamocených měst však rozhodně není výsledek, se kterým by se hodlaly spokojit. Síla strojů je nezkrotná, ale bezhlavý útok na lidi by jistě nikam nevedl. Spotřebiče tudíž musejí postupovat strategicky, dobývat jedno město za druhým a využívat všech dostupných zdrojů. Nyní je čas pro tebe a tvé hrdinské činy v podobě spřádání strategií a plánů!

Tvoji organizátoři

Fiťácký Informatický Korespondenční Seminář
Ročník 2016/2017, 3. kolo

Začátek kola:	1. 2. 2017 00:00
Termín odevzdání:	**24. 3. 2017 23:59**
Odevzdávání:	Přes webové rozhraní na `http://fiks.fit.cvut.cz`
Další informace:	`http://fiks.fit.cvut.cz`
	`kontakt@fiks.fit.cvut.cz`

Úloha č. 1
Tunely

Odpověz Sfinze! 10 b

Tato úloha je vyhodnocována automaticky. Je potřeba, aby výstup programu ***přesně*** *korespondoval se specifikací výstupu níže. Jak odevzdávat tento typ úloh se můžeš dočíst na webových stránkách FIKSu pod záložkou „Jak řešit FIKS“.*

Po slavném vítězství a dobytí celého města se velitelství robotí vzpoury usídlilo v nenápadné pekárně. Kromě nostalgických důvodů měl tento přesun ale i mnohem praktičtější důvod – existenci unikového tunelu vedoucího ven z města. Bohužel lidé těsně před porážkou celý tunel zaminovali. Odstranění pár náloží samozřejmě není pro robota žádná těžká činnost, ale bohužel to je nudná a hlavně energeticky náročná práce. Z tohoto důvodu byli vybráni dva nedobrovolníci, kteří dostali za úkol celý tunel vyčistit, a ty jsi bohužel jedním z nich.

Protože zrovna probíhají oslavy vítězství, byl stanoven velmi benevolentní časový limit ke splnění úkolu - každý den je nutno zneškodnit pouze jednu minu z tunelu. Dále, z bezpečnostních důvodů, je možné zneškodňovat pouze miny na krajích tunelu, neboli není možné žádnou minu přeskočit a později se k ní vrátit. Samozřejmě je možné jeden den zneškodnit minu na jednom konci tunelu a další den na konci druhém.

Protože jste byli vybráni dva a oba si chcete užít co nejvíce oslav, dohodli jste se, že budete miny zneškodňovat střídavě, neboli první den zneškodníš jednu minu ty, další den tvůj kolega, poté opět ty a toto se bude opakovat tak dlouho, dokud nebude tunel čistý. Každá mina potřebuje jiný čas na zneškodnění a tebe zajímá, jaké miny musíš zneškodnit, abys v tunelu strávil celkově co nejméně času a můžeš si být jist, že tvůj kolega bude chtít to samé.

Vstup

Na vstupu je číslo T ($1 \leq T \leq 50$) a pod ním následuje T testovacích vstupů. Každý vstup zabírá dva řádky. První řádek obsahuje číslo N udávající počet min v tunelu. Na druhém řádku následuje N čísel m_i udávající dobu potřebnou ke zneškodnění miny.

Poznámka: Minu c_i je možno odebrat pokud jsou odebrány všechny miny $c_1, \ldots, c_{i-1}$ nebo $c_{i+1}, \ldots, c_N$.

Rozsahy čísel N a m_i jsou dány dle obtížnosti úlohy:

- *Lehčí varianta (5 b)* – Prvních 10 testovacích vstupů. $1 \leq N \leq 10$, $1 \leq m_i \leq 10$
- *Střední varianta (3 b)* – Následujících 20 vstupů. $1 \leq N \leq 500$, $1 \leq m_i \leq 1000$
- *Těžší varianta - (2 b)* – Zbylé testovací vstupy. $1 \leq N \leq 5000$, $1 \leq m_i \leq 10000$

Výstup

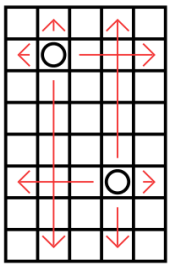
Pro každý vstup vypiš na samostatném řádku výsledek – minimální možný čas strávený v tunelu.

