Soutěž dětí a mládeže v programování

Obvodní kolo Prahy 1 a 2, rok 2022

Autoři: Adam Benda, Martin Horský, David Šertler; verze 1.05, 2022-02-21

Každá úloha má koeficient obtížnosti, kterým se násobí hodnocení vašeho řešení. Odevzdejte nám prosím řešení přes odevzdávátko (https://soutez.lnker.xyz/) jako komprimovanou složku zip. Kromě splnění zadané funkčnosti se hodnotí přehlednost vašeho kódu (komentáře, popisující názvy proměnných) a efektivita vašeho řešení.

Při soutěži je zakázána komunikace (přítel na telefonu). Vyhledávač a Internetové zdroje používat můžete ALE: cílem je zjistit vaši schopnost algoritmizace. Zkopírovat si několik řádků kódu ze stackoverflow (uveďte zdroj) nebo použít knihovní funkci řešící dílčí část problému je v pořádku. Zkopírovat odněkud hotové řešení celé úlohy není přípustné. V případě pochybností konzultujte porotce.

Používání knihoven je povoleno, jelikož mnoho programovacích jazyků má relativně omezenou standardní knihovnu (např. grafické zobrazování v okýnkách). Nepoužívejte ale knihovny, které víceméně vyřeší úlohu za vás.

Testovací vstupy k úlohám naleznete na https://soutez.lnker.xyz/res/

Úloha 1: Parallax efekt - Endscreen z Pokémonů (koef. 1,5)

Vytvořte program, který bude vykreslovat (popř. animovat) profily mrakodrapů v různých vzdálenostech (viz https://en.wikipedia.org/wiki/File:Parallax_scroll.gif).

Vstup můžete vzít buď ze souboru nebo ze standardního vstupu buď v textovém formátu, JSON, YAML nebo XML. Stačí implementovat načtení jednoho formátu.

Na vstupu dostanete vzdálenosti od kamery (vrstvy) a délky, po kterých se mají opakovat, a poté seznam mrakodrapů v podobě vzdálenosti, posunu zleva, šířky a výšky. Délky X a Y můžete interpretovat jako pixely. Vzdálenost vrstev slouží pouze pro určení pořadí překrývání a relativních rychlostí vrstev v animaci.

Formát textového vstupu je následující:

```
N_VRSTEV VZDALENOST DELKA VZDALENOST DELKA…
N_MRAKODRAPU
VZDALENOST POZICE_ZLEVA SIRKA VYSKA
VZDALENOST POZICE_ZLEVA SIRKA VYSKA
…
```

Ostatní vstupy používají obdobné seskupení.

Můžete předpokládat validní vstup a hodnoty “rozumné velikosti”. Nemusíte řešit astronomická čísla. Budovy ale mohou mírně přesahovat meze opakování vrstev.

Pro vykreslování použijte nějakou rozumnou velikost obrázku/okna (např. referenční řešení používá 800x600). Konkrétní rychlost animace, škálování rychlosti různých vzdáleností nebo použité barvy jsou na vás.

Hodnocení:

- Varianta A (snazší): vykreslete jeden obrázek a zobrazte (popř. uložte do souboru) (30% hodnocení)
- Varianta B (těžší): animujte pohyb kamery pro dosažení parallaxu (100% hodnocení)

Úloha 2: Televizní program - Telenovela speedrun (koef. 3)

Babička se v televizi ráda kouká na romantiku a detektivky. Právě si koupila televizní program na nový týden a od vás chce abyste jí sledování naplánovali.

