EVROPSKÁ UNIE
Evropské strukturální a investiční fondy
Operační program Výzkum, vývoj a vzdělávání

MINISTERSTVO ŠKOLSTVÍ, MLÁDEŽE A TĚLOVÝCHOVY

Programování a algoritmizace

Bubble Sort

Strategický projekt UTB ve Zlíně, reg. č. CZ.02.2.69/0.0/0.0/16_015/0002204

Ing. et Ing. Erik Král, Ph.D.
ÚPKS
Fakulta aplikované informatiky

Obsah

Bubble sort

Popis implementace

Úvod

- V následujících snímcích probereme algoritmus Bubble Sort.
- Na těchto příkladech si demonstrujeme práci s jednorozměrným polem s pevnou délkou [1], vnořené cykly *for* a ukončení cyklu pomocí příkazu *break* [2].

Bubble Sort

- Algoritmus Bubble Sort představuje jednoduchý algoritmus pro seřazení prvků v poli.
- Reálně se nepoužívá kvůli složitosti $O(n^2)$ [3], ale slouží k pochopení algoritmizace.
- Postupně procházíme pole a porovnáváme dva sousedící prvky a pokud nejsou seřazené v žádaném pořadí, tak je prohodíme a posuneme se na další prvek.
- Protože se v každé iteraci dostane prvek s nejvyšší hodnotou na konec neseřazeného pole, tak můžeme v každé iteraci skončit o jeden prvek dříve.

Animace Bubble Sort [4]

Bubble Sort příklad

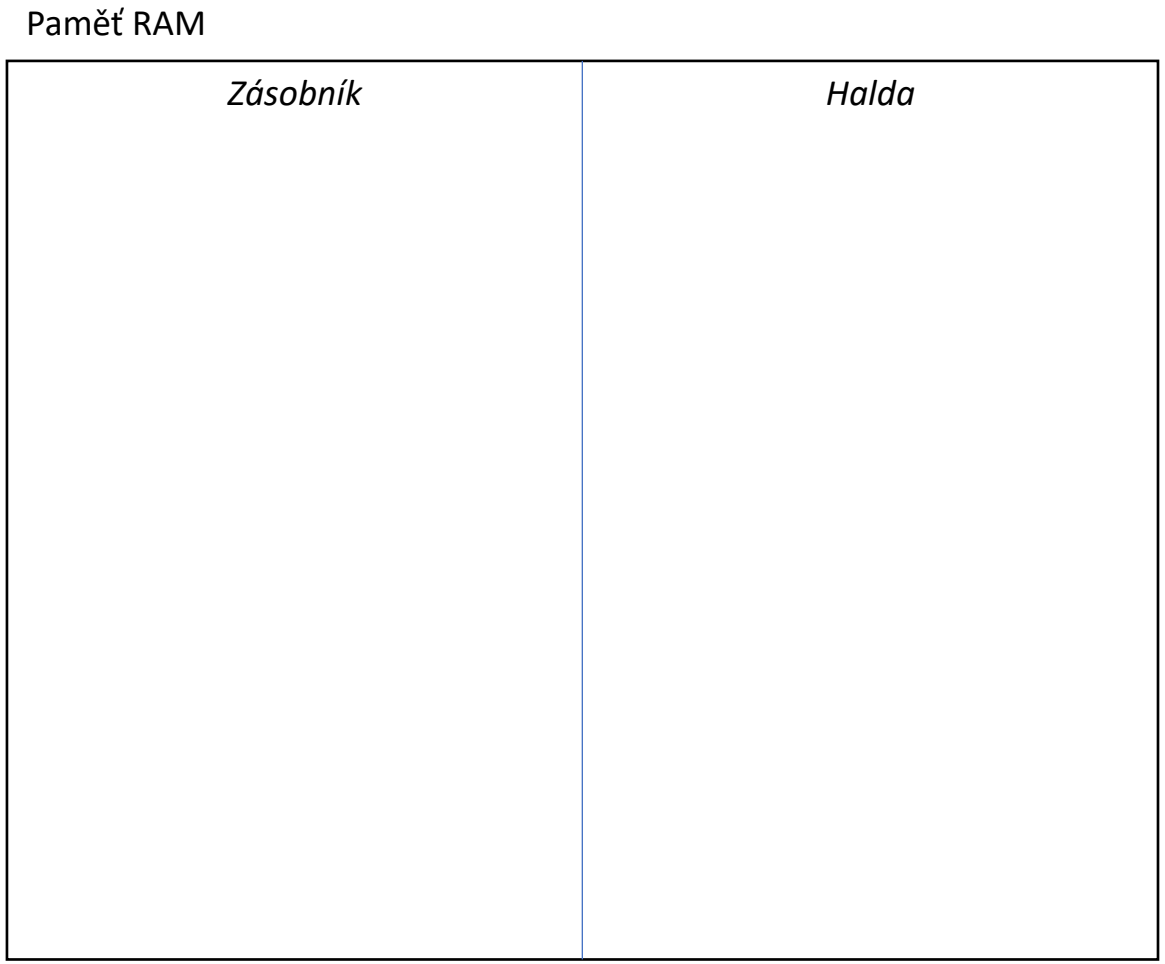
- V následujícím příkladu uvedeme příklad na Bubble Sort se stejnými hodnotami prvků jako ve videu na předcházejícím snímku.
- Protože jsou prvky po šesté iteraci už seřazené, tak už je zbytečné v algoritmu pokračovat a vyhledávání ukončíme když jsou prvky už seřazené.
- Také v každé iteraci skončíme o jeden prvek v poli dříve, protože v každé iteraci se na konec dostane největší prvek ze zatím neseřazené části pole.

Algoritmus a paměť

- Algoritmus si alokuje paměť pro parametry, lokální proměnné a další hodnoty na zásobníku (Stack) a pro dynamicky alokované objekty alokuje paměť na haldě (Heap).
- V příkladech je **zjednodušeně** demonstrováno využití paměti z hlediska zásobníku a haldy.
- Práce se zásobníkem je ve skutečnosti složitější a v příkladech jsou zobrazeny **pouze proměnné přímo související s algoritmem** a jsou vynechány uložené hodnoty registrů nebo návratové hodnoty. Také pořadí předávaných argumentů a parametrů metody může být jiné.