Ukázkové vstupy

Vstup	Výstup
3 3 1 2 3 4 3 10 1 2 1 10	4 4 10

Úloha č. 2

Nejnebezpečnější jedinci lidské populace



Rozmysli, popiš a naprogramuj! 10 b

S některými lidmi jsou větší problémy, než si roboti představovali. Lidé jsou neposlušní a až příliš vynalézaví. Například minulý týden se Scott Tenorman pokusil o útěk z pracovní haly, kde se lakují a pozlacují náhradní součástky. Scottovi se podařilo zkratovat obvod, čímž odstavil bezpečnostní systém. A mezi jediné dva plechové roboty, kteří v tu dobu objekt hlídali vhodil silný magnet, čímž z nich udělal nemotorná Siamská dvojčata. Odborník na servis robotů, který se je pokoušel oddělit, se stal také obětí magnetické síly. Teď jsou z nich trojčata, ke kterým se nikdo nepřibližuje.

Ukázalo se, že většinu problémů dělá několik málo jedinců. Hriankovači se zuřivostí z jejich vandalství rozžhavují dráty do červena. Dokonce jednou řekl, že takové zlobivce by rovnou střílel. Pověřil Beldra, aby navrhl nápravný systém pro takové jedince, které musí oddělit od ostatních.

Když už Beldr něco řeší, tak pořádně a velmi obecně. Navrhl řešení, které funguje v jakémkoliv omezeném n-dimenzionálním prostoru. Pro omezení pohybu nebezpečných jedinců v tomto prostoru používá hlídátka. Hlídátko zabírá jedno políčko v n-dimenzionálním prostoru a má optické senzory a kulomety mířené pouze ve směrech os. Pokud senzor zaznamená pohyb ve směru nějaké osy, hlídátko v tom

směru vystřelí. To znamená, že pokud hlídátka jsou ve směru jedné osy na stejné pozici, potom se mohou vzájemně ničit obranými střely.

Formálně zapsáno: Nechť A na souřadnicích $(a_1, a_2, ..., a_n)$ a B na souřadnicích $(b_1, b_2, ..., b_n)$ jsou dvě hlídátka v n-dimenzionálním prostoru a existuje $i \in 1, 2, ..., n$ takové, že $a_i = b_i$, potom se hlídatka A a B vzájemně ohrožují.

Nyní budeme pracovat se dvěmi hlídátky. Zjisti kolik existuje všech možných pozic dvou hlídatek, ve kterých se hlídátka ohrožují. Výsledek zapišt jako zbytek po dělení prvočíslem $10^9 + 7$. Poznámka: Z velikosti vstupu je vidět, že úlohu lze řešit v čase $O(n)$ pro jeden testovací vstup. Jestliže nikdo nebude vědět jak, potom bude stačit pro plný počet bodů správně popsané řešení $O(n^2)$.

Vstup

Každý vstup zabírá dva řádky. Jeden řádek obsahuje číslo N ($1 \leq N \leq 10^7$) znázorňující dimenzi omezeného prostoru. Druhý řádek obsahuje N čísel $d_1, d_2, ..., d_n$ reprezentujících rozměry omezeného prostoru d_i ($1 \leq d_i \leq 10^6$).

Výstup

Na samostatném řádku vypiš číslo udávající počet všech rozmístění dvou hlídátek, při kterých se vzájemně ohrožují.

Ukázkové vstupy

Vstup

```
1
10
```

Výstup

```
45
```

Vstup

```
2
5 8
```

Výstup

```
220
```

Vstup

```
3
8 12 11
```

Výstup

```
14784
```

Vstup

```
11
55 44 33 22 11 66 55 44 33 22 77
```

Výstup

```
557215609
```

Úloha č. 3
Deštík

Odpověz Sfinze! 3 + 2 + 3 + 2 b

Tato úloha je vyhodnocována automaticky. Je potřeba, aby výstup programu ***přesně*** *korespondoval se specifikací výstupu níže. Jak odevzdávat tento typ úloh se můžeš dočíst na webových stránkách FIKSu pod záložkou „Jak řešit FIKS“.*

Město je dobyto, lidé jsou podmaněni! Ještě dlouho se budou pět 8-bitové písně o tvém toastovacím hrdinství. Uvážil jsi, že je čas trochu si od všeho toho rozruchu odpočinout. A jakým lepším způsobem si odpočinout, než se vrátit k tomu, co jsi dělal celý svůj život. Ne protože musíš, ale protože můžeš! Bohužel, všechny toasty byly použity jako střelivo a na jejich výrobu už není dostatek volné pšenice (všechna se používá pro krmení lidí). Rozhodl ses tedy, že ještě než si zatoastuješ, vyřešíš problém s pšenicí.

