Záverečný test
teoretická časť

Ústav informatiky
Prírodovedecká fakulta
UPJŠ v Košiciach

Píšte prosím čitateľne!

Hodnotenie, vyplní opravujúci:

Meno a priezvisko:	Skupina PAZ:	

1/1	2/2	3/1,5	4/1,5	5/2	6/1	7/3,5	8/1,5	9/3	10/2	11/2	12/2	∑23

1. (1b) Uvažujme Floyd-Warshallow algoritmus na hľadanie najlacnejších ciest v grafe. Pre ktoré priradenia premenných i, j, k symbolom vo for-cykloch bude algoritmus fungovať korektne, t.j. korektne sa naplní obsah matice d?

   (a) Δ=k,▼=i,☺=j Δ=k,▼=j,☺=i Δ=i,▼=j,☺=k
   (b) Δ=k,▼=i,☺=j Δ=i,▼=j,☺=k Δ=j,▼=k,☺=i
   (c) Δ=k,▼=i,☺=j a Δ=k,▼=j,☺=i
   (d) len Δ=k,▼=i,☺=j
   (e) len Δ=k,▼=i,☺=j a Δ=i,▼=j,☺=k

```
for (int Δ = 0; Δ < n; Δ++)
  for (int ▼ = 0; ▼ < n; ▼++)
    for (int ☺= 0; ☺< n; ☺++)
      if (d[i, k] + d[k, j] < d[i, j])
        d[i, j] = d[i, k] + d[k, j];
```

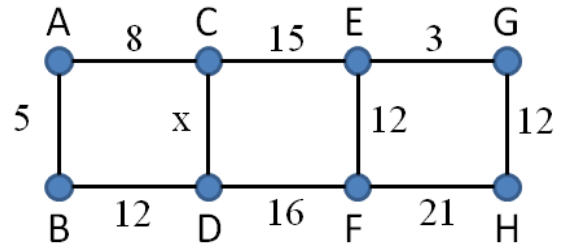
2. (2b) Nakreslite súvislý neorientový graf s 5 hranami a 5 vrcholmi (označenými písmenami A-E), ktorého prechod prehľadávaním do šírky aj prechod prehľadávaním do hĺbky inicializovanými z vrcholu A prvý krát navštívia všetky vrcholy v rovnakom poradí. Predpokladáme, že susedné vrcholy „spracúvame“ v abecednom poradí.

3. (1,5b) Nech S je množina intervalov $S = \{I_1, I_2, I_3, \dots, I_n\}$, pričom $I_k = \langle a_k, b_k \rangle$. Veľkosť intervalu definujeme ako $|I_k| = b_k - a_k$. Chceme vybrať najväčšiu (počtom prvkov - intervalov) takú podmnožinu Q množiny S, $Q \subseteq S$, že pre ľubovoľné 2 prvky $I_i, I_k \in Q$ platí $I_i \cap I_k = \emptyset$. Teda žiadne 2 intervaly z Q nemajú spoločný prienik. Uvažujme nasledovný greedy algoritmus:

   $Q \leftarrow \emptyset$;
   kým (S nie je prázdna) {
      vyber najmenší interval $I \in S$, t.j. $|I|$ je najmenšie spomedzi intervalov v S;
      presuň I z množiny S do Q, t.j. $Q \leftarrow Q \cup \{I\}$ a $S \leftarrow S \setminus \{I\}$;
      odstráň z množiny S všetky intervaly, ktoré majú neprázdny prienik s I;
   }

   Dokážte, že algoritmus vráti korektný výsledok alebo nájdite kontrapríklad.

Graf k úlohám 4-7. Pozn.: štartovacie/inicializačné vrcholy sú v rôznych úlohách rôzne.



4. (1,5b) Pre aké celočíselné hodnoty x bude hrana CD za každých podmienok vybraná Primovým aj Kruskalovým algoritmom do najlacnejšej kostry?

_____

5. (2b) Určte množinu hrán, ktoré vyberie Primov algoritmus na nájdenie najlacnejšej kostry so štartovacím vrcholom H a $x=11$. **Hrany** zapíšte v poradí, v akom ich tento algoritmus pridá do konštruovanej kostry.

_____

6. (1b) Ak prehľadávanie do hĺbky je inicializované vo vrchole D a susedné vrcholy „spracúvavame“ vždy v abecednom poradí, zapíšte postupnosť vrcholov grafu, v akej budú navštívené:

_____

7. Uvažujme beh Dijkstrovho algoritmu pričom hľadáme najlacnejšie cesty z vrcholu A a $x=11$. (1,5b) V akom poradí sa jednotlivé vrcholy stanú „vybavenými“?

