Korespondenční Seminář z Programování

ZAČÁTEČNICKÁ KATEGORIE

37. ročník **KSP-Z** **Duben 2025**

Právě se díváte na leták páté série 37. ročníku **KSP-Z**, neboli **K**orespondenčního **S**emináře z **P**rogramování, **Z**ačátečnické kategorie. Zapojit se může **každý středoškolák i základoškolák**.

Letos bude v KSP-Z **pět sérií** po **čtyřech úlohách** za celkem **220 bodů**. Pokud budete mít jakoukoliv otázku, neváhejte se zeptat. Kontaktní adresy najdete v patičce na konci letáku. Přejeme hodně štěstí!

Termín série: **neděle 25. května ve 32:00** (tedy další ráno v 8:00), praktické úlohy za třetinu bodů až do 1. června

Obsah série: **3 praktické úlohy** (značené 🖩) – K těmto úlohám je nutné napsat program (v libovolném vhodném jazyce), stáhnout si z našeho webu vstupní data a odevzdat odpovídající výstup. **1 teoretická úloha** (značená 💡) – U této úlohy nás zajímá hlavně slovní popis řešení, ve kterém byste měli zdůvodnit jeho funkčnost a ideálně nás i přesvědčit o jeho efektivitě.

Odevzdávání: Přes web na adrese *https://ksp.mff.cuni.cz/z/odevzdavatko/*

Zadání páté série začátečnické kategorie 37. ročníku KSP

🖩 37-Z5-1 Tramvaj — 8 bodů

Kevin se pozdě večer vrací domů z jubilejního Hrošího candrbálu. Stejně jako ostatní návštěvníci už stíhá jen poslední spoj. Protože ví, že tramvaj bude plná lidí, rozhodne se místo čekání vydat po stejné trase napřed. Kevin ovšem bydlí až na konečné, takže by se rád alespoň kousek svezl. Pomůžete mu zjistit, na jaké zastávce musí nastoupit, pokud se chce projít co nejdéle?

Zastávky jsou označené čísly od 1 do N. Tramvaj vyjíždí z depa, stanice 0, zatímco Kevin ve stejném okamžiku vyráží ze zastávky číslo Z. Známe časové intervaly přejezdu tramvaje mezi jednotlivými zastávkami a také víme, že Kevin chodí K-krát pomaleji. Nastupování zabírá nulový čas. Speciálně, pokud Kevin a tramvaj dorazí do nějaké zastávky ve stejný čas, tak zde Kevin tramvaj ještě stíhá.

Toto je praktická open-data úloha. V odevzdávacím systému si necháte vygenerovat vstupy a odevzdáte příslušné výstupy. Záleží jen na vás, jak výstupy vyrobíte.

Formát vstupu: Na prvním řádku jsou tři přirozená čísla N, Z a K – počet zastávek, počáteční zastávka Kevina a násobek jeho zpomalení. Na dalších N řádcích jsou časy přesunů tramvaje mezi navazujícími zastávkami. První čas značí dobu přesunu z depa do první zastávky, následující pak dobu přesunu ze zastávky 1 do zastávky 2 atd.

Slibujeme, že Kevin určitě nestihne dojít na konečnou dříve, než by tam dojela tramvaj.

Formát výstupu: Na výstup vypište číslo zastávky, kde Kevin musí nejpozději nastoupit.

Ukázkový vstup:

```
8 3 2
4
6
2
2
3
5
3
4
```

Ukázkový výstup:

```
6
```

Na šestou zastávku Kevin dojde za $(2 + 3 + 5) \cdot 2 = 20$ jednotek času, kdežto tramvaj tam bude v čase 22. Pokud by se Kevin rozhodl jít ještě o zastávku dál, došel by tam v čase 26, ovšem tramvaj by mu už v čase 25 před nosem ujela.