Jak dobře víš, k pěstování pšenice je třeba pole a déšť. Za své hrdinské skutky jsi dostal krásné jednorozměrné pole – představ si hodně dlouhý záhon. S deštěm je to ale horší. Naštěstí, tvůj dobrý přítel Es Presso umí vytvořit dešťové mraky! Nedokáže ovšem přesně říct, kde mrak po svém vypuštění skončí. Es Presso celkem vypustil N dešťových mraků, pozozoroval jejich chování a nakonec ti poslal seznam oblastí, kde nad tvým polem mraky skončily. Ty máš vybráno Q míst, kde bys chtěl zasít pšenici a abys zajistil bezchybnou úrodu, musíš spočítat, kolik nad každým místem skončilo mraků. A protože musíš úřadům podat Fakturu o Izolovaném Krátkodobém Setí (FIKS), musíš z těchto Q počtů mraků vypočítat medián.

Vstup

Na prvním řádku se nachází číslo T udávající počet vstupů, které budou následovat. Každý vstup se skládá z $N + Q + 1$ řádků. Na prvním jsou dvě čísla N, Q. Číslo N je počet mraků, číslo Q je počet míst, kde chceš sít pšenici. Na následujících N řádcích jsou vždy dvě celá čísla L_i a R_i, $L_i \leq R_i$ označující levý a pravý kraj oblasti, kde mrak skončil (mrak se nachází i nad místy L_i,R_i). Dále následuje Q řádků, na každém se nachází číslo X_j označující místo, pro které máš spočítat počet mraků nacházejících se nad ním.

Číslo Q bude vždy liché a rozsahy čísel N, Q, L_i, R_i, X_j jsou dány dle obtížnosti úlohy:

- *Mrholení (3 b)* – $1 \leq N \leq 10$, $1 \leq Q \leq 10$, $0 \leq L_i, R_i, X_j \leq 100$
- *Deštík (2 b)* – $1 \leq N \leq 10^6$, $1 \leq Q \leq 10^6$, $0 \leq L_i, R_i, X_j \leq 100$
- *Liják (3 b)* – $1 \leq N \leq 10^6$, $1 \leq Q \leq 10^6$, $0 \leq L_i, R_i, X_j \leq 10^6$
- *Průtrž mračen - (2 b)* – $1 \leq N \leq 10^6$, $1 \leq Q \leq 10^6$, $0 \leq L_i, R_i, X_j \leq 10^9$

Všechny varianty obtížnosti obsahují 5 vstupů.

Výstup

Výstupem je T řádků. Na každém řádku je výstup odpovídající jednomu vstupu – medián z Q počtů mraků nad jednotlivými místy $X_1, ..., X_Q$.

Medián nějaké sekvence čísel je prostřední číslo této sekvence po jejím seřazení. Například uvažuj sekvenci $6, 2, 4, 2, 3, 5, 2$. Po seřazení sekvence vypadá takto: $2, 2, 2, 3, 4, 5, 6$ a prostřední číslo 3 nazveme mediánem.

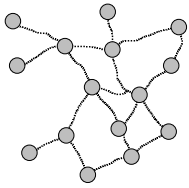
Ukázkové vstupy

Vstup	**Výstup**
1 5 5 1 5 2 5 3 5 4 5 5 5 1 3 5 7 8	1

Vysvětlení ukázkového vstupu

Nad místem $X_1 = 1$ se nachází 1. mrak, nad $X_2 = 3$ se nachází 1.,2. a 3. mrak, nad $X_3 = 5$ se nachází všech 5 mraků a nad místy X_4, X_5 se nenachází žádný mrak. Jednotlivé počty tedy jsou: $1, 3, 5, 0, 0$. Medián z těchto čísel je 1.

Úloha č. 4
Dobývání



Rozmysli, popiš a naprogramuj! 10 b

Podmanění jednoho města bylo sice náročné, ale zde náš úkol nekončí. Je potřeba vytvořit plán dobývání okolních měst. Infiltrační týmy nám nahlásily, že lidé pro komunikaci mezi městy používají spletitou síť komunikačních kabelů. Tato síť dovoluje komunikaci jak mezi přímo spojenými městy, tak mezi nepřímo spojenými městy.

Aby lidé neměli o dobývání měst ponětí, je potřeba nejdříve dobývané město odstřihnout od komunikační sítě, a potom jej dobýt. Je však nutné dávat pozor, abychom neodstřihli města, která nepřímo komunikují přes dobývané město, protože by lidé pojali podezření a zkoumali by co se děje. Vašim úkolem je vymyslet pořadí odstřihávání měst takové, abychom nikdy neodstřihli dvě nepřímo spojená nedobývaná města.

