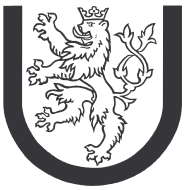

ZÁPADOČESKÁ
UNIVERZITA
V PLZNI

Samostatná práce
z předmětu
Počítače a programování 2

Jméno a příjmení: *Martin Sloup*
Osobní číslo: *A04372*
Studijní skupina: *B13*
Obor: *Informatika*
E-mail: *msloup@students.zcu.cz*

Označení zadání: *B - From Dusk Till Dawn*

Datum odevzdání: 11.5.2005

Zadání

Od večera do rána (From Dusk Till Dawn)

Vladimír má bílou pleť, velmi dlouhé zuby a je mu 600 let. Žádný div, Vladimír je totiž upír. Být upírem nečinilo nikdy Vladimírovi problém. Je znamenitým lékařem, který si zásadně bere noční služby, díky čemuž má mezi svými kolegy mnoho přátel. Ovládá velice působivý trik, který předvádí na večírcích: dokáže stanovit krevní skupinu pouhým ochutnáním krve. Vladimír miluje cestování, ale díky tomu, že je upír, se musí potýkat se třemi překážkami.
Může cestovat pouze vlakem, protože si sebou musí vzít svou rakev. Mimochodem, protože investoval značné množství peněz do dlouhodobých cenných papírů, může si dovolit jet první třídou.

Může cestovat jen od večera do rána, konkrétně od 18:00 do 6:00. Během dne musí zůstat na nádraží v rakvi.

Musí si sebou vzít něco k snědku. Na každý den potřebuje jeden litr krve, který pije vždy v poledne (12:00) ve své rakvi.
Pomozte Vladimírovi nalézt cestu mezi dvěmi zadanými městy tak, aby mohl cestovat s minimem krve (tj. časově nejkratší) a vyhnul se tak všetečným otázkám: „A na co vlastně potřebujete tolik krve?“

Vstup

První řádka na vstupu obsahuje přirozené číslo určující počet testovaných případů, které dále následují. Každý testovaný případ začíná jedním přirozeným číslem udávající, kolik tras bude následovat. Každá trasa obsahuje jména dvou měst, čas odjezdu z prvního města a dobu jízdy, než vlak dorazí do druhého města. Všechny časy jsou uváděny v hodinách (pozn.: Vladimír nemůže cestovat vlakem, který odjíždí dříve než v 18:00 nebo přijíždí později než v 6:00). Maximální počet měst je 100 a počet tras mezi městy je menší než 1000. Žádná trasa netrvá méně než hodinu a více než 24 hodin. Vladimír samozřejmě může využít jen trasy dlouhé nejvýše 12 hodin (od večera do rána). Názvy měst nejsou delší než 32 znaků. Na vstupu dále následuje řádka s názvy dvou měst. První z nich je výchozí město pro Vladimírovu cestu, druhé pak město cílové.

Výstup

Pro každý testovaný případ se vypíše jeho číslo pořadí na vstupu následované textem `"Vladimir needs # litre(s) of blood."` resp. `"There is no route Vladimir can take."` (Přeloženo: Vladimír potřebuje # litru krve - resp. - Cesta neexistuje.)

Příklad vstupu

```
2
3
Ulm Muenchen 17 2
Ulm Muenchen 19 12
Ulm Muenchen 5 2
Ulm Muenchen
10
Lugoj Sibiu 12 6
Lugoj Sibiu 18 6
Lugoj Sibiu 24 5
Lugoj Medias 22 8
```

```
Lugoj Medias 18 8
Lugoj Reghin 17 4
Sibiu Reghin 19 9
Sibiu Medias 20 3
Reghin Medias 20 4
Reghin Bacau 24 6
Lugoj Bacau
```

Příklad výstupu

```
Test Case 1.
There is no route Vladimir can take.
Test Case 2.
Vladimir needs 2 litre(s) of blood.
```

Vypracování

Popis použitého algoritmu

Pro následující problém jsem použil graf. Algoritmus jsem dimenzoval pro 100 měst. Použil jsem pole vrcholů (měst) a následně u každého vrcholu (města) spojový seznam pro uchování vlaků k různým vrcholům (městům) s odjezdy a jejich dobou jízdy. Jako ukazatel cílové stanice jsem použil index určující pozici v poli vrcholů (měst). U každého vrcholu byla použita proměnná *cas*, která určovala nejmenší čas dosáhnutí tohoto vrcholu při hledání cesty mezi dvěma městy. Na začátku jí byla přiřazena maximální hodnota *Integru* 2^{31}-1.

