

# Korespondenční Seminář z Programování

## ZAČÁTEČNICKÁ KATEGORIE

38. ročník



KSP-Z

Leden 2026

Právě se díváte na leták třetí série 38. ročníku **KSP-Z**, neboli **Korespondenčního Semináře z Programování**, **Začátečnické** kategorie. Zapojit se může **každý středoškolák i základoškolák**.

V průběhu roku vydáme několik dalších sérií úloh podobných této. Pokud budete mít jakoukoliv otázku, neváhejte se zeptat. Kontaktní adresy najdete v patičce na konci letáku. Přejeme hodně štěstí!

**Termín série:** **neděle 8. února ve 32:00** (tedy další ráno v 8:00), praktické úlohy za třetinu bodů až do 15. února

**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 třetí série začátečnické kategorie 38. ročníku KSP

#### **38-Z3-1 Osobní pokladna** **8 bodů**

Kevin má rád Sárú a Sářina maminka pracuje jako vlakvedoucí v Polárním expresu. Jako každá správná vlakvedoucí má u sebe svoji osobní pokladnu, v tomto případě polské výroby. Díky té může prodávat jízdenky, vyhledávat cestujícím spoje a mnoho dalšího. Jednou z funkcí je i kalkulačka, nebo spíš, jednou z funkcí by *měla* být kalkulačka. Někdo sice vytvořil uživatelské rozhraní, ale neimplementoval část zajišťující samotné počítání. Je třeba napsat program, který pro zadaný výraz spočítá jeho hodnotu.

Kevin se tedy nabídl, že pro Sářinu maminku kalkulačku doprogramuje. Po přeložení nepříliš detailní dokumentace, která zmiňovala výraz „odwrotna notacja polska“ z polštiny, zjistil, že to nebude tak jednoduché. Kalkulačka sice podporuje jen sčítání, odčítání, násobení a celočíselné dělení, výrazy jsou však zapsány na pohled poněkud nezvykle: na rozdíl od běžného zápisu, kde jsou operátory psány mezi operandy, jsou zde psány za operandy. Z výrazu, který se běžně zapisuje jako  $2 + 3$ , se tedy stane  $2\ 3\ +$ . Operátory se vyhodnocují hladově zleva:  $2\ 3\ 4\ +\ *$  se vyhodnotí nejprve na  $2\ 7\ *$  (nahrazením  $3\ 4\ +$  za 7) a pak na 14.

Pomůžete Kevinovi kalkulačku doprogramovat a tím získat Sářinu přízeň?

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  $N$  – počet operací ve výrazu. Na dalším řádku dostanete výraz, jehož hodnotu je třeba spočítat. Výraz se skládá z čísel a operátorů – sčítání (+), odčítání (−), násobení (\*) a dělení (/). Kalkulačka pracuje pouze s celými čísly a místo klasického dělení používá celočíselné dělení, počítané jako dolní celá část běžného podílu, v Pythonu případně i pomocí operátoru //.

Slibujeme, že výraz bude korektní – nebude se v něm vyskytovat dělení nulou a bude mít správný počet čísel vzhledem k počtu operátorů (čísel je vždy o jedno více než operátorů). Dále slibujeme, že výsledek (včetně mezivýpočtů) se vejde do 64-bitového čísla, do 32-bitového už se ale vejít nemusí. Pokud programujete v Pythonu, nemusí vás to trápit, ale např. v Céčku budete potřebovat `long long int`. Čísla a

operátory oddělujeme mezerami.

*Formát výstupu:*

Na jeden řádek vypíšete hodnotu výrazu.

*Ukázkový vstup:* *Ukázkový výstup:*

4 3  
 $-3\ 5\ 4\ +\ 8\ /\ -2\ /\ *$

*Vysvětlení ukázkového vstupu:* Nejdříve se vyhodnotí  $5 + 4$ , což se rovná 9. Poté se vyhodnotí  $9/8$ , což se rovná 1 (/ znamená celočíselné dělení). Pak se vyhodnotí  $1/ -2$ , což se rovná  $-1$  (dolní celá část  $-0,5$  je  $-1$ ). Nakonec se vyhodnotí  $-3 \times (-1)$ , což se rovná 3, a to je i konečný výsledek.