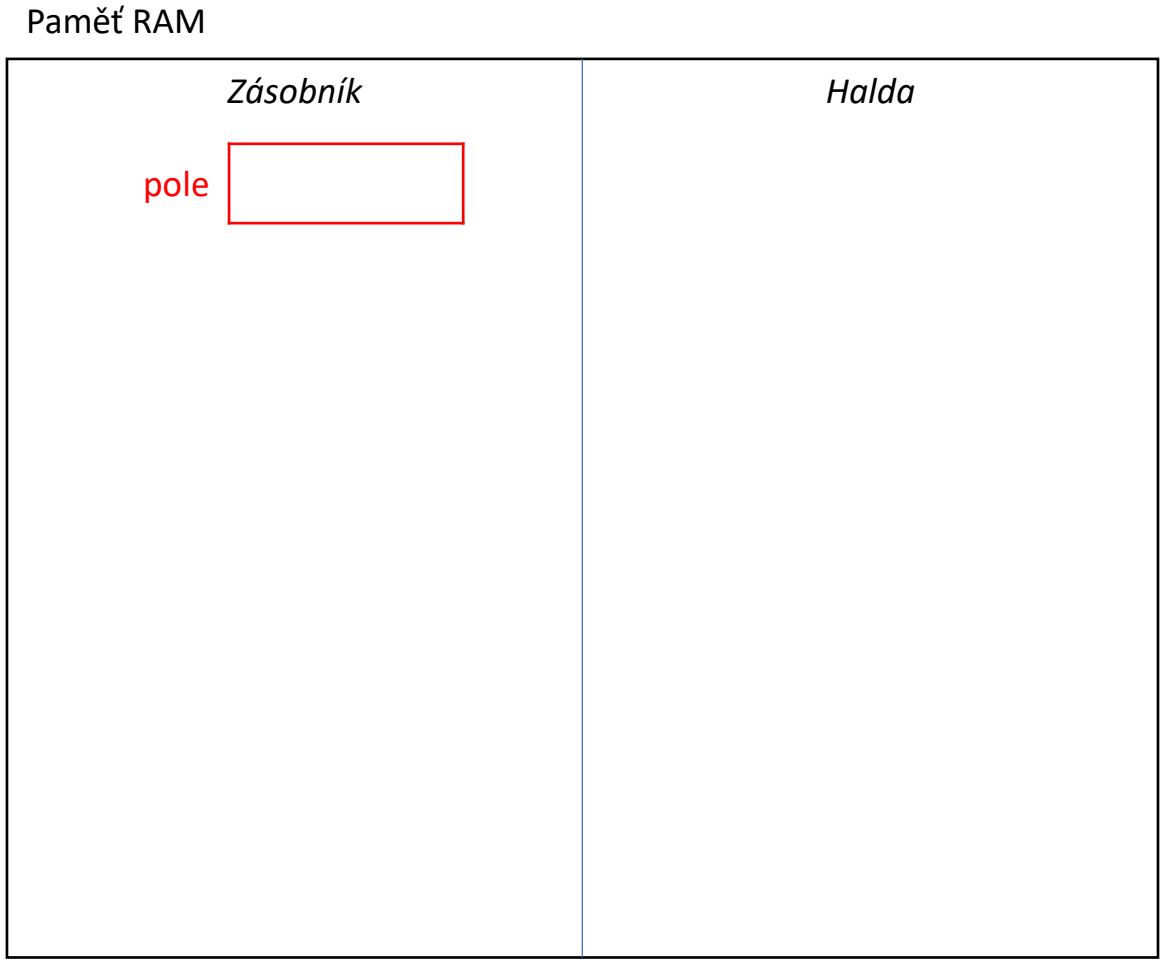
Bubble Sort

```
int[] pole = new int[]
{ 8, 4, 10, 3, 6, 1, 9, 2, 7, 5 };

for (int i = 0; i < pole.Length; i++)
{
    bool serazeno = true;

    for (int j = 0; j < pole.Length - 1 - i; j++)
    {
        if (pole[j] > pole[j + 1])
        {
            int tmp = pole[j];
            pole[j] = pole[j + 1];
            pole[j + 1] = tmp;

            serazeno = false;
        }
    }

    if (serazeno) break;
}
```



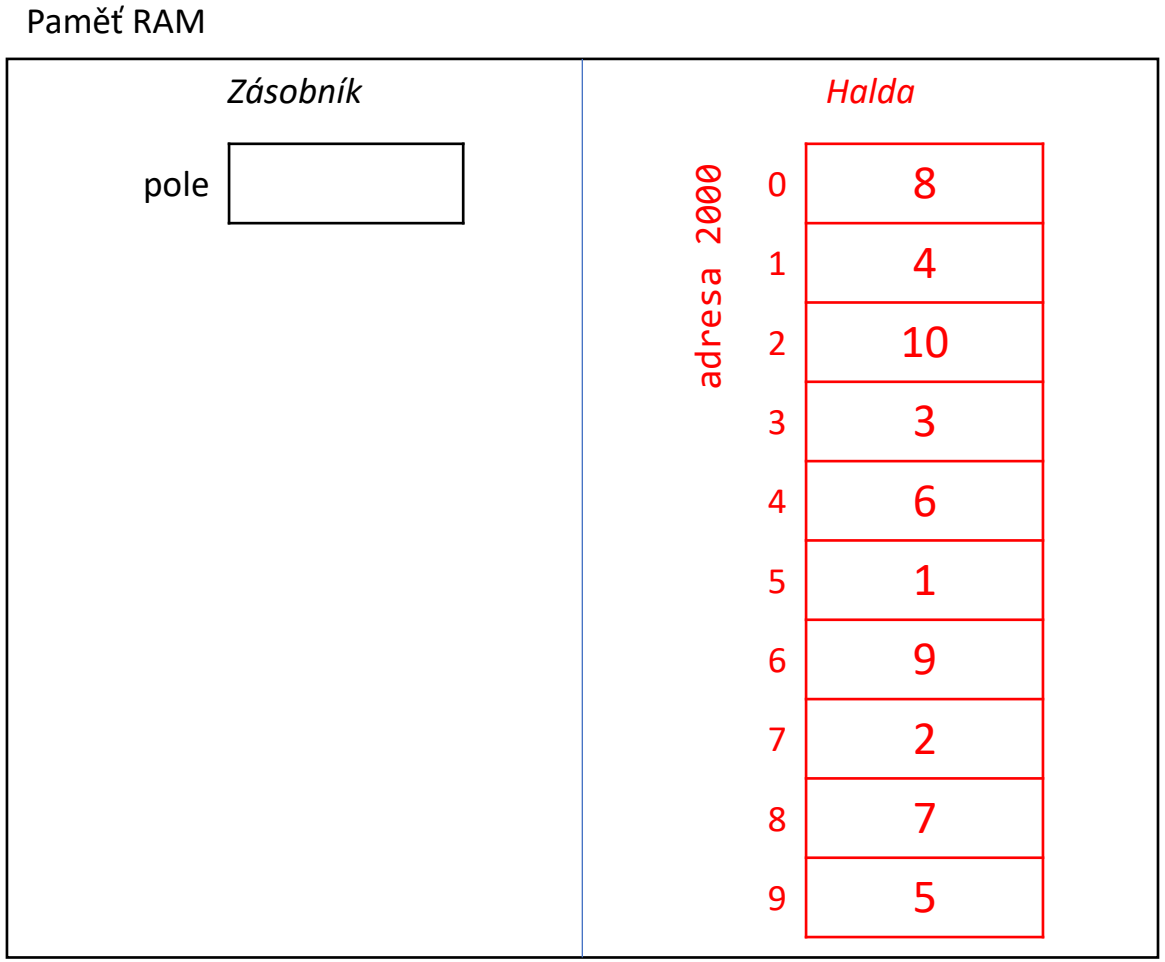
Bubble Sort

```
int[] pole = new int[]
{ 8, 4, 10, 3, 6, 1, 9, 2, 7, 5 };
```