🖩 37-Z5-2 Závorky — 10 bodů

Před měsícem si Kevin objednal robota, který mu bude vyhodnocovat aritmetické výrazy. Dneska ten robot konečně dorazil a Kevin, plný nedočkavosti, mu hned zadal nějaký výraz, a... Ach ne, dostal error, že výraz obsahuje špatné uzávorkování.

Kevin se znovu koukl na napsaný výraz, ale přijde mu, že všechno je v pořádku. Počkat, našel místo, kde omylem napsal pravou závorku místo levé, a taky má tři levé závorky na konci výrazu. Ve výrazu objevil tolik chyb, že zahodil všechna čísla a zaměřil se pouze na závorky. Teď se snaží připsat závorky tak, aby uzávorkování opravil.

Kevina už ale z psaní všech těch závorek bolí ruka, tudíž jich chce doplnit co nejméně. Pomůžete mu?

Správné uzávorkování je sestavené z dvojic levých a pravých závorek, kde pro danou dvojici platí, že levá závorka se objevuje před pravou. Příkladem *správného uzávorkování* je třeba `()(())` nebo `((()(())))`. Příkladem *špatného uzávorkování* je naopak `())(())` nebo `((((`, ale první špatné uzávorkování lze doplnit na `(())(())` nebo `()((()))`, v obou případech přidáním jedné levé a jedné pravé závorky.

Toto je praktická open-data úloha. V odevzdávacím systému si necháte vygenerovat vstupy a odevzdáte příslušné výstupy. Záleží jen na vás, jak výstupy vyrobíte.

Formát vstupu:

Na prvním řádku dostanete číslo N, které značí počet závorek na vstupu. Na druhém řádku dostanete nějaké uzávorkování obsahující N kulatých závorek.

Formát výstupu:

Jako výstup vypište nějaké správné uzávorkování, které je úpravou vstupního uzávorkování. Všechny původní závorky musíte ponechat, smíte však přidat nové závorky kamkoli do výrazu. Hledáme takové řešení, které přidá závorek co nejméně.

Ukázkový vstup:

```
7
)(()))(
```

Ukázkový výstup:

```
(()(()))()
```

37-Z5-3 Krabice na pásech — 12 bodů

Kevin se rozhodl, že půjde na brigádu do továrny Krabice, Sáčky, Paklíky. Továrna má mnoho pásů, po kterých se pohybují krabice. V krabicích se nacházejí různé objednávky (typicky menší krabice, sáčky nebo paklíky). Někdy se stane, že si zákazníci změní objednávku, a danou krabici je potom třeba najít a přebalit. Kevin dostal za úkol zjistit pozici červené krabice, o které ví, že před T sekundami nastoupila na pás na pozici r_K, s_K. Pomůžete mu?

Toto je praktická open-data úloha. V odevzdávacím systému si necháte vygenerovat vstupy a odevzdáte příslušné výstupy. Záleží jen na vás, jak výstupy vyrobíte.

Jako vstup dostanete mapu továrny s orientací pásů. Každý pás směřuje buď nahoru (`^`), dolů (`v`), doleva (`<`) nebo doprava (`>`). Vaším úkolem je pro zadané počáteční souřadnice vstupu krabice na pás a číslo T zjistit, kde bude krabice po T sekundách. Krabice se přitom z jednoho pásu na druhý přesune přesně za jednu sekundu.

Čas T může být velmi velký a nemusí se vejít do *32-bitového číselného typu.* Pokud programujete v Pythonu, pak toto řešit nemusíte. Pokud ale programujete v C++, doporučujeme ukládat čas v číselném datovém typu `long long`.

Formát vstupu: První řádek obsahuje dvě čísla R a S – počet řádků a sloupců mapy továrny. Na dalším řádku najdete 3 hodnoty r_K, s_K a T, které označují řádek a sloupec krabice v čase 0 a čas T, ve kterém chcete zjistit její pozici. Levý horní roh továrny leží na souřadnicích $0, 0$.

