# Korespondenční Seminář z Programování

## ZAČÁTEČNICKÁ KATEGORIE

**36. ročník** | **KSP-Z** | **Prosinec 2023**

Právě se díváte na leták třetí série 36. 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 11. února ve 32:00** (tedy další ráno v 8:00), praktické úlohy za třetinu bodů až do 18. 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 36. ročníku KSP

### 🖩 36-Z3-1 Kebab — 8 bodů

Kevin dlouho do noci pracoval na domácím úkolu a dostal velký hlad. A když má Kevin hlad, tak to nevěstí nic dobrého. Věděl, že dokud se nenají, tak s domácím úkolem pořádně nepohne. Měl ale jako vždy velké štěstí, v jeho městě se zrovna otevřelo několik nových podniků, které servírují jeho oblíbený pokrm, kebab. Kevin si bohužel také šetří na nový počítač a nemůže si dovolit moc utrácet. Rád by tedy měl nejlevnější kebab, který je možné sehnat.

Kevinovo město má tvar přímky, na které jsou v jednotlivých bodech umístěny restaurace servírující kebab. Kevin se nachází v bodě 0. Po městě se Kevin pohybuje pomocí tramvaje, za kterou se platí podle počtu ujetých kilometrů. Pomozte Kevinovi sehnat nejlevnější večeři.


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

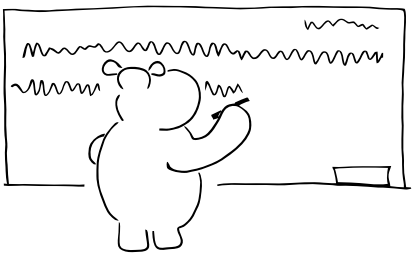
*Formát vstupu:* Na prvním řádku dostanete čísla $N$, $J$, kde $N$ je počet obchodů s kebabem ve městě a $J$ je cena jízdenky za kilometr jízdy tramvají. Na dalších $N$ řádcích pak dostanete čísla $k_i, l_i$, kde $k_i$ je cena kebabu v $i$-tém podniku a $l_i$ je vzdálenost podniku od Kevina.

*Formát výstupu:* Na výstup vypište jediné číslo – nejnižší cenu, za kterou může mít Kevin kebab.

| *Ukázkový vstup:* | *Ukázkový výstup:* |
|---|---|
| 3 4<br>2 7<br>3 3<br>4 5 | 15 |

### 🖩 36-Z3-2 Přednášky — 10 bodů

Nový semestr, starý známý stres pro studenty Matfyzu. Zápis přednášek. Ač se to na první pohled nemusí zdát, zápis přednášek je jako biatlon. Každá sekunda se počítá a jeden špatný zásah kurzorem může znamenat semestr bez oblíbené volitelné přednášky o kávě s profesorem Hrochem. Pokud možno by si proto Prokop rád zapsal všechny volitelné přednášky.



Naneštěstí nemá Hermionin Obraceč času a tak musí co nejvíce využít toho málo času, co má. Aby mu to fakulta neudělala tak jednoduché, jsou všechny Prokopovy povinné přednášky rozházeny mezi dvě budovy Matfyzu: moderní IMPAKT a historickou budovu na Malé Straně. Přejezd trvá celých 40 minut, při kterých se Prokop nezvládne soustředit na nic jiného než hledání cesty spletitým systémem pražské hromadné dopravy. Naštěstí jsou všechny volitelné přednášky streamované online, takže se na ně může Prokop dívat z jakékoliv budovy, ale ne během přejezdu mezi nimi.

Aby si Prokop mohl zapsat co nejvíce volitelných přednášek, rozhodl se napsat si program, který mu určí, kolik si jich může nejvíce zapsat tak, aby stíhal přejezd mezi budovami Matfyzu. Prokop má seznam povinných přednášek, na které musí ať chce nebo nechce. Také má seznam volitelných přednášek, ze kterých si jich chce zapsat co nejvíce. Každá přednáška má svůj stanovený začátek a konec. Každá povinná přednáška má ještě k tomu určeno, na které z budov se koná. Žádné dvě přednášky se nepřekrývají časově, ani volitelná s povinnou, a je vždy možné stihnout všechny povinné přednášky (a je to taky povinné). Před první povinnou přednáškou nemusí Prokop nikam přejíždět, takže se může soustředit na všechny volitelné před ní.