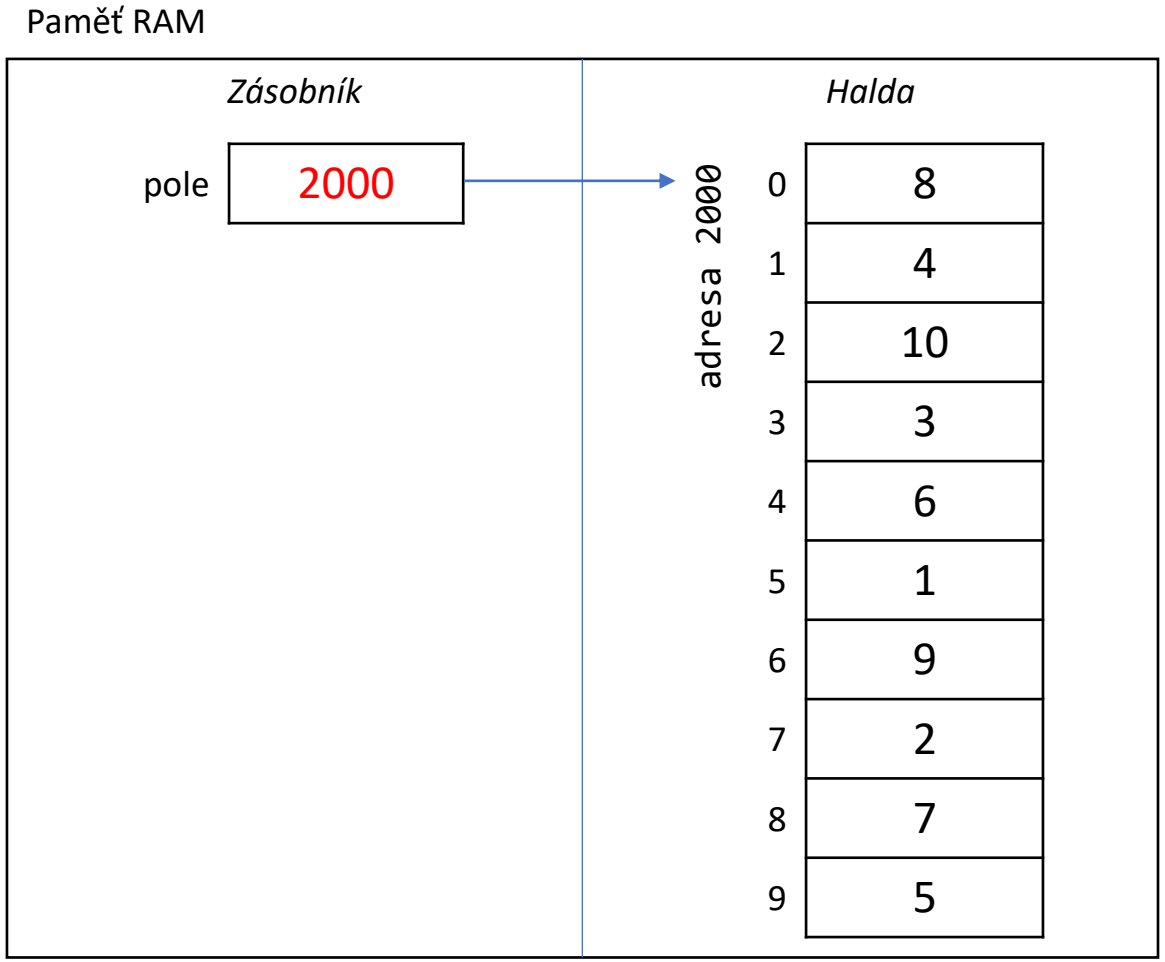
Bubble Sort

```
int[] pole = new int[]
{ 8, 4, 10, 3, 6, 1, 9, 2, 7, 5 };
```



