Záverečný test
teoretická časť

Ústav informatiky
Prírodovedecká fakulta
UPJŠ v Košiciach

Píšte prosím čitateľne!

Hodnotenie vyplní opravujúci:

Meno a priezvisko:	Skupina PAZ:	

1/3	2/1	3/5	4/2	5/1.5	6/1	7/1.5	8/2.5	9/4	∑/21.5

1. (3b) Metóda `finalterm` dostane ako parameter `g` maticu susedností jednoduchého ohodnoteného neorientovaného grafu a v poli `s`, ktorého dĺžka je rovná počtu vrcholov grafu, informáciu, ktoré vrcholy grafu patria do nejakej množiny S (`s[i] == true` $\Leftrightarrow i \in S$).

```
public static double finalterm(double[][] g, boolean[] s) {
      double result = Double.POSITIVE_INFINITY;
      for (int i = 0; i < g.length; i++)
            for (int j = i + 1; j < g.length; j++)
                  if (s[i] && (!s[j]))
                        result = Math.min(result, g[i][j]);

      return result;
}
```

Nájdite taký ohodnotený neorientovaný <u>graf</u> s 8 vrcholmi a takú 4-prvkovú <u>množinu</u> vrcholov S, že metóda `finalterm` vráti číslo 29. Graf zakreslite a uveďte prvky množiny S.

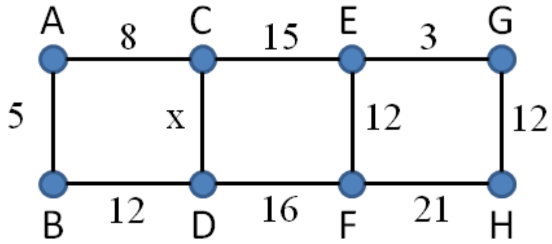
2. (1b) Pri hľadaní najkratších ciest v grafe je základnou operáciou relaxácia hrany. Nech *d[i]* je aktuálny odhad dĺžky najkratšej cesty zo štartovacieho vrcholu do vrcholu *i*. Podobne nech *d[j]* je takýto odhad pre vrchol *j*. Ďalej nech *g[i][j]* je ohodnotenie orientovanej hrany z vrcholu *i* do vrcholu *j*. Napíšte „zdrojový kód“ **operácie relaxácie** orientovanej hrany z vrcholu *i* do vrcholu *j*:

3. (3b za výpočet **d**, 2b za výpočet **p**) Predpokladajme, že matica **d** (`double[][] d`) obsahuje maticu susednosti orientovaného ohodnoteného grafu s hodnotou ∞ indikujúcou neprítomnosť hrany. Ďalej nech štvorcová matica **p** (`int[][] p`) obsahuje na indexe *p[i][j]* hodnotu *i*, resp. *-1* ak v grafe orientovaná hrana z *i* do *j* neexistuje. Napíšte časť zdrojového kódu Floyd-Warshallovho algoritmu, ktorého aplikovaním bude v matici *d* namiesto matice susednosti uložená v políčku *d[i][j]* dĺžka najkratšej cesty z vrcholu *i* do vrcholu *j* (ak existuje, inak ∞) a pre každú dvojicu vrcholov bude v matici *p* v políčku *p[i][j]* uložený index vrcholu, ktorý je predchodcom vrcholu *j* na najkratšej ceste z vrcholu *i* do vrcholu *j* (ak existuje):

4. (2b) V obchode s anténovou technikou predávajú k typov útlmových článkov s odpormi $r_1, r_2, \ldots, r_k$. Všetky majú rovnakú cenu. Ak technik potrebuje nejaký útlm R, zapojí za sebou vhodné útlmové články tak, aby súčet ich odporov bol presne R. Aby sa minimalizovala cena (resp. počet článkov), technik používa takúto stratégiu:
   (1) vyber článok s najväčším možným odporom r takým, že $r \leq R$;
   (2) ak $R\text{-}r > 0$, pokračuj v hľadaní výberu článkov pre odpor $R\text{-}r$ podľa bodu (1) s tým, že hľadáme útlmové články vytvárajúce celkový odpor $R\text{-}r$.