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 vstupu dostanete čísla $P$, velikost seznamu povinných přednášek, a $V$, velikost seznamu volitelných přednášek. Následuje celkem $P$ řádků, kde na každém řádku je povinná přednáška zapsána jako dvě čísla a písmenko. První číslo určuje čas začátku přednášky a druhé číslo určuje čas konce přednášky. Všechny časové údaje jsou zapsány v minutách od začátku semestru. Poté následuje písmenko určující v jaké budově se přednáška koná: `I` pro IMPAKT a `M` pro Malou Stranu. Poté následuje $V$ řádků, kde na každém řádku je volitelná přednáška zapsána pomocích dvou čísel. První určující její začátek a druhé zase její konec. Slibujeme, že jak seznam povinných přednášek, tak seznam volitelných je chronologicky seřazený. Avšak každý samostatně. Slibujeme také, jak již bylo výše řečeno, že žádné dvě přednášky se nebudou překrývat.

*Formát výstupu:* Na výstupu vypište jediné číslo – největší možný počet volitelných přednášek, které může Prokop za tento semestr stihnout.

*Ukázkový vstup:*

```
3 4
5 10 I
10 20 I
100 150 M
1 4
25 30
40 60
60 70
```

*Ukázkový výstup:*

```
3
```

## 36-Z3-3 Peřeje — 12 bodů

Sára se ocitla v zapeklité situaci. Kormidluje loď, která se nezadržitelně žene kupředu posvátnou hroší řekou. Hroši jsou ale hladové bestie a Sára se jim chce vyhnout. Její situaci příliš nepomáhá fakt, že loď je značně neovladatelná. Proud řeky ji posouvá dolů a Sára může lodí zatáčet jen o jedno políčko řeky vpravo nebo vlevo.

Sářina loď má podobu políčka $1 \times 1$. Řeka je obdélníková mřížka $W \times H$, ve které jsou políčka obsazena hrochy, nebo prázdná. V každém kroku se loď posune o jednu řádku níž. Sára může ovlivnit, jestli se zároveň celá loď posune o jedno políčko doleva, doprava nebo zůstane ve stejném sloupečku. Dokážete Sáře poradit, kudy má řeku projet tak, aby v žádném momentu nenarazila do žádného hrocha?

Slibujeme že první řádka je bez hrochů. Cestu můžete začínat kdekoliv na první řádce. Pokud cesta neexistuje, odevzdejte řetězec `NEEXISTUJE`.

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 2 přirozená čísla $W$ a $H$. Na dalších $H$ řádcích následuje $W$-krát `.` nebo `X`, kde `.` odpovídá prázdnému políčku řeky a `X` políčku s hrochem.

*Formát výstupu:* Pokud cesta neexistuje, odevzdejte jediný řádek `NEEXISTUJE`. Pokud existuje, odevzdejte bezkolizní cestu řekou v podobě horizontální pozice lodi, každou hodnotu na samostatném řádku. Sloupce jsou indexované od 0.

*Ukázkový vstup:*

```
10 5
..........
.X.XXX....
..XX...X..
XXX.XX.X.X
XX.XXXX.X.
```

*Ukázkový výstup:*

```
5
6
5
6
7
```

## 36-Z3-4 Výlet — 14 bodů

Kevin si plánuje výlet na vysokou horu. Jako správný horolezec chce vylézt až na její vrchol. Jenže nemá mapu a neví, kde se takový vrchol nachází. Jediné, co mu zbývá, je stahovat data o hoře z internetu. Jeho připojení je ale strašlivě pomalé a může pouze pokládat dotazy na výšku hory v určitém bodě. Na druhou stranu o hoře ví, že nemá žádná sedla – tj. nejdřív stoupá, pak následuje vrchol a poté klesá.

Nyní trochu formálněji. Výšky bodů hory si představme jako posloupnost čísel. Můžete zavolat funkci, které předáte index v posloupnosti a ona vám v konstantním čase vrátí výšku hory v tomto bodě. O posloupnosti máte slíbeno, že je nejdříve rostoucí, pak následuje její maximum a poté je klesající. Najděte toto maximum. Snažte se, aby vaše řešení bylo lepší než lineární.

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

## 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/*