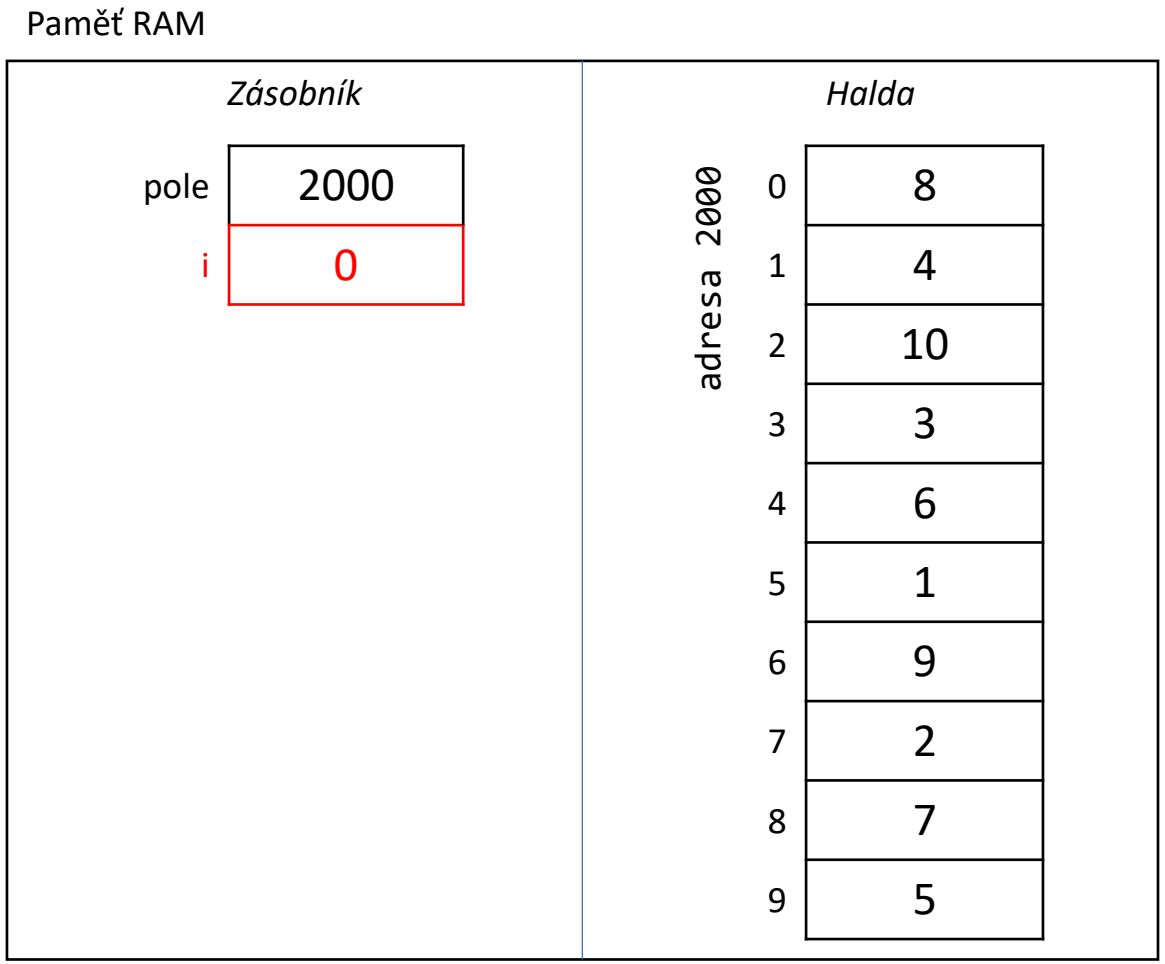
Bubble Sort

```
int[] pole = new int[]
{ 8, 4, 10, 3, 6, 1, 9, 2, 7, 5 };
```



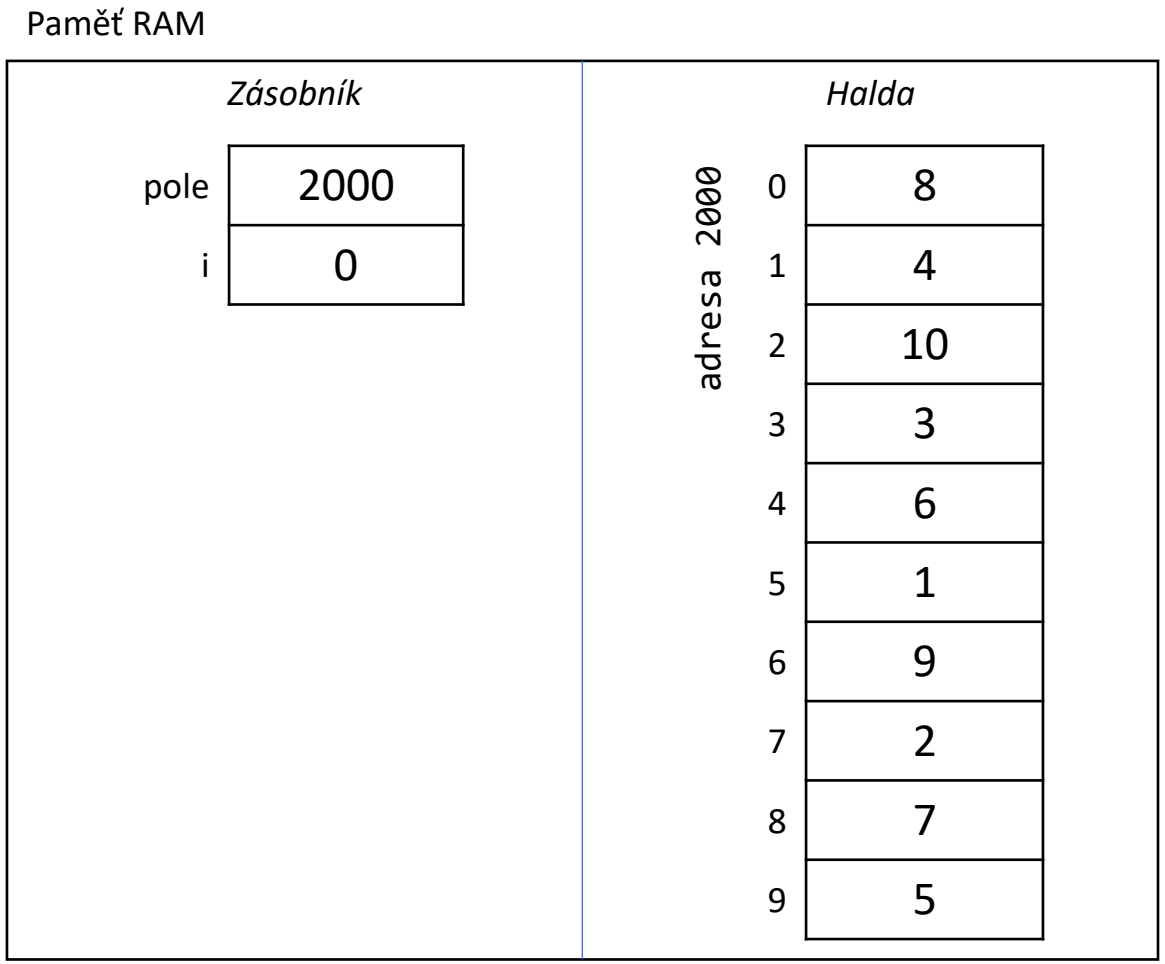
Bubble Sort

```
int[] pole = new int[]
{ 8, 4, 10, 3, 6, 1, 9, 2, 7, 5 };

for (int i = 0; i < pole.Length; i++)
{




}
```



