

## 2.test DSA 12.5.2017 Jméno:

---

### 1. Příklad na indukci

Dokažte matematickou indukci, že souvislý graf o  $n$  hranách a  $n+1$  uzlech je strom.

-----

#### Řešení:

Stroj je souvislý acyklický graf. Podle předpokladu je graf souvislý, stačí tedy dokázat, že takový graf o  $n$  hranách a  $n+1$  uzlech je acyklický.

Počáteční krok:  $n = 1$ , platí, že graf, který má právě jednu hranu a právě dva uzly je acyklický.

Indukční krok: předpokládáme platnost pro  $n$  a dokazujeme platnost pro  $n+1$ .

Předpoklad vyjádříme: souvislý graf o  $n$  hranách a  $n+1$  uzlech je acyklický. Přidáme-li další hranu a graf bude mít  $n+1$  hran, pak aby zůstal acyklickým souvislým grafem, musíme přidat nejméně jeden uzel, jinak by tato hrana byla součástí cyklu – v acyklickém grafu je mezi dvěma uzly právě jedna jednoduchá cesta, nová hrana by představovala alternativní cestu a dohromady by tvořily cyklus.

Novou hranu připojíme k novému uzlu a výsledný graf má  $n+2$  uzlů.

**q.e.d.**

### 2. Příklad na pravděpodobnost

Jaká je pravděpodobnost, že funkce  $f$  vrátí `true`? Předpokládejte, že pole  $a$  i  $b$  obsahují náhodnou permutaci prvků  $1 : : n$  ( $a$  tedy mají shodnou délku). Svoji odpověď zdůvodněte.

```
boolean f(int[] a, int[] b) {
    int r1 = random(0, a.length - 1); // funkce random vrati nahodne cislo
                                     // mezi 0 a a.length - 1 vctne
    int r2 = random(0, b.length - 1);
    return a[r1] <= b[r2];
}
```

-----

#### Řešení:

Označme si délku polí  $a.length = b.length = n$ . Pravděpodobnost, že na indexu  $r1$  v poli  $a$  je hodnota  $k$ , je  $1/n$ . Pravděpodobnost, že na indexu  $r2$  v poli  $b$  je hodnota  $l$ , je  $1/n$ . Oba jevy jsou na sobě nezávislé, takže pravděpodobnost, že  $a[r1] \leq b[r2]$ , je:

$$\sum_{k=1}^n \sum_{l=k}^n \frac{1}{n} \cdot \frac{1}{n} = \frac{(n+1) \cdot n}{2} \cdot \frac{1}{n} \cdot \frac{1}{n} = \frac{n+1}{2n}$$

**q.e.d.**

### 3. Příklad na rozptylování (hashing)

Uvažte rozptylovací tabulku rozměru  $M = 7$  a následující rozptylovací funkci  $h: h(k) = k \bmod M$ .

Uvažte dále, že pro ukládání do tabulky používáte lineární prohledávání a odpovídající funkci je  $h1(k, i) = (h(k) + i) \bmod M$ , kde  $i$  je pořadí pokusu. Nakreslete obsah tabulky po vložení posloupnosti hodnot: 7, 5, 21, 10, 1, 14, 8.

-----

#### Řešení:

Postup plnění tabulky lze zachytit následující řadou:

7						
---	--	--	--	--	--	--

Pokus č.1:  $((7 \bmod 7) + 1) \bmod 7 = 1$

7					5	
---	--	--	--	--	---	--

Pokus č.1:  $((5 \bmod 7) + 1) \bmod 7 = 6$

7	21				5	
---	----	--	--	--	---	--

Pokus č.1:  $((21 \bmod 7) + 1) \bmod 7 = 1$  – kolize

Pokus č.2:  $((21 \bmod 7) + 2) \bmod 7 = 2$

7	21		10		5	
---	----	--	----	--	---	--

Pokus č.1:  $((10 \bmod 7) + 1) \bmod 7 = 4$

7	21		10		5	
---	----	--	----	--	---	--

Pokus č.1:  $((1 \bmod 7) + 1) \bmod 7 = 2$  – kolize

Pokus č.2:  $((1 \bmod 7) + 2) \bmod 7 = 3$

7	21	1	10	14	5	
---	----	---	----	----	---	--

Pokus č.1:  $((14 \bmod 7) + 1) \bmod 7 = 1$  – kolize

Pokus č.2:  $((14 \bmod 7) + 2) \bmod 7 = 2$  – kolize

Pokus č.3:  $((14 \bmod 7) + 3) \bmod 7 = 3$  – kolize

Pokus č.4:  $((14 \bmod 7) + 4) \bmod 7 = 4$  – kolize

Pokus č.5:  $((14 \bmod 7) + 5) \bmod 7 = 5$

7	21	1	10	14	5	
---	----	---	----	----	---	--

Pokus č.1:  $((14 \bmod 7) + 1) \bmod 7 = 1$  – kolize

Pokus č.2:  $((14 \bmod 7) + 2) \bmod 7 = 2$  – kolize

Pokus č.3:  $((14 \bmod 7) + 3) \bmod 7 = 3$  – kolize

Pokus č.4:  $((14 \bmod 7) + 4) \bmod 7 = 4$  – kolize

Pokus č.5:  $((14 \bmod 7) + 5) \bmod 7 = 5$

7	21	1	10	14	5	8
---	----	---	----	----	---	---

Pokus č.1:  $((8 \bmod 7) + 1) \bmod 7 = 2$  – kolize

Pokus č.2:  $((8 \bmod 7) + 2) \bmod 7 = 3$  – kolize

Pokus č.3:  $((8 \bmod 7) + 3) \bmod 7 = 4$  – kolize

Pokus č.4:  $((8 \bmod 7) + 4) \bmod 7 = 5$  – kolize

Pokus č.5:  $((8 \bmod 7) + 5) \bmod 7 = 6$  – kolize

Pokus č.6:  $((8 \bmod 7) + 6) \bmod 7 = 7$

**q.e.d.**

## 4. Příklad na implementaci haldy

Naimplementujte následující funkce haldy (kód je v Javě, indexujte od 0):

```
class MaxHeap {
    int[] heap; // pole reprezentující haldu celých čísel
    int heap_size; // počet prvků haldy

    // Kontrolujte korektnost parametrů metod. V případě chyby
    // vyhoďte výjimku, nebo alespoň vypište chybovou hlášku.

    int right(int i) { // Vrátí pravého potomka uzlu i.
        // váš kód
    }

    int left(int i) { // Vrátí levého potomka uzlu i.
```

```

// váš kód

}

boolean isHeap(int i) { // Vrátí true, právě když je podstrom
                        // s kořenem i korektní max-haldou.
// váš kód

}
}

```

---

### Řešení:

```

class MaxHeap {
    int[] heap; // pole reprezentující haldu celých čísel
    int heap_size; // počet prvků haldy

    // Kontrolujte korektnost parametrů metod. V případě chyby
    // vyhodte výjimku, nebo alespoň vypište chybovou hlášku.

    int right(int i) { // Vrátí pravého potomka uzlu i.
        // váš kód

    }

    int left(int i) { // Vrátí levého potomka uzlu i.
        // váš kód

    }

    boolean isHeap(int i) { // Vrátí true, právě když je podstrom
                            // s kořenem i korektní max-haldou.
        // váš kód

    }
}
q.e.d.

```

## 5. Příklad na implementaci řazení

Naimplementujte metodu `countingSort` pro vzestupné řazení celých čísel v intervalu  $0 \dots \text{MAX}-1$ . Plný počet bodů dostanete za stabilní verzi, za nestabilní budete hodnoceni polovinou.

```
int MAX = ...;

int[] countingSort(int[] a) {
    // váš kód

}
```

---

### Řešení:

```
int MAX = ...;

int[] countingSort(int[] a) {
    // váš kód

}
```

q.e.d.