Dále následuje R řádků, na kterých je S znaků reprezentujících orientace pásů v továrně.

Formát výstupu: Na jeden řádek vypište dvě čísla r_T a s_T označující řádek a sloupec továrny, kam krabice doputuje po T sekundách z počáteční pozice r_K, s_K. *Máte zaručeno, že krabice nikdy neopustí továrnu.*

Ukázkový vstup:

```
4 5
0 0 11
>v>v^
^v<<^
>v>^v
>>^vv
```

Ukázkový výstup:

```
2 1
```

37-Z5-4 Vedení elektřiny — 14 bodů

Okolí Hrošího Týnce bohužel zastihla bouřka, která poničila elektrickou síť. V okolí je N měst a v některých je postavená elektrárna. Dále bouřku přežilo M vedení, každé mezi dvěma městy. Město je propojeno s elektrárnou, pokud existuje posloupnost navazujících vedení, která začíná v daném městě a končí v elektrárně (posloupnost také může být prázdná, pokud se v městě rovnou nachází elektrárna). Kam máme vedení postavit, aby každé město bylo propojeno s nějakou elektrárnou, a zároveň postavených vedení bylo co nejméně?

Toto je teoretická úloha. Není nutné ji programovat, odevzdává se pouze slovní popis algoritmu. Více informací zde: `http://ksp.mff.cuni.cz/viz/tinfo`

Formát vstupu: Na prvním řádku dostanete tři čísla N, M, E – počet měst, počet vedení a počet měst s elektrárnou.

Na druhém řádku dostanete E čísel – čísla měst s elektrárnou.

Na dalších M řádcích dostanete dvojice čísel u, v říkající, že mezi městy u a v vede vedení.

Formát výstupu: Za každé vedení, které je zapotřebí postavit, vypište na řádek dvojici čísel u, v – čísla měst, mezi kterými dané vedení má vést.

Počet těchto vedení musí být minimální možný.

Ukázkový vstup:

```
6 2 2
1 3
1 2
4 5
```

Ukázkový výstup:

```
1 4
3 6
```

Na začátku jsou propojená s elektrárnou města 1, 2 a 3. Po postavení dvou vedení jsou města 4 a 6 hned vedle elektrárny, město 5 je propojené skrze 4. Lze dokázat, že při postavení méně než 2 vedení bude existovat město bez elektřiny.

Praktický kurz programování

Pokud Tě lákají praktické úlohy, ale ještě neumíš žádný programovací jazyk, můžeš se podívat na náš Základní kurz programování, kde se můžeš naučit základy Pythonu: *https://ksp.mff.cuni.cz/kurz/* .

Zdrojáky praktických úloh

Řešení praktických úloh může být ze začátku složité. Velmi často i nějaká triviální technická chyba ve zdrojovém kódu programu může znamenat, že program vrací špatný výsledek – a některé chyby se ze začátku špatně hledají. Proto Ti nabízíme možnost poslat zdrojový kód programu nějaké úlohy na adresu *zdrojaky@ksp.mff.cuni.cz*, kde se Ti pokusíme poradit. Do emailu prosím připiš:

- Jakou úlohu by měl program řešit.
- Slovní popis, co by měl program podle Tebe dělat.

Před termínem série Ti nemůžeme radit s algoritmem, ale pomůžeme s odladěním zdrojáku. Po termínu série pak můžeme poradit i s návrhem algoritmu – získáš tak znalosti do dalších sérií.

matfyz

KSP pro vás připravují studenti Matematicko-fyzikální fakulty Univerzity Karlovy.
Realizace projektu byla podpořena Ministerstvem školství, mládeže a tělovýchovy.

Webové stránky: *https://ksp.mff.cuni.cz/*
E-mail: *ksp@mff.cuni.cz*
Organizátoři a kontakty: *https://ksp.mff.cuni.cz/kontakty/*