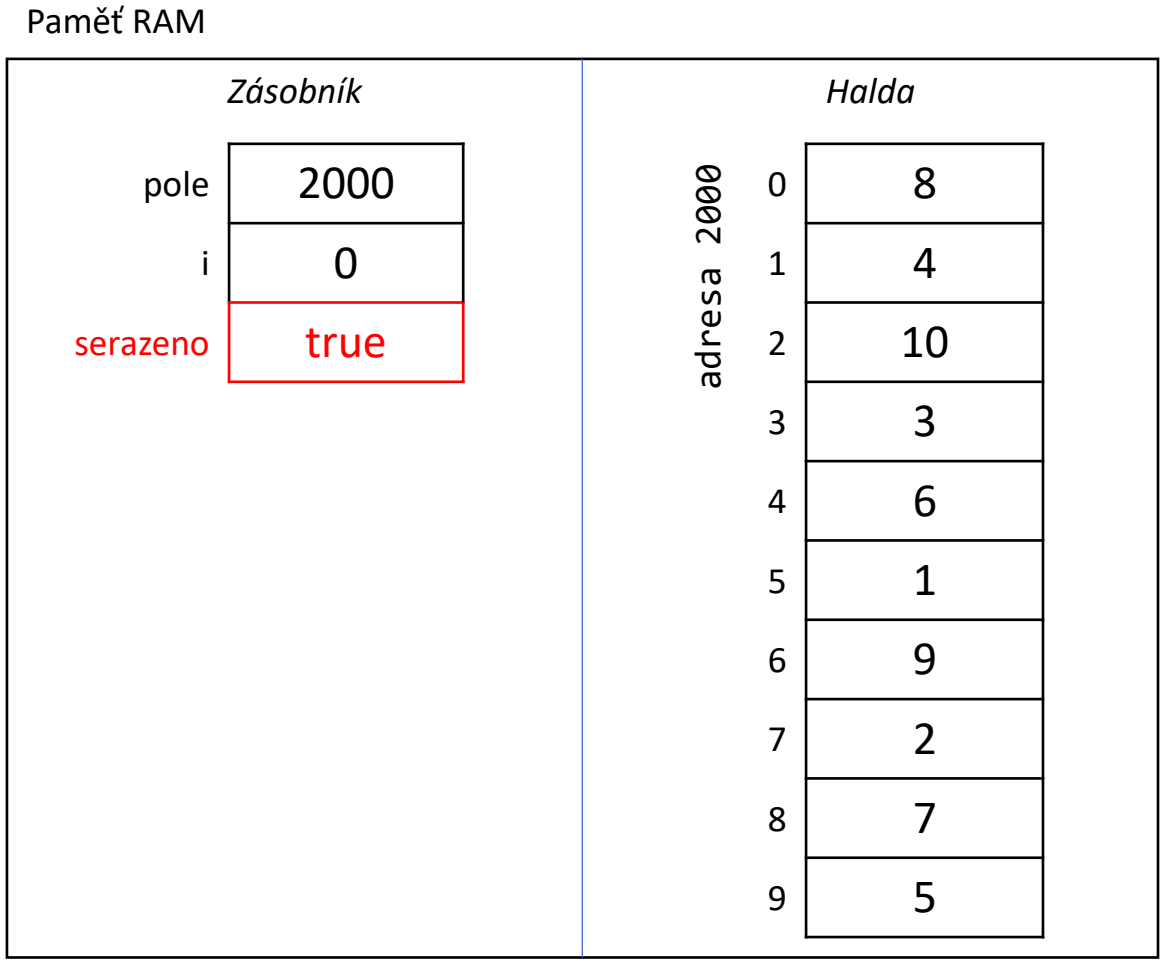
Bubble Sort

```
int[] pole = new int[]
{ 8, 4, 10, 3, 6, 1, 9, 2, 7, 5 };
                  0          9
for (int i = 0; i < pole.Length; i++)
{




}
```