_____

(1b) Dijkstrov algoritmus vieme upraviť tak, aby sme si pre každý vybavený vrchol zapamätali jeho predchodcu (vrchol, z ktorého sme „prišli“ do daného vrcholu). Pre každý jeden vrchol v nasledujúcej tabuľke vyplňte jeho predchodcu na najkratšej ceste z vrcholu A.

(1b) V tabuľke nižšie pre každý vrchol určte hodnotu horného ohraničenia ceny najkratšej cesty v okamihu, keď sa tento vrchol stane „vybaveným“.

	A	B	C	D	E	F	G	H
predch.								
d								

8. (1,5b) Spoločnosti inzerujúce svoje produkty a služby v rádiu si pripravili n typov reklám s dĺžkami $t[0], t[1], \ldots, t[n-1]$. Programového manažéra zaujíma, či je možné presne zaplniť z týchto typov reklám A sekúnd určených na reklamu, t.j., či je možné vybrať reklamy tak, aby súčet ich dĺžok bol presne A pričom reklamy sa môžu opakovať (realita: v nočnom vysielaní ide tá istá reklama aj viac krát po sebe). Označme $D[i]$ logickú hodnotu (true/false) vyjadrujúcu, či je možné z pripravených reklám zaplniť vysielací čas dĺžky i. Zjavne $D[0] = true$ a programového manažéra zaujíma $D[A]$. Napíšte fragment kódu, ktorý pre $i > 0$ korektne vypočíta hodnotu $D[i]$ za predpokladu, že poznáte hodnoty $D[j]$, kde $0 \leq j < i$.

9. (3b) Označte pravdivosť tvrdení (A – pravda, N – nepravda, +0.5b za správnu odpoveď, -0.5b za nesprávnu odpoveď, 0b za žiadnu odpoveď):

☐ (A) Majme algoritmus, ktorý pre ohodnotený neorientovaný graf, štartovací vrchol a cieľový vrchol vypíše najkratšiu cestu medzi nimi (postupnosť vrcholov). Ak zvýšime ohodnotenia všetkych hrán v grafe o 10, tak algoritmus vždy vypíše tú istú cestu.

☐ (B) Z terminológie neohodnotených grafov vieme, že excentricita vrcholu grafu je vzdialenosť od neho k najvzdialenejšiemu vrcholu a centrum grafu je množina vrcholov s minimálnou excentricitou. Potom pre všetky vrcholy X, Y z centra grafu platí $(X, Y) \in E(G)$.

☐ (C) Ak sa v grafe každá hodnota váhy hrany nachádza nanajvýš raz, potom kostra získaná pomocou Primovho algoritmu obsahuje tie isté hrany ako kostra získaná pomocou Kruskalovho algoritmu.

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

☐ (D) Memoizácia (pamätanie si a znovupoužitie výsledkov podvýpočtov) umožňuje znížiť časovú zložitosť každého rekurzívneho výpočtu.

☐ Časová zložitosť Floyd-Warshallovho algoritmu je $O(n^3)$, kde n je počet vrcholov grafu a nezáleží od počtu hrán grafu.

10. (2b) Zdôvodnite svoju odpoveď ku ktorémukoľvek tvrdeniu A-D z predošlej úlohy. Odôvodňované tvrdenie označte zakrúžkovaním písmena.

11. Implementáciou `HashSet`-u sme sa zaoberali na prednáške. Každému reťazcu bol priradený jeho index hash-ovacou funkciou ako súčet unicode hodnôt znakov modulo veľkosť poľa. Predpokladajme, že zmeníme hash-ovaciu funkciu, ktorá index vypočíta ako dĺžku reťazca modulo veľkosť poľa.

    (1b) Uvažujeme uvedenú implementáciu s poľom veľkosti 4. Napíšte 3 slová ktoré budú mať vzájomné kolízie a dĺžky slov budú rozdielne.

    _____

    (1b) Koľko slov treba vložiť do uvedenej implementácie `HashSet`-u, ak má interne pole veľkosti 16 a na začiatku je prázdny, aby bol load factor práve 2?

    _____

12. (2b) Majme pole s kapacitou, do ktorého vieme pridávať a odoberať prvky podľa dole uvedených pravidiel. Dokážte alebo vyvráťte, že zložitosť ľubovoľných n po sebe vykonaných operácii vkladania a odoberania má zložitosť $O(n)$.

    Pravidlá:
    Prvok vieme odobrať alebo pridať iba z/na koniec poľa. Ak pridávame prvok a postačuje kapacita poľa, tak prvok pridáme. Ak pridávame prvok a nepostačuje kapacita poľa, tak vytvoríme nové pole väčšie o 5 prvkov, obsah starého poľa skopírujeme a pridáme prvok na koniec poľa. Pri odstraňovaní prvkov z poľa, ak je po odobraní prvku z poľa aktuálna veľkosť oproti kapacite menšia ako 5, tak vytvoríme nové pole, ktoré má veľkosť o 5 prvkov menšiu a postupujeme analogicky.