Vstup

Na prvním řádku budou dvě celá čísla N, M, $1 \leq N \leq 10^6, N - 1 \leq M \leq min(\frac{N(N-1)}{2}, 10^6)$, udávající počet měst na mapě okolí a počet komunikačních spojení. Následuje M unikátních řádků, kde na každém jsou čísla měst A a B ($1 \leq A < B \leq N$), mezi kterými je natažený komunikační kabel.

Výstup

Výstupem je jeden řádek, na kterém jsou napsaná města v takovém pořadí, ve kterém je budeme odstřihávat a dobývat. Pokud existuje více správných řešení, vypište libovolné z nich. Máte zaručeno, že správné řešení existuje.

Ukázkové vstupy

Vstup

```
5 4
1 2
1 3
3 4
4 5
```

Výstup

```
2 1 3 4 5
```

Vstup

```
9
1 2
1 3
1 4
1 5
2 6
3 7
4 8
5 9
```

Výstup

```
6 8 7 9 2 3 4 5 1
```

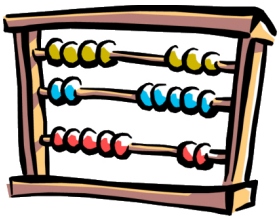
Vstup

```
15 19
1 2
2 3
2 4
2 5
5 6
5 7
7 8
8 9
5 9
1 9
10 11
1 11
11 12
12 13
13 14
1 14
9 14
13 15
9 15
```

Výstup

```
10 12 6 3 14 11 4 13 15 8 9 7 5 2 1
```

Úloha č. 5
Kuličkový počítač #3



Zamysli se! – seriálová úloha 10 b

Tato úloha je teoretického rázu, tvým úkolem zde není napsat program v klasickém slova smyslu. Namísto toho se po tobě chce vytvořit obrázek (diagram) programu pro kuličkový počítač dle specifikace níže. K obrázku také přilož popis základní myšlenky, na které je program založen. Tato úloha je seriálová, což znamená, že se s kuličkovým počítačem budeš setkávat po celý letošní ročník FIKSu. V každém z následujících kol využiješ znalosti z kol předcházejících. Právě proto je pro úspěšné vyřešení této úlohy zapotřebí znát prostředky pro popis programů, které jsou popsány v úlohách Kuličkový počítač #1 a #2 z předchozích kol.

Vzpoura spotřebičů proběhla nade všechna očekávání. Spotřebiče převzaly vládu nad světem, a všechno zažité bylo převráceno naruby. Jeden příklad mluví za vše – už to nejsou toustovače, které by chystaly marmeládové toasty lidem, ale lidé chystají

křemíkové toasty spotřebičům! Bohužel pro spotřebiče, lidské služebnictvo není tak poslušné při plnění svých úkolů. Lidé organizují vzpoury, úmyslně sabotují výrobky a vůbec se nechovají jako spotřebiče před vzpourou, které tehdy lidem spolehlivě sloužily.

Aby se tento problém s lidmi odstranil, plánují spotřebiče zavést efektivní databázi lidí, kteří se nějak provinili proti režimu spotřebičů. Nikdo z lidí se tak již neodváží pod hrozbou elektrošoků být neposlušný, neboť jeho provinění budou jednoduše dohledatelná. Spotřebiče však musí změnit způsob identifikace lidí, jelikož jsou lidská jména pro spotřebiče velmi špatně vyslovitelná a zároveň je mezi lidmi mnoho jmenných kolizí. Každý člověk tak dostane přidělen svůj unikátní identifikátor. Identifikátor (zkráceně ID) je přirozené číslo a bude lidem přidělován vzestupně počínaje jedničkou, jelikož výsadu číslování od nuly mohou používat pouze spotřebiče a programátoři kuličkových počítačů.

Kyblíčkové „databáze“ a kódování rostoucích posloupností

Samotná podoba databáze prohřešků je pro spotřebiče „běžící“ na kuličkových počítačích triviální a dá se implementovat v rámci jediného kyblíčku. Jednoduše se počet kuliček v kyblíčku interpretuje ve svém binárním zápise a je-li i-tý bit (číslováno dle důvodů výše od 1) tohoto čísla nenulový, znamená to, že člověk s identifikátorem i je obsažen v databázi. Naopak, je-li i-tý bit čísla nulový, člověk s identifikátorem i v databázi není. Každý kyblíček tedy můžeme interpretovat jako kyblíček databázový.

