Soutěž v programování 2024
Obvodní kolo Prahy 2
Zhodnocení úloh

Vážení účastníci,

letos na obvodním kole programovalo 40 účastníků a sešlo se nám 105 řešení úloh. 22 řešení úloh bylo ohodnoceno maximálním počtem bodů. Nejlepší řešitel získal 80,7 bodů z 95 možných.

S přísným časovým limitem čtyř hodin tedy soutěžící odvedli úžasné množství práce - formulace problémů, rozvaha které nástroje zvolit, algoritmizace, testování a ladění.

Doufáme, že jste si řešení úloh užili - a že se vám zkušenost bude někdy v budoucnu hodit. Abyste se s námi mohli ještě jednou nad úlohami zamyslet, přinášíme krátké zasazení úloh do kontextu, a rozbor nejčastějších chyb.

Adam Benda, Martin Horský, David Šertler - porotci

Zadání úloh, testovací vstupy i referenční řešení naleznete na webu programovacisoutez.cz

Úloha 1: Pyramida

O úloze

Tuto úlohu na **základní algoritmizaci** vyřešila úspěšně většina účastníků.
Spousta z vás se dobrala k správnému vzorci pro počet zobrazených znaků a mezer pro daný řádek - úvahou či experimentálně. Rovněž ale šlo řešit postupným zvyšováním těchto počtů mezi jednotlivými řádky - podobné matematické indukci.

Nejčastější chyby

- Chyba v počtu řádek o ± 1
- Chybějící možnost výběru znaku pro vykreslení

Úloha 2: Inventář plyšáků

O úloze

Pro řešení úlohy musel soutěžící sám vymyslet **způsob ukládání** inventáře. To byla stěžejní část úlohy. Možností je celá řada:

- ukládání do souboru (nejčastější)

- uložení v relační databázi (například SQLite či MySQL) - dva soutěžící
- JS local storage - ti, kteří použili jako běhové prostředí webový prohlížeč

Všechna tato řešení jsou rozumná, ale každé má své omezení.
Ukládání do souboru je asi nejjednodušší přístup. Program ale musí mít dodatečnou logiku pro zpracování souboru v konkrétním formátu. Zároveň nesmí nastat pokus o otevření souboru pro zápis, když už je jednou otevřený.
Ukládání do SQL databáze je složitější z pohledu zajištění přístupu do databáze, ale následně je možné provádět některou logiku programu přímo v databázi. Například hledání v inventáři.
Součástí zadání byl také požadavek na efektivní hledání v inventáři. Toho lze snadno dosáhnout ukládáním plyšáků do stromu nebo hash tabulky, které jsou zpravidla implementované ve standardních knihovnách.

Nejčastější chyby

- Lineární hledání v inventáři = při hledání musím projít všechny plyšáky
- Hledání v SQL databází načtením všech dat do programu a lineárním průchodem
- Vícenásobné otevírání souboru s databází
- Pevné absolutní cesty k souboru databáze

Námět k rozšíření

Všechna řešení v Pythonu (a většina řešení v jiných jazycích) byla textová a nezaobírala se grafickým rozhraním. Je jasné že v našem formátu soutěže je textová interakce nejspíš snazší a rychleji implementovatelná, a hodnocení ji nepenalizuje.
Reálně by klikací GUI řešení bylo pro uživatele o mnoho atraktivnější. Zkuste tedy prozkoumat možnosti svého jazyka! V Pythonu je například součástí standardní knihovny Tkinter.

Úloha 3: Já tě vidím

O úloze

Grafická úloha, ve které se bez základů analytické geometrie neobejdete.

Úkolem bylo dosáhnout několika specifických cílů. Prvním byla možnost konfigurace, která mohla být implementována prostřednictvím grafického uživatelského rozhraní nebo načítáním konfigurace ze souboru. Při volbě načítání konfigurace ze souboru bylo klíčové určit vhodný datový formát nebo definovat vlastní strukturu dat. Dalším úkolem bylo efektivně využít grafickou knihovnu, přičemž kreslení očí a zorniček vektorově se ukázalo jako nejvhodnější přístup. Další důležitou součástí byla práce se souřadnicemi, kde hlavní výzvou bylo přesné škálování souřadnic myši. Bez ohledu na to, zda byly souřadnice určeny vzhledem k celé obrazovce nebo pouze k vykreslovacímu oknu, bylo nezbytné je korektně přiřadit k pozici očí a přizpůsobit velikosti a umístění oka na obrazovce. Nejefektivnější řešení nevyžadovalo použití goniometrických funkcí, ale založilo se na vypočítání vzdálenosti kurzoru myši od středu oka, dostupného prostoru uvnitř oka a maximální možné

vzdálenosti zorničky od středu oka. Následně bylo potřeba určit měřítko pro přepočet souřadnic myši, aby zorničky zůstaly uvnitř oka.

Nejčastější chyby

- Pozice myši byla počítána jako pozice na obrazovce, ale interpretována jako pozice oproti oknu programu
- Program byl málo nastavitelný
- Trhaná animace

Úloha 4: Pošta

O úloze

Pokročilá úloha související s **teorií grafů**, což je nesmírně zajímavá a prakticky používaná oblast informatiky.

Úloha je zjednodušený travelling salesman problem. Cílem je najít nejkratší okružní cestu v grafu, nad malým počtem cílů a s možností opakování hran. Pro takový problém známe tři rozumná řešení:

1. 2D Dijkstra
2. NxN Disjkstra + vyzkoušení všech cest
3. Floyd-Warshall + vyzkoušení všech cest

Možnosti 2. a 3. mají celkem jednoduchý princip. Typicky pro garanci nalezení nejkratší cesty můžeme najít všechny přijatelné cesty a najít mezi nimi tu nejkratší. Každá okružní cesta se zaručeně bude skládat z nejkratších cest mezi příjemci. Tak tedy určíme nejkratší cesty mezi každou dvojicí cílů a vyzkoušíme všechna uspořádání cílů. Na nalezení nejkratších cest mezi cíli můžeme použít například opakování Dijkstrova algoritmu nebo algoritmus Floyd-Warshall.
Možnost 1. je o něco komplikovanější. Používá jeden průchod Dijkstrova algoritmu, ale jako vnitřní stav algoritmu nepoužívá pouze pozici v grafu, ale kombinaci pozice v grafu a již navštívených cílů.
Všechny tyto možnosti jsou typicky dostatečně efektivní, avšak každý z těchto algoritmů je asymptoticky mírně rychlejší na konkrétních vstupech. Více můžete dohledat na https://www.baeldung.com/cs/shortest-path-to-nodes-graph.

Nejčastější chyby

Všechna řešení, která nacházela okružní cestu, používala hladový algoritmus s Dijkstrou. Z každého cíle našla další nejbližší cíl Dijkstrovým algoritmem. Tímto způsobem na nahodilých grafech občas nachází nejkratší cestu, ale není to garantované. Zároveň je jednoduchý způsob jak vymyslet graf, na kterém nalezne takový algoritmus cestu se skoro dvojnásobnou délkou.