Záverečný test
praktická časť

Ústav informatiky
Prírodovedecká fakulta
UPJŠ v Košiciach

Doplňujúce zdrojové kódy sú na stránke predmetu PAZ1b. Funkčnosť každého riešenia musí byť preukázaná spustením na testovacom vstupe – nespustiteľné riešenia neumožňujú zisk príslušných bodov.

Pozdravy od Mateja – Panem et circenses

Heroes on the boat (6 + 4 + 3 + 6 bodov, backtracking)

V mnohých počítačových hrách sa môžete stretnúť s hrdinami a jednotkami, ktoré treba prepraviť loďou. Nie vždy je to ako na ilustračnom obrázku, že každá jednotka zaberá rovnaké miesto. V moderných hrách kapacita lode nevyjadruje počet jednotiek ale nosnosť, veľkosť alebo objem jednotiek. Aj malé jednotky môžu oplývať veľkou silou (trpaslíci), preto je prirodzené sa pýtať, ktoré jednotky chceme naloditť na loď s kapacitou tak, aby mala posádka, čo najväčšiu silu. Pozn.: Predpokladáme, že hmotnosť aj sila jednotky môže byť reálne číslo a každú jednotku máme práve raz.

Úlohy:
(6b) Implementujte program, ktorý načíta (napr. z textového súboru) nosnosť lode a zoznam jednotiek, ktoré môžeme naloditť. Zoznam jednotiek obsahuje o každej jednotke jej hmotnosť a jej silu. Výstupom programu je zoznam jednotiek s najväčšou možnou silou, ktorú je možné na naloditť na loď.
(+4b) Upravte program tak, aby načítal aj počet lodí (lode majú rovnakú nosnosť) a vrátil taký zoznam jednotiek jednotlivých lodí , pri ktorom je sila nalodených jednotiek najväčšia.
(+3b) Upravte program tak, aby načítal aj počet lodí a ich nosnosti (nosnosti lodí môžu byť rôzne). Program by mal vrátiť, ktoré jednotky treba naloditť na ktorú loď, aby bola sila nalodených jednotiek najväčšia.
(+6b) Podľa efektívnosti. Algoritmus je tým efektívnejší, či viac eliminujete výpočty, ktoré nevedú k prípustnému riešeniu.

Lanchesters square law + Rozdeľuj a panuj alebo ako vyhrať proti presile

O stratégii rozdeľuj a panuj ste už počuli. Viete si predstaviť, že naša armáda o sile 1000 porazí nepriateľskú armádu o sile 500 a v ďalšom boji zvyšok našej armády porazí druhú nepriateľskú armádu o sile 500. Aká bude sila našej armády po prvej bitke a po druhej bitke? Odpoveď na túto otázku a ďalšie podobné otázky prvý krát publikovali F. Lanchester a M. Osipov nezávisle od seba už v roku 1916 počas prvej svetovej vojny. Odpoveď v sebe zahŕňa postupnosť diferenciálnych rovníc a viacero predpokladov, ale zjednodušený výsledok sa dá vyjadriť veľmi jednoduchou rovnicou. Majme armády o silách *a* a *b*, kde *a* > *b*, potom po vzájomnom boji armád bude sila armády *b* nulová a zostatkovú silu armády a_z vypočítame nasledovne: $a_z = \sqrt{a^2 - b^2}$. Môžete si všimnúť, že výsledný zjednodušený vzorec sa podobá na úpravu Pytagorovej vety. Ak *a* <= *b* potom je a_z rovné nule.

Lanchesters square law I a cesta proti okupantom (18 bodov, grafové algoritmy)

Mnoho hier využíva myšlienku cesty, ktorou sa treba prebojovať. Začnete doma a máte sa prebojovať do cieľa. Počas cesty narážate na odpor a čím ste ďalej tým je vaša sila menšia. V realite táto situácia nastáva napríklad pri útoku do tilu nepriateľa.
Neuveríte, čo sa stalo v našom kráľovstve! VeľkoPAZske kráľovstvo bolo napadnuté nepriateľskou armádou, ktorá obsadila všetky mestá okrem hlavného mesta H a veliteľ nepriateľskej armády Bek TreKING von Zložitosť si založil svoje veliteľstvo v meste Z. Mapa kniežatstva je graf (neorientovaný, neohodnotený), o každom meste (vrchole) vieme aká silná nepriateľská armáda v ňom sídli. Múdra kráľovná Gabriela von UPJŠ ešte pred obsadením stiahla všetkých vojakov do hlavného mesta a vie, že ak dobyje nepriateľské veliteľstvo, zvyšok nepriateľskej armády sa rozpŕchne a ona oslobodí kráľovstvo. Kráľovná musí postupovať po grafe vždy tak, že ak dobyje nepriateľské mesto, sila jej armády klesne podľa Lanchestrovho vzorca. Môže kráľovná dobyť nepriateľské veliteľstvo? Akou cestou sa má vybrať kráľovná, aby dobyla nepriateľské veliteľstvo?

Úloha: (12b) Vytvorte program, ktorý načíta graf, veľkosť posádok v jednotlivých mestách, kráľovnine hlavné mesto H a vráti, aká silná armáda ostane kráľovnej po dobytí nepriateľského veliteľstva Z. Pozn. Ak sa nedá dobyť mesto Z, potom nech program vráti hodnotu 0.
Hint 1.: Keďže armáda (ohodnotenie) je vo vrcholoch, viete ho preniesť na hrany. Napríklad neorientovanú hranu medzi vrcholmi A a B s posádkami 100 a 200 nahradíte dvojicou orientovaných hrán (A, B) s váhou 200 a (B, A) s váhou 100. To znamená, že ak idete dobyť mesto B musíte sa stretnúť s armádou o sile 200.
Hint 2.: Úlohu je možné riešiť algoritmom Bellman-Ford. Namiesto vzdialenosti pre každý vrchol budeme počítať, s akou najväčšou armádou sa vie kráľovná dostať do vrcholu. Pričom relaxácia bude zmenená vzhľadom ku počítanej hodnote a Lanchesterovmu vzorcu.
Hint 3.: Vzhľadom ku Lanchestrovmu vzorcu je nutné používať desatinné čísla.
(+6b) Ak kráľovná môže dobyť mesto Z, tak program vypíše cestu, akou má kráľovná postupovať. Inak vypíše „prehrali sme“.