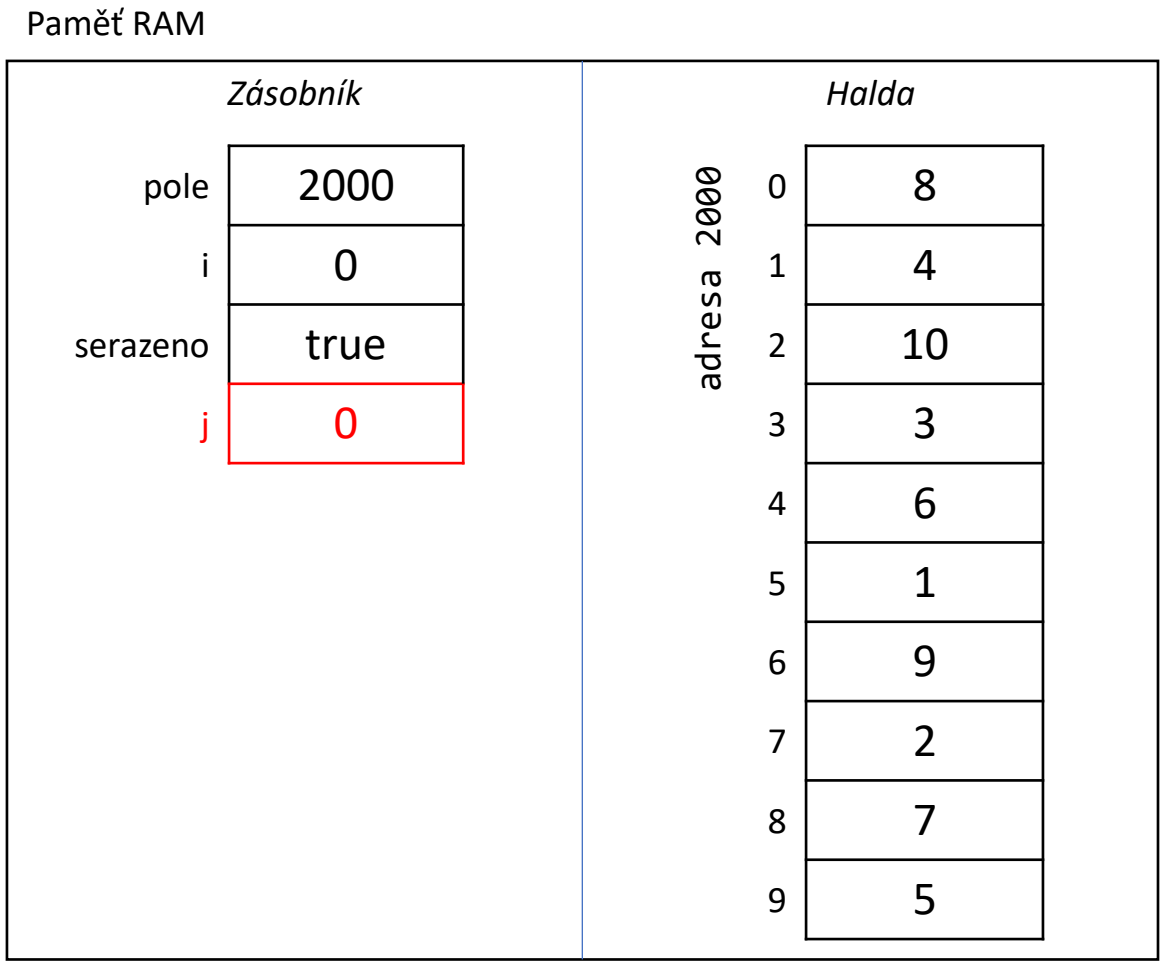
Bubble Sort

```
int[] pole = new int[]
{ 8, 4, 10, 3, 6, 1, 9, 2, 7, 5 };

for (int i = 0; i < pole.Length; i++)
{
    bool serazeno = true;









}
```



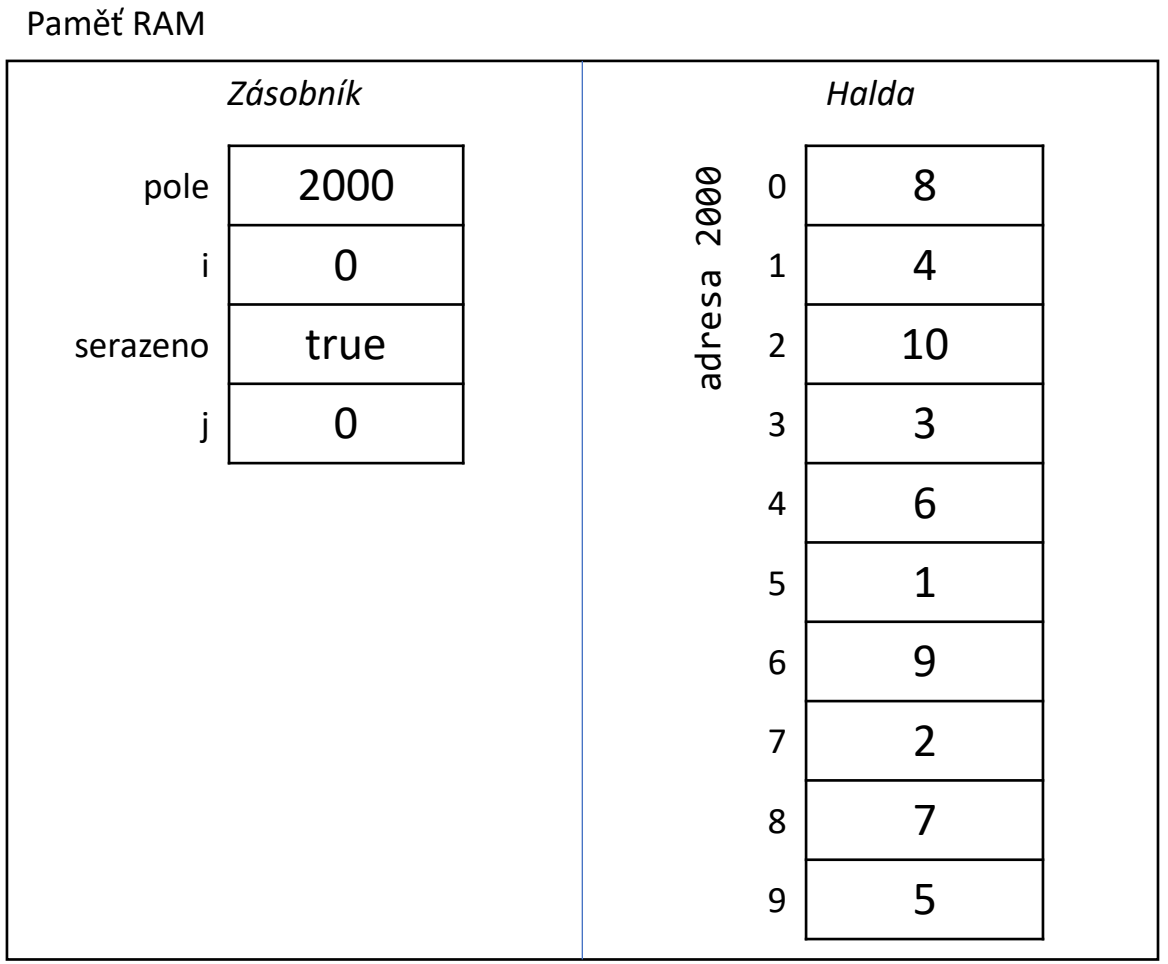
Bubble Sort

```
int[] pole = new int[]
{ 8, 4, 10, 3, 6, 1, 9, 2, 7, 5 };

for (int i = 0; i < pole.Length; i++)
{
    bool serazeno = true;

    for (int j = 0; j < pole.Length - 1 - i; j++)
    {









    }


}
```



