Kontrolní úlohy po druhém prezenčním setkání
KIKM/ALGD – Algoritmy a datové struktury
Jiří Škoda – skodaji1@uhk.cz
Fakulta informatiky a managementu, Univerzita Hradec Králové
Hradec Králové, 2021

---

1. Vytvořte algoritmus, který z pole, ve kterém jsou uložená celá čísla, odebere všechny sudé prvky. Toto odebírání bude probíhat v rámci tohoto pole, bez pomocného pole. (Pozn.: když narazíte na sudý prvek, všechny ostatní prvky posunete o jednu pozici vpřed a pole jakoby zkrátíte o 1.)

ŘEŠENÍ

```
for (int i = 0; i < array.length; i++)
{
    if (array[i] % 2 == 0)
    {
        for (int j = i; j < (array.length - 1); j++)
        {
            array[j] = array[j + 1];
        }
        i--;
    }
}
```

Časová složitost: $\mathcal{O}(n^2)$

2. V poli jsou uložená celá čísla. Vytvořte algoritmus, který za každý lichý prvek vloží 1. (Bez pomocného pole. Nezapomeňte si na začátek pole deklarovat dost dlouhé, abyste měli, kde vkládat nové hodnoty.)

ŘEŠENÍ

```
// Predpoklad: pole ma dostatecne velkou znamou delku
for (int i = 0; i < array.length; i++)
{
    if (pole[i] % 2 == 1)
    {
        for (int j = (array.length - 2); j > i; j--)
        {
            array[j + 1] = array[j];
        }
        array[i + 1] = 1;
    }
}
```

Časová složitost: $\mathcal{O}(n^2)$

3. Doplňte následující část algoritmu, který určí součin lichých členů posloupnosti, jejichž index je dělitelný 3.

```
{
    …
    int soucin = ..........;
    int i;
    i = ..........;
    while (i .......... n)
    {
        if (..........)
        {
            soucin = soucin * a[i];
        }
        i = ..........;
    }
}
```

ŘEŠENÍ

```
{
    …
    int soucin = 1
    int i;
    i = 3;
    while (i < a.length)
    {
        if (a[i] % 2 == 1)
        {
            soucin = soucin * a[i];
        }
        i = i + 3;
    }
}
```

Časová složitost: $\mathcal{O}(n)$

4. Odkrokujte následující kód:

```
…
int n = 6;
int m = -1;
for (i = 0; i < n; i++)
{
    if (a[i] % 2 == 1)
    {
        m = m +1;
        b[m] = a[i];
    }
}
…
```

ŘEŠENÍ

i	a[i]	a[i] % 2 == 1	m
0	4	false	-1
1	7	true →	0
2	8	false	
3	4	false	
4	5	true →	1
5	2	false	

Pole	Index	0	1	2	3	4	5
a	Vstupní hodnota	4	7	8	4	5	2
b	Výstupní hodnota	7	5				

Časová složitost: $\mathcal{O}(n)$