Lanchesters square law II a rozdeľuj a panuj (1-7 bodov)

Kráľovná má vojsko o sile *D* a nepriatelia majú vojsko o sile *Z*, kde *D* < *Z*. Kráľovná vie zabezpečiť, aby sa nestretla s celým nepriateľským vojskom naraz, ale stretne sa n-krát s vojskami o sile *Z*/*n*. Napíšte metódu, ktorá vráti najmenšiu hodnotu *n* tak, aby kráľovná postupne vyhrala všetkých *n* súbojov (výsledok súboja sa počíta Lanchestrovym vzorcom).
(1b) Za riešenie v čase $O(n)$.
(7b) Za riešenie v čase $O(\log n)$.

Po bitke - Návrat k midtermu (6 bodov)

Kráľovnej sa podarilo oslobodiť kráľovstvo! Keďže udržiavanie veľkej armády na jednom mieste vyžaduje veľké úsilie (zásobovanie a logistika) armádu opäť rozoslala do svojich domácich miest, kde vojaci bývajú pri rodine. Ak sa nebojuje, môžu sa venovať iným zamestnaniam.
Kráľovná má v jednotlivých mestách posádky *d[0],d[1],…,d[n-1]*. Žiaľ všetky nepriateľské vojská sa nerozpŕchli a po krajine majú posádky o veľkostiach *z[0],z[1],…,z[m-1]*. Kráľovná môže poslať každú zo svojich posádok ešte na jednu výpravu proti ľubovoľnej jednej posádke nepriateľov.

Úloha: (6b) Vytvorte program, ktorý pre polia *d* a *z* na vstupe vráti, najviac koľko nepriateľských posádok môže byť porazených. Pozn.: Očakáva sa riešenie v čase $O(k \log k)$, kde $k = \max(n, m)$.

No more minerals (17 bodov, dynamické programovanie)

Budovateľské aj stavebné strategické hry využívajú suroviny, za ktoré sa nakupujú vojaci, budovy alebo sa stavia priemysel. Zoberme si hru, ktorá ma iba 2 druhy surovín. Predpokladáme, že ďalšie suroviny už nemožno získať a hráčovi ostalo *x* surovín jedného druhu a *y* surovín druhého druhu. Hráč chce využiť <u>všetky</u> ostávajúce suroviny, pričom ich môže využiť na stavbu jednotiek z ponuky *n* rôznych jednotiek, kde *i*-ta jednotka stojí X[i] prvej a Y[i] druhej suroviny.

Úlohy: (10b) Napíšte metódu, ktorá pre zoznam jednotiek, ktoré môže hráč stavať a ostávajúce suroviny x, y na vstupe vráti, či môže hráč využiť všetky suroviny alebo nie. Akceptované sú iba riešenia v polynomiálnom čase.
(+7b) Ak hráč môže využiť všetky suroviny doplňte program, aby vypísal jedno z riešení, aké jednotky má postaviť (koľko ktorých jednotiek).

Hint: Označme si *R[j,k]* hodnotu true/false označujúcu, či vieme postaviť budovy za *j* prvej suroviny a *k* druhej suroviny alebo nie.

Slinging (8 - 16 bodov, návrat k midtermu + greedy algoritmy, ???)

V tímových strategických hrách sa môžete stretnúť so stratégiou nazývanou „sling“ alebo „slinging“, kedy hráč ekonomicky podporuje spoluhráča (posiela mu väčšinu svojich surovín). Spoluhráč vďaka dvojnásobnej ekonomike môže rýchlejšie postupovať technológiami a následne stavať väčší počet technologicky pokročilejších jednotiek. Profesionálny hráči vedia presne určiť, v akom čase začne „slinging“ a aké jednotky sa v akom čase začnú vyrábať z dodatočných surovín. Jedna produkčná budova môže naraz vyrábať jednu jednotku. Zaujíma nás koľko najmenej produkčných budov potrebujeme, aby sme všetky jednotky zo vstupu mohli vyrobiť podľa harmonogramu.

Úloha: (8b) Na vstupe je zoznam dvojíc: čas začiatku výroby jednotky a dĺžka výroby jednotky (časy sú uvedené v sekundách). Implementujte program, ktorý načíta vstup a vráti najmenej koľko produkčných budov je potrebných na výrobu jednotiek podľa harmonogramu. Očakáva sa riešenie v polynomiálnom čase.
(+8b) Podľa časovej zložitosti v závislosti od veľkosti vstupu a časov. Najlepšie riešenie je v čase $O(n\log n)$ a nezávislé od času.

Vzorový vstup:

```
0 3
5 5
5 2
8 2
0 4
```

Tretí riadok vyjadruje, že jednotka sa začne vyrábať v čase 5 a dĺžka jej výroby je 2, teda bude dokončená v čase 7. Program na vzorovom vstupe má vrátiť hodnotu 2.
Hint: Pre niektoré riešenia je jednoduchšie uvažovať, že o každej jednotke vieme čas začiatku jej výroby a čas konca výroby.