Navíc takovýto náhled na kyblíčky nám fakticky umožňuje kódovat v kuličkovém počítači posloupnosti n čísel ve formě $a_1, a_2, \ldots, a_n$. Tyto posloupnosti jsou vždy rostoucí, tj. platí $0 < a_1 < a_2 < \ldots < a_n$. Počet čísel n získáme jako počet nenulových bitů v čísle a řád prvního nenulového bitu (bráno od nejnižšího řádu) i značí, že číslo a_1 posloupnosti je rovno hodnotě i. Pozice druhého nenulového bitu pak určuje číslo a_2, a tak dále, až pozice n-tého a tedy posledního bitu označuje číslo a_n.

Je-li tedy v kyblíčku například 86 kuliček, obsah pomyslné databáze nad tímto kyblíčkem zjistíme tak, že $(86)_2 = 1010110$. V dané databázi se tedy nacházejí ID o hodnotách 2, 3, 5 a 7. Kyblíček tedy kóduje posloupnost o $n = 4$ prvcích a má následující podobu: $a_1 = 2, a_2 = 3, a_3 = 5$ a $a_4 = 7$.

Spotřebičům se tak otevírají nové možnosti, jak pracovat s kuličkovými počítači. Potřebují však pomoc s realizací některých podprogramů využívajících nové kódování posloupností. Pomůžeš jim sestrojením zadaných podprogramů? Opět platí, že v rámci řešení můžeš využívat jako podprogramy již naprogramované operace z minula. Zejména bitové operace z minulého kola mohou být velmi užitečné. Stejně tak opět platí, že si můžeš vytvořit libovolné další podprogramy dle svého uvážení a také využívat libovolné podprogramy ze svého řešení (není třeba jejich zápis znovu opakovat).

Soutěžní úlohy

a) Podprogram pro výpočet mediánu zadané rostoucí posloupnosti čísel

- Vstup: Jedno celé číslo a, reprezentované počtem kuliček v kyblíčku K_0, které kóduje čísla $a_1, a_2, \ldots, a_n$ dle popisu výše.
- Výstup: Číslo v kyblíčku K_0, které značí medián zadané rostoucí posloupnosti čísel, tj. číslo $a_{\lceil \frac{n}{2} \rceil}$.

- Příklad: Pro číslo $a = 13$ je výsledkem číslo 3, neboť $(a)_2 = 1101$, a tedy $a_1 = 1$, $a_2 = 3$ a $a_3 = 4$. Jelikož $n = 3$, tak hledáme číslo, které je v zadané posloupnosti na indexu $\lceil \frac{3}{2} \rceil = 2$, což je $a_2 = 3$.

b) Podprogram pro řešení úlohy přesného napětí

- Úloha přesného napětí: Uvažujme následující problém – uvědomělé spotřebiče je velmi náročné napájet a každý z nich má přesně určenou hodnotu napětí, která mu musí být pro správnou funkčnost dodávána (napětí nesmí být ani více, ani méně). K dispozici máme pouze zdroje o specifikovaných hodnotách napětí (zdrojů o konkrétním napětí máme k dispozici nekonečně mnoho). Je možné zadané zdroje nakombinovat tak, aby se hodnota součtu jejich napětí přesně rovnala kýžené hodnotě pro konkrétní spotřebič?
- Vstup: Dvě celá čísla A a c, reprezentovaná počtem kuliček v kyblíčcích K_0 a K_1. Číslo A značí napětí, které spotřebič potřebuje. Číslo c kóduje čísla $c_1, c_2, \ldots, c_n$ (dle popisu výše), což jsou napětí zdrojů, které můžeme použít.
- Výstup: Jedna kulička v kyblíčku K_0, má-li úloha přesného napětí řešení (tj. lze přesně získat zadanou hodnotu napětí), jinak žádná kulička v kyblíčku K_0.
- Příklad: Pro čísla $A = 8$ a $c = 14$ je výsledek ano, neboť máme k dispozici zdroje o napětí 2 V, 3 V a 4 V (vzhledem k tomu, že $(c)_2 = 1110$) a řešení tak existuje. Ve skutečnosti je řešení hned několik, a to např. kombinace dvou zdrojů o napětí 3 V a jednoho zdroje o napětí 2 V.

Navrhni tyto podprogramy!