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.

Sudoku (10+3 bodov, backtracking)

Poznáte hru Sudoku? Cieľom hry je do prázdnych políčok mriežky 9 x 9 doplniť čísla od 1 po 9 tak, aby:

- v každom riadku sa každé číslo nachádzalo práve raz,
- v každom stĺpci sa každé číslo nachádzalo práve raz,
- v každej vyznačenej „podmriežke“ 3 x 3 sa každé číslo nachádzalo práve raz.

		1					9	7
	9			7	5			6
5						4		
6	8		9					1
			8	2	4			
7					6		8	3
		7						9
1			7	8			2	
9	3					1		

Väčšina ľudí túto hru považuje za náročnú. Nie však absolventi predmetu PAZ1b. Pre nich je to nudná hra, keďže vedia ľahko vytvoriť program, ktorý akékoľvek zadanie hry vyrieši.

Zadanie: Vytvorte program, ktorý z textového súboru načíta plán hry Sudoku. Prázdne políčka môžete napríklad „kódovať“ symbolom X alebo číslom 0. Program následne vygeneruje riešenie pre načítaný plán hry.

Deň D (4+8 bodov, grafové algoritmy)

Súostrovie PAZ Oceánia sa skladá z n ostrovov číslovaných $0 \dots n-1$. PAZ Oceánia získala grant na vybudovanie pravidelných obojsmerných lodných spojení medzi ostrovmi. Už je známy aj harmonogram, v akom sa jednotlivé spojenia budú postupne otvárať. Každý deň je v pláne otvoriť práve jedno nové (obojsmerné) lodné spojenie. Obyvatelia PAZ Oceánie sa už nevedia dočkať dňa, kedy sa bude dať cestovať medzi ľubovoľnými dvoma ostrovmi súostrovia.

Zadanie: V textovom súbore je zadané číslo n - počet ostrovov PAZ Oceánie. V každom z ďalších riadkov sa nachádza dvojica medzerou oddelených čísel x a y, ktoré označujú vytvorené obojsmerné lodné spojenie medzi ostrovmi s číslami x a y. Spojenia v súbore sú uložené chronologicky, t.j. podľa času, kedy sa uvedú do prevádzky. Vytvorte program, ktorý vráti, v koľký deň podľa harmonogramu sa bude dať po prvý krát cestovať medzi ľubovoľnými dvoma ostrovmi súostrovia (potenciálne aj s prestupmi).

Bodovanie: 4b za riešenie v polynomiálnom čase + 8b za riešenie v čase $O(n^2)$.

Návraty k midtermu (5 bodov)

Vieme, že platí toto tvrdenie: **„***V čase* $O(n.logn)$ *môžeme zistiť, ktorá hodnota sa v n-prvkovom poli vyskytuje najväčší počet krát.*“

Naprogramujte metódu, ktorá v uvedenom (aj priemernom) čase nájde tú hodnotu v poli, ktorá sa v ňom vyskytuje najväčší počet krát. Ak je takých hodnôt viac, metóda nech vráti ľubovoľnú z nich. Metóda nesmie modifikovať referencované pole p. Akceptovaná pamäťová zložitosť je $O(n)$.

```
public int najcastejsiaHodnota(int[] p)
```

Stromovač (15 bodov, stromy)

Uvažujme triedu `Uzol` z prednášky o binárnych stromoch:

```
public class Uzol {
	private int hodnota;
	private Uzol lavy;
	private Uzol pravy;

	public Uzol(int hodnota, Uzol lavy, Uzol pravy) {
		this.hodnota = hodnota;
		this.lavy = lavy;
		this.pravy = pravy;
	}

	public void vypisStrom(int uroven) {
		if (lavy != null) {
			lavy.vypisStrom(uroven + 1);
		}

		System.out.println(uroven + " " + hodnota);

		if (pravy != null) {
			pravy.vypisStrom(uroven + 1);
		}
	}

	public int getHodnota() {
		return hodnota;
	}

	public void setHodnota(int hodnota) {
		this.hodnota = hodnota;
	}
}
```

Ak zavoláme metódu `vypisStrom(0)` na koreni stromu, dostaneme textový zápis štruktúry a obsahu tohto stromu. V tomto zápise je každý vrchol charakterizovaný svojou hĺbkou (vzdialenosťou od koreňa) a hodnotou, pričom postupnosť týchto hodnôt je výsledkom inorder prechodu binárnym stromom.

Do triedy `Uzol` pridajte statickú metódu `vytvorStrom`, ktorá vráti referenciu na koreň novovytvoreného stromu podľa zadaného textového súboru. Vytvorený strom má mať tú vlastnosť, že ak na jeho koreni zavoláme metódu `vypisStrom(0)` dostaneme výpis

totožný s obsahom tohto textového súboru. Môžete predpokladať, že súbor obsahuje korektný vstup.

```
public static Uzol vytvorStrom(File suborSPopisomStromu)
```

Najdlhšia palindromická podpostupnosť (7+7 bodov, dynamické programovanie)

Hľadanie najdlhších podpostupností a palindrómové veci sú už také klasické úlohy. Prečo to ale nespojiť a neskúsiť nájsť najdlhšiu vybranú podpostupnosť zadaného reťazca, ktorá je palindrómom? Uvažujme reťazec `cacabakla`. Tento reťazec má mnoho podpostupností (napr. `ccaba`, `acakla`, …). Nás ale zaujímajú palindromické podpostupnosti (napr. `aca`, `aaaa`, `aka`, …). V reťazci `cacabakla` je však najdlhšia palindromická podpostupnosť reťazec `aabaa` (viď: `cacabakla`).

Úloha: S využitím dynamického programovania naprogramujte metódu, ktorá pre zadané slovo nájde jeho najdlhšiu palindromickú podpostupnosť.

Hodnotenie: 7 bodov nájdenie dĺžky najdlhšej palindromickej podpostupnosti, 7 bodov za nájdenie tejto podpostupnosti. Očakáva sa polynomiálna časová zložitosť riešenia.

Návod: Uvažujme reťazec $r_1 r_2 r_3 \dots r_n$ dĺžky n. Označme si $P[i,k]$ dĺžku najdlhšej palindromickej podpostupnosti reťazca $r_i r_{i+1} r_{i+2} \dots r_k$. Triviálne $P[i,i] = 1$. Vo všeobecnosti platí:

$$P[i,k] = \begin{cases} \max\{P[i+1,k], P[i,k-1]\}, & r_i \neq r_k \\ P[i+1,k-1] + 2, & r_i = r_k \end{cases}$$