Pomozte babičce vytvořit zadání pro televizní ovladač, který jí umožní:
- Algoritmus 1: vidět co nejvíce minut televizního vysílání, které babičku zajímá (50% hodnocení)
- Algoritmus 2: vidět **od začátku do konce** co nejvíc pořadů, které babičku zajímají (50% hodnocení)

Vstup je rozdělen do jednotlivých kanálů označených řetězcem - například “Prima Cool” nebo “ČT1”. Na kanále jsou naplánovány jednotlivé pořady, které mají začátek a konec uvedené ve formátu hodiny:minuty, například “9:15”. Minuta začátku patří do pořadu, koncová minuta ne - můžete přepnout jinam - například pořad 9:02 - 9:07 běží 5 minut, spočítejte si je.
Alespoň jeden žánr pořadu by měl souhlasit s alespoň jedním babiččiným oblíbeným žánrem, aby babičku zajímal. Např. když má babička oblíbené krimi, romance a komedie, bude ji jistě zajímat *krimi western* i *hudební sci-fi romance*.

Máte k dispozici příkazy “Zapni televizi”, “Vypni televizi” a “Přepni na kanál *[jménoKanálu]*”. Váš úkol je napsat posloupnost příkazů, aby babička strávila u televize co nejvíce času (algoritmus 1), nebo aby babička shlédla co nejvíce kompletních pořadů (algoritmus 2). Každý příkaz společně s jeho časem vypište na samostatném řádku. Babička by měla šetřit proudem, proto vždy, když nedávají její oblíbený pořad, televizi vypněte. **Na posledním řádku vypište shrnutí** svého běhu: např. “Nakoukáno 123 minut” v případě algoritmu 1, “Shlédnuto 12 celých pořadů” v algoritmu 2.
Přepnutí/vypnutí/zapnutí tv se stane okamžitě a to na počátku zadané minuty. Na začátku je televize je vypnutá.

Vstup úloh je posytnut v CSV (hodnoty oddělené čárkou, neuvozované) a nebo v JSON - vyberte si formát, se kterým se vám bude pracovat pohodlněji. Data jsou v obou formátech ta samá. CSV vstup jsou dva soubory, babicka.csv obsahuje pouze jednu řádku (preferované

žánry) a program.csv obsahuje seznam pořadů se sloupci: jméno kanálu, jméno pořadu, začátek, konec, žánry (jednotlivá slova oddělená mezerou).

Váš program budeme testovat i na větších datech (cca 50 kanálů, 20 tisíc pořadů) a běh omezíme 5s. Neefektivní řešení tedy o část bodů přijde. Velké sady jsou i v testovacích datech.

Úloha 3: Sokoban - 4K Definitive Edition (koef. 2)

Naimplementujte hru Sokoban https://en.wikipedia.org/wiki/Sokoban. Podobnou hru jste jistě už někde viděli. V této hře ovládáte panáčka na mapě se zdmi a boxy. Panáček se může pohybovat po ortogonálních osách - nahoru, dolu, doleva, doprava. Při pohybu může posouvat boxy, tak že na ně tlačí. Box může být posunut pouze pokud je za boxem volné políčko. Nemůže posouvat boxy do zdí nebo posouvat vícero boxy v řadě současně. Boxy jsou na mapě na začátku rozmístěné. Zároveň jsou na mapě cílová políčka pro boxy. Cílem hry je umístit každý box na některé cílové políčko. Počet boxů je roven počtu cílových polí.

Vaše hra by měla načíst mapu ze souboru nebo standardního vstupu v textovém formátu, JSON, YAML nebo XML. Stačí implementovat jeden formát. Textový formát obsahuje šířku a výšku mapy na prvním řádku. Další řádky popisují mapu. Řádek odpovídá řádku mapy a obsahuje jeden znak na jedno políčko:

`W` - Zeď
`E` - Prázdné pole
`T` - Prázdné cílové pole
`P` - Panáček
`B` - Box
`p` - Panáček na cílovém poli
`b` - Box na cílovém poli

Ostatní vstupy používají obdobné seskupení za pomoci seznamů. Pokud načítáte mapu ze souboru, můžete cestu k souboru brát buď z argumentů programu, zobrazením dialogu nebo nějakým jiným “rozumným” způsobem.