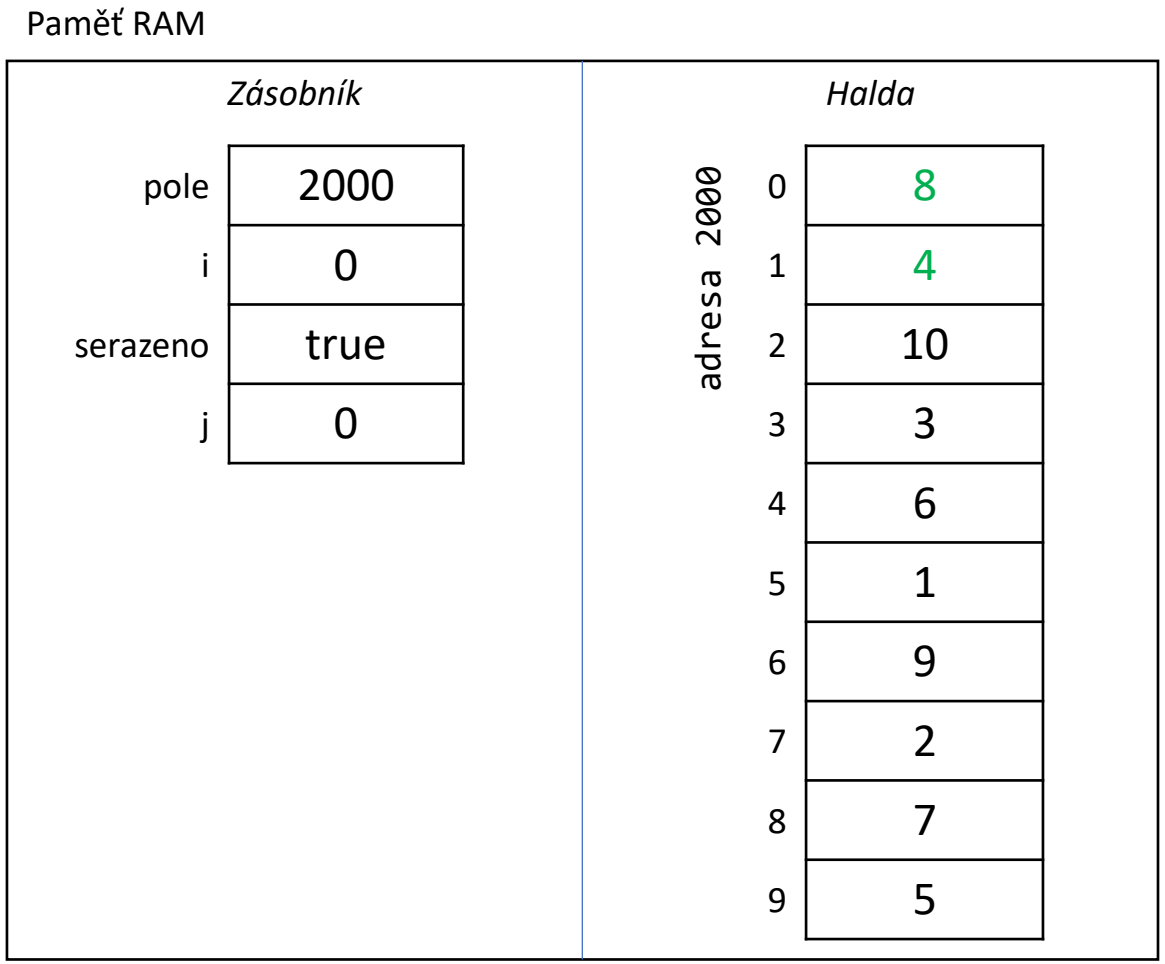
Bubble Sort

```
int[] pole = new int[]
{ 8, 4, 10, 3, 6, 1, 9, 2, 7, 5 };

for (int i = 0; i < pole.Length; i++)
{
    bool serazeno = true;

                    0          9         0
    for (int j = 0; j < pole.Length - 1 - i; j++)
    {









    }


}
```