#### **38-Z3-2 Největší dort** **10 bodů**

V Polárním expresu se blíží čas vánoční večeře a Zuzka jako hlavní kuchařka musí připravit třešňový dort ve tvaru obdélníku pro všechny cestující. Dort je obrovský a třešně jsou rozmístěny úplně nahodile. Dnes má navíc narozeniny pan průvodčí a Zuzka mu chce darovat co největší kousek dortu, který má třešně přesně ve všech čtyřech rozích.

Zuzka nemá čas na hledání, a proto se obrací na vás. Na nakrájení takového dortu není možné použít normální nůž, a proto vám dala speciální kráječ, který dokáže krájet rovnoběžně s osami dortu. Vaším úkolem je najít největší možný obdélník na dortu (se stranami rovnoběžnými s osami), jehož všechny čtyři rohy leží přesně na třešních. Výsledek vraťte jako obsah tohoto obdélníku a souřadnice jeho rohů.

Jinými slovy: Dort má rozměry  $N$ ,  $M$  a obsahuje  $K$  třešní na daných souřadnicích. Hledáte obdélník s maximálním obsahem (šířka  $\times$  výška), kde každý roh obdélníku leží přesně na pozici nějaké třešně. Obdélník musí mít strany rovnoběžné s osami dortu.

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 první řádce vstupu jsou tři čísla  $N$ ,  $M$  a  $K$ , kde  $N$  a  $M$  označují šířku a výšku dortu a  $K$  počet třešní na dortu. Následuje  $K$  řádků, z nichž každý obsahuje souřadnice  $x_i$  a  $y_i$  označující pozici  $i$ -té třešně na dortu.

Poznámka k implementaci: Zuzka má opravdu velký dort, tedy vstupní čísla i výsledek mohou být velmi velká. Speciálně slibujeme jen, že se vejdou do znaménkového 64-bitového čísla. Pokud programujete v Pythonu, nemusíte se tímto faktem znepokojoovat, ale např. v Céčku je nutné použít typ `long long int`.

*Formát výstupu:*

Na první řádek výstupu vypište obsah největšího obdélníku. Na další čtyři řádky vypište páry souřadnic  $x$  a  $y$  označující rohy obdélníku.

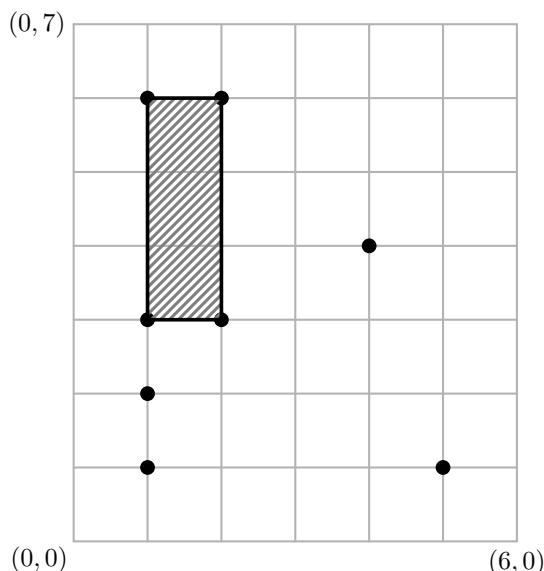
Může se stát, že existuje více největších obdélníků, v tomto případě můžete vypsat kterýkoliv z nich. Slibujeme, že alespoň jeden validní obdélník vždy existuje.