Můžete předpokládat validní vstup “rozumné velikosti”. Platí, že panáček je pouze jeden a počet boxů je stejný jako počet cílových polí. Splnitelnost mapy neřešte (na kombinatorické programování je druhá úloha 😉).

Pro zobrazení hry můžete použít čistě textovou variantu, kdy zobrazíte stav hry za pomoci znaků v příkazové řádce, kterou budete opakovaně přepisovat. Pro lepší bodový zisk musíte ale hru zobrazit graficky. Vykreslování můžete provést ručním kreslením piktogramů nebo obrázky (např. zde volně dostupné https://github.com/borgar/sokoban-skins).

Vaše hra by měla mít možnost pohybu panáčka. Ideálně za pomoci WASD nebo šipek. Můžete také použít tlačítka zobrazená v okně.

Při dosažení cílového stavu hry upozorněte uživatele na konec hry a hru ukončete, případně nabídněte spuštění znovu nebo načtení jiné mapy.

Hodnocení:

- Funkčnost herního modelu - 40% hodnocení
- Grafický výstup - 40% hodnocení
    - Textový výstup - 10% hodnocení
- Ovládání klávesy - 20% hodnocení
    - Ovládání tlačítky - 10% hodnocení

Úloha 4: Kuchařka - Hrnečku vař (koef. 1,5)

Vaše máma už neví, co má vařit, a pomalu ji dochází veškeré nápady a inspirace. Je zkrátka bezradná. Proto vás poprosila o program, který vypíše všechna jídla, která může uvařit z toho, co má doma. Suroviny bude zadávat po jedné a program je potvrdí. Tedy jeden řádek = jedna surovina. Program po zadání suroviny potvrdí její přijetí a vypíše kolikátá je. Poslední surovinu zadá tak, že na další řádek napíše „KONEC“. Po zadání poslední suroviny jí program vypíše seznam možných jídel a všeho, co k tomu potřebuje. V programu se nespecifikuje množství surovin, duplicity ignoruje. Zadání vstupu uživatele můžete řešit také graficky.

Vy máte naprogramovat jádro programu, které dostává zprávy (jako řádky textu):

//Modrá = vstup uživatele, Zelená = výstup programu

```
VEJCE
Byla přidána 1. ingredience
SUL
Byla přidána další (2.) ingredience
MLEKO
Byla přidána další (3.) ingredience
MLEKO
Duplicita => Ignoruji
HLADKA_MOUKA
Byla přidána další (4.) ingredience
KONEC
Poslední položka byla přidána.
Hledám recepty se 4 ingrediencemi: VEJCE, SUL, MLEKO a HLADKA_MOUKA
Nalezen 1 recept:
PALACINKY
SUL
VEJCE
MLEKO
```

`HLADKA_MOUKA`

Po přidání poslední ingredience se všechny ingredience vypíší pro kontrolu. Počet ingrediencí v receptu musí být vždy stejný nebo menší než v seznamu. Nemusí být využity všechny ingredience ze seznamu, ale navržený recept se musí skládat pouze z ingrediencí v seznamu.

Cokoliv co následuje po řádku „KONEC“ je ignorováno. Po vypsání všech platných receptů běh programu ukončete. Pokud program nenajde žádný recept, tak vypište chybovou hlášku program ukončete.

Recepty načítejte ze souboru “recepty.txt”, popř. “.json”, “.yaml”, “.xml”. Stačí implementovat jeden formát vstupu. Žádný recept nebude mít více než tisíc položek, receptů nebude více jak tisíc. Textový formát má recepty oddělené prázdnými řádky, kde v každém bloku je na prvním řádku název jídla a na dalších jednotlivé ingredience.

Př:

```
PALACINKY
SUL
VEJCE
MLEKO
HLADKA_MOUKA

ZAZVOROVA_MEDICINA
CITRON
ZAZVOR
MED
```

Hodnocení:

- Implementace zpracování vstupu - 50% Hodnocení
- Implementace hledání receptů - 50% Hodnocení