   Za predpokladu, že z útlmových článkov je možné poskladať celkový odpor R, je pravdou, že uvedená stratégia vždy nájde výber s minimálnym počtom útlmových článkov? Ak áno, napíšte zdôvodnenie. Ak nie, nájdite také typy útlmových článkov a požadovaný odpor R, aby aplikovanie stratégie neviedlo k minimálnemu počtu použitých útlmových článkov.

V úlohách 5-7 budeme uvažovať nasledujúci <u>neorientovaný ohodnotený</u> graf:



5. (1.5b) Ak prehľadávanie do hĺbky je inicializované vo vrchole A a susedné vrcholy „spracúvame" vždy v abecednom poradí, zapíšte postupnosť vrcholov grafu, v akej budú navštívené:

____

6. (1b) Určte také ohodnotenie (cenu) pre hranu CD, aby bola vybraná Kruskalovým algoritmom do minimálnej kostry.

____

7. (1.5b) Zapíšte postupnosť hrán grafu, v akej budú pridané jednotlivé hrany grafu do minimálnej kostry pri aplikovaní Kruskalovho algoritmu pri takej voľbe x, kedy je hrana CD vybraná do minimálnej kostry:

____

8. (2.5b) Kľúčovým prvkom Knuth-Morris-Prattovho algoritmu je skip matica, ktorá pre aktuálnu pozíciu vo vzorke a písmeno na vstupe rozhoduje, o koľko pozícií sa má posunúť začiatok vzorky vo vstupe (narozdiel od naivného algoritmu, kde sa pri nezhode posúva začiatok vzorky vždy o 1 pozíciu doprava). Vyplňte skip maticu pre zadanú vzorku a množinu písmen vzorky. Stĺpce odpovedajú aktuálnej pozícii vo vzorke a riadky písmenu, ktoré je na vstupe.

	A	A	B	A	A
A		0			
B					
iné písmeno			3		

9. (4b) Označte pravdivosť tvrdení (A – pravda, N – nepravda):

☐ Dijkstrov algoritmus narozdiel od Bellman-Fordovho funguje korektne iba v prípade, že ohodnotenia hrán sú nezáporné.

☐ Každý program využívajúci rekurziu má vždy aspoň exponenciálnu zložitosť.

☐ Uvažujme prehľadávanie súvislého grafu algoritmom do šírky resp. do hĺbky, ktoré štartuje v nejakom vrchole r. Hrany, ktorými sme nejaký vrchol navštívili po prvý krát tvoria kostru. Ak zoberieme vrchol r ako koreň kostry, potom kostra prehľadávania do šírky má výšku, ktorá je menšia alebo rovná ako výška kostry prehľadávania do hĺbky.

☐ Základná myšlienka Bellman-Fordovho algoritmu je v orientovanom grafe s n vrcholmi n-1 krát postupne vykonať relaxáciu všetkých orientovaných hrán grafu. Vykonávať všetkých n "fáz" relaxácií všetkých hrán grafu nie je potrebné. Výpočet môžeme ukončiť ihneď po vykonaní takej "fázy" relaxácií všetkých hrán grafu, v ktorej sa pre žiaden vrchol neznížil odhad ohodnotenia najkratšej cesty zo štartovacieho vrcholu do tohto vrcholu.

☐ Pre každý súvislý ohodnotený graf, v ktorom majú všetky hrany navzájom rôzne ceny, platí, že má vždy práve jednu najlacnejšiu (minimálnu) kostru.

☐ V Kruskalovom algoritme hrany vybrané do minimálnej kostry tvoria vždy (počas celého vykonávania algoritmu) súvislý podgraf.

☐ Greedy algoritmy sú založené na myšlienke uchovania si riešení podproblémov, ktoré sme už riešili, a ich použití pri ich opakovanom výskyte.

☐ Ak v prehľadávaní do hĺbky ohodnoteného grafu budeme navštevovať susedné vrcholy aktuálneho vrcholu v poradí podľa ceny hrán (t.j. v každom kroku vyberieme nenavštívený susedný vrchol taký, že cena hrany, ktorá spája aktuálny a susedný vrchol, je najmenšia možná), potom vytvorená DFS kostra je zároveň najlacnejšou kostrou.