*Ukázkový vstup:*

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

*Ukázkový výstup:*

```
3
1 6
2 3
1 3
2 6
```



### 38-Z3-3 Polární expres 12 bodů

Lokomotiva polárního expresu je velmi neopatrná a nedala si pozor na své vagóny. To vedlo k tomu, že je po nějaké době chudák všechny poztrácela. Protože si ale polární expres nemůže dovolit ztratit jakýkoli vagón, musí je všechny posbírat zpátky, a to v libovolném pořadí. Naštěstí má každý vagón GPS senzor, tím pádem umíme přesně lokalizovat každý vagón v mřížce  $R \times S$ . Mašinka se v mřížce umí pohybovat rychlostí 1 políčko za sekundu, a pokaždé, když přejede políčko s vagónem, tak se souprava na začátku příští

sekundy o tento vagón prodlouží. Souprava si také musí dát pozor, aby nikdy nenabourala do sebe. (Pokud vám tento popis připomíná staré počítačové hry Vlak<sup>1</sup> a Had,<sup>2</sup> tak naprosto oprávněně.)

Mřížka, ve které se souprava nachází, má jednu zvláštnost: její dolní a horní okraj jsou spojené, stejně jako její levý a pravý okraj. Když tedy například lokomotiva přejede přes pravý okraj mapy, objeví se ve stejné výšce na levém okraji.

Pomozte lokomotivě najít cestu, jak sesbírat všechny vagóny. Nemusíte hledat nejkratší cestu; stačí, když vámi nalezená cesta bude mít méně než  $10^6$  kroků.

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 první řádce dostanete počet řádků  $R$ , počet sloupců  $S$ , počáteční pozici lokomotivy  $x, y$  a následně počet vagónů k sesbírání  $N$ .

Na následujících  $N$  řádcích dostanete čísla  $x_i, y_i$  značící pozici  $i$ -tého vagónu. Předpokládejte, že počet vagónů je mnohem menší než počet políček mřížky.

*Formát výstupu:*

Na jednom řádku vypište řetězec popisující pohyb lokomotivy sestávající se ze znaků `v`, `^`, `<` a `>` popisujících pořadí pohyb dolů, nahoru, doleva a doprava.

*Ukázkový vstup:*

```
10 10 5 5 4
1 1
8 1
1 8
8 8
```

*Ukázkový výstup:*

```
>>>vvvvvvv>>>vvvvvvvvvv
```

### 38-Z3-4 Nákup v poslední chvíli 14 bodů

Jídelní vůz Polárního expresu nabízí zcela unikátní zkušenost „vyrob si svou vlastní zmrzlinu“. Po té pochopitelně všichni cestující touží, a proto se u vstupu tvoří dlouhé fronty. Zoufalý manažer otevírá stále další a další stanoviště, kde se dá zmrzlina vyrábět, ale fronta se jen prodlužuje. Pomůžete mu vyřešit jeho problém?

Na vstupu dostanete seznam zákazníků spolu s časy, kdy přišli do fronty. Jednomu výrobnímu stanovišti trvá 5 minut, než si cestující zmrzlinu vyrobí. Pokud se nějaký cestující nedostane ke stanovišti během patnácti minut čekání ve frontě, tak odejde a napíše negativní recenzi. Kolik nejméně stanovišť potřebuje vlak mít, aby byli všichni zákazníci spokojeni? Nezapomeňte zdůvodnit správnost svého řešení.

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

<sup>1</sup> [https://cs.wikipedia.org/wiki/Vlak\\_\(po%C4%8D%C3%ADta%C4%8Dov%C3%A1\\_hra\)](https://cs.wikipedia.org/wiki/Vlak_(po%C4%8D%C3%ADta%C4%8Dov%C3%A1_hra))

<sup>2</sup> [https://cs.wikipedia.org/wiki/Had\\_\(po%C4%8D%C3%ADta%C4%8Dov%C3%A1\\_hra\)](https://cs.wikipedia.org/wiki/Had_(po%C4%8D%C3%ADta%C4%8Dov%C3%A1_hra))



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](mailto:ksp@mff.cuni.cz)

**Organizátoři a kontakty:**  
<https://ksp.mff.cuni.cz/kontakty/>