Při tvoření vrcholů jsem bral v potaz již podmínky, kdy upír Vladimír může vlak použít. Pokud tedy nevyhovoval vlak podmínkám, tak jsem ho tam ani nepřidával. Taktéž pro výpočet bylo lepší přepočítat odjezdy vlaků z *18:00 – 06:00* na *00:00 – 12:00*, protože se s tím lépe počítalo.

Po naplnění grafu vrcholy provedeme výpočet časů k dosáhnutí vrcholu. Vrcholu, z kterého začneme počítat, přiřadíme *cas = 0* a pro všechny sousedící vrcholy provedeme výpočet času k jejich dosažení podle následujícího vzorce:

$$cas = aktualni\_vrchol.cas + \left|(24 - (aktualni\_vrchol.cas \ \% \ 24 - vlak.odjezd)) \ \% \ 24\right| + vlak.trvani$$

Pokud tento vypočítaný čas je menší než čas vrcholu, kde má vlak cílovou stanici, tak tento čas do tohoto vrcholu zapíše. Toto provedeme pro všechny sousedící vrcholy (města).

Nakonec pak stačí přečíst hodnotu konečné stanice, kam musí dorazit upír Vladimír. Pokud tato hodnota je rovna maximální hodnotě *Integru* (2^{31}-1), pak Vladimír do této stanice nikdy nedojede. Pokud ale naopak se nerovná této hodnotě, pak máme k dispozici celkový čas (v hodinách) potřebný k dosažení tohoto vrcholu (města). Získaný čas vydělíme hodnotou 24 a provedeme korekce pro zajištěni správného počtu litrů krve.

Popis programu a použitých tříd

Všechny potřebné třídy bylo nutné kvůli robotu uložit do jednoho souboru. Pro správnou funkci Vladimíra bylo potřeba x tříd. Třídy *Polozka* a *Fronta* pro práci s frontou. *Vrchol* pro

uchovávání informací o vrcholu (městu). *Soused* pro evidenci vlaků, jejich odjezdů a době trvání. *Graf* pro veškerý výpočet, od zadávání až po zjištění potřebného množství krve. A nakonec třída *Main* potřebná pro obsluhu celého výpočtu.

Důležité metody:
Třída Graf:

- ***`void pridej(String mesto_z, String mesto_kam, int odjezd, int trvani)`***
  Přidá vlak z města *mesto_z* do města *mesto_kam* s odjezdem odjezd a délkou trvání trasy *trvani*.
- ***`int vratIDMesta(String mesto, boolean vytvořit)`***
  Pomocná procedura pro získání indexu vrcholu v poli z názvu města *mesto*. Parametr *vytvorit* určuje, zda-li se má vytvořit požadované město, pokud nebude nalezeno.
- ***`int pocetLitru(String mesto_z, String mesto_kam)`***
  Provede výpočet potřebného množství krve pro cestu z města *mesto_z* do města *mesto_kam*.

Třída Main:

- ***`static String ctiSlovo()`***
  Načítá ze vstupu znaky až do nalezení mezery nebo konce řádku, přečtené znaky ukládá do objektu typu StringBuffer a nakonec převede na řetězec, který je pak jejím výstupem.

Ověření správnosti

```
Dear Martin Sloup:

Your JAVA program has solved Ok the problem 10187 (From Dusk Till Dawn)
in 0.262 seconds using as much as 2676 kbytes of virtual memory.
Congratulations!

--

PS: Check the board at http://acm.uva.es/board/

The Online Judge (Linux acm.uva.es 2.4.18-27.7.x i686)
Judge software version 2.8 [http://acm.uva.es/problemset/]
Sun May 8 20:44:52 UTC 2005
```

Závěr

Zadaný problém nebyl až tak složitý, jak jsem si myslel. Stačilo se jen trochu zamyslet a použít znalosti, co jsme získali z přednášek *Počítače a programování 2*. Jediný velký problém dělaly vstupy z klávesnice, kdy u mě na počítači to fungovalo, ale robot vracel *Wrong Answer*. A také různé další omezení robota. Například braní v úvahu, že robot má JDK 1.1.

Použité zdroje

- Přednášky z předmětu Počítače a programování 2 – Prof. Ing. Jiří Šafařík, CSc.
- eKniha Počítače a programování 2 - Prof. Ing. Jiří Šafařík, CSc.
- Dokumentace k programovacímu jazyku Sun Java