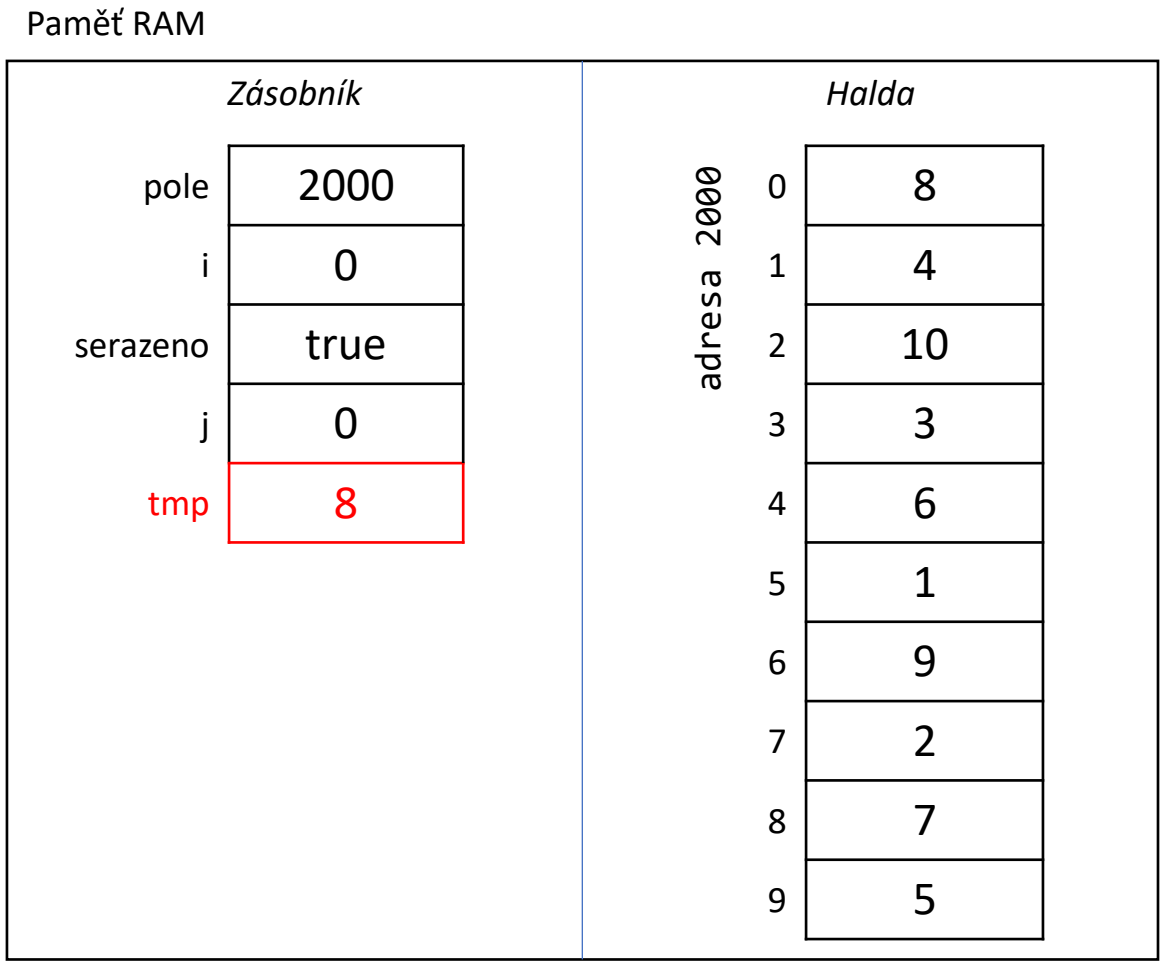
Bubble Sort

```
int[] pole = new int[]
{ 8, 4, 10, 3, 6, 1, 9, 2, 7, 5 };

for (int i = 0; i < pole.Length; i++)
{
    bool serazeno = true;

    for (int j = 0; j < pole.Length - 1 - i; j++)
    {           8        4
        if (pole[j] > pole[j + 1])
        {









        }
    }


}
```



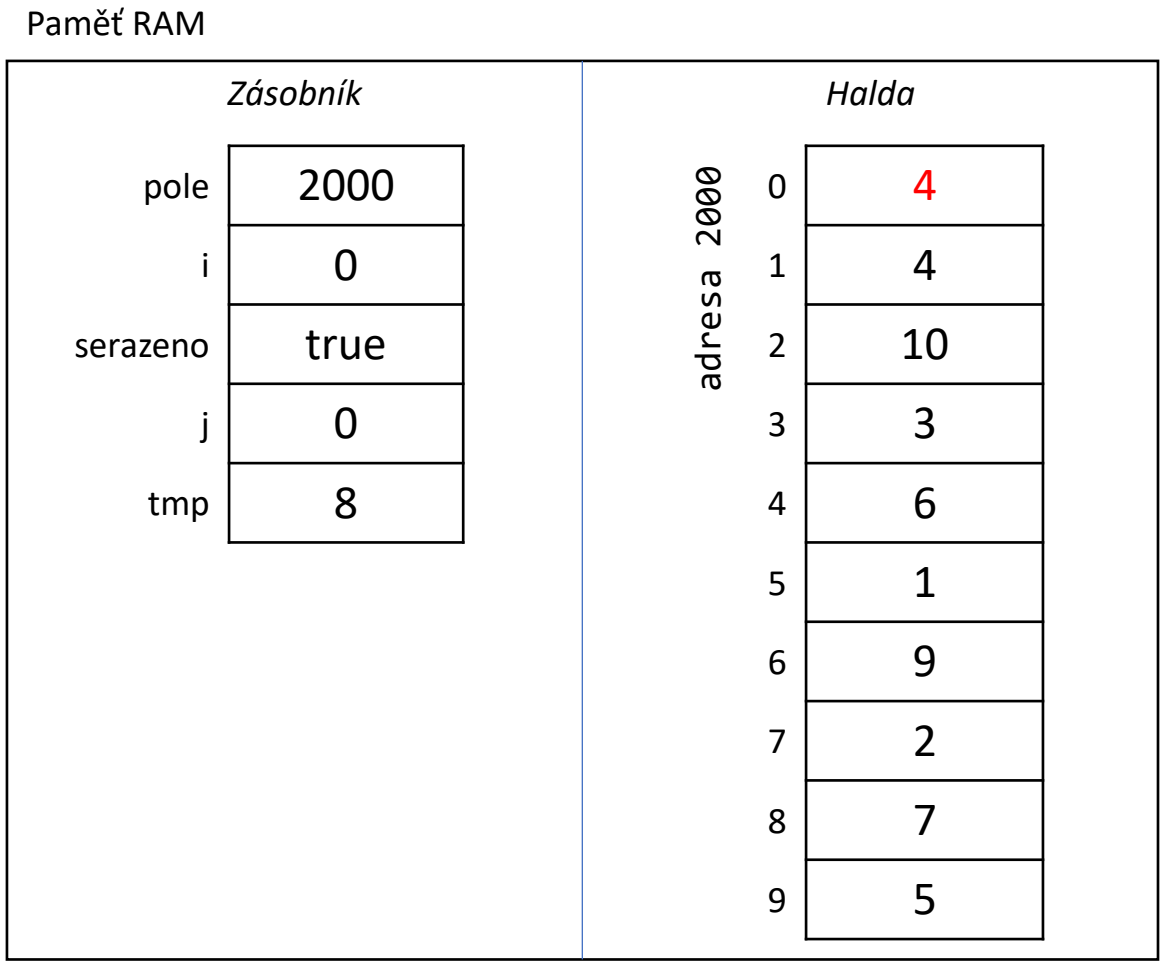
Bubble Sort

```
int[] pole = new int[]
{ 8, 4, 10, 3, 6, 1, 9, 2, 7, 5 };

for (int i = 0; i < pole.Length; i++)
{
    bool serazeno = true;

    for (int j = 0; j < pole.Length - 1 - i; j++)
    {
        if (pole[j] > pole[j + 1])
        {
            int tmp = pole[j];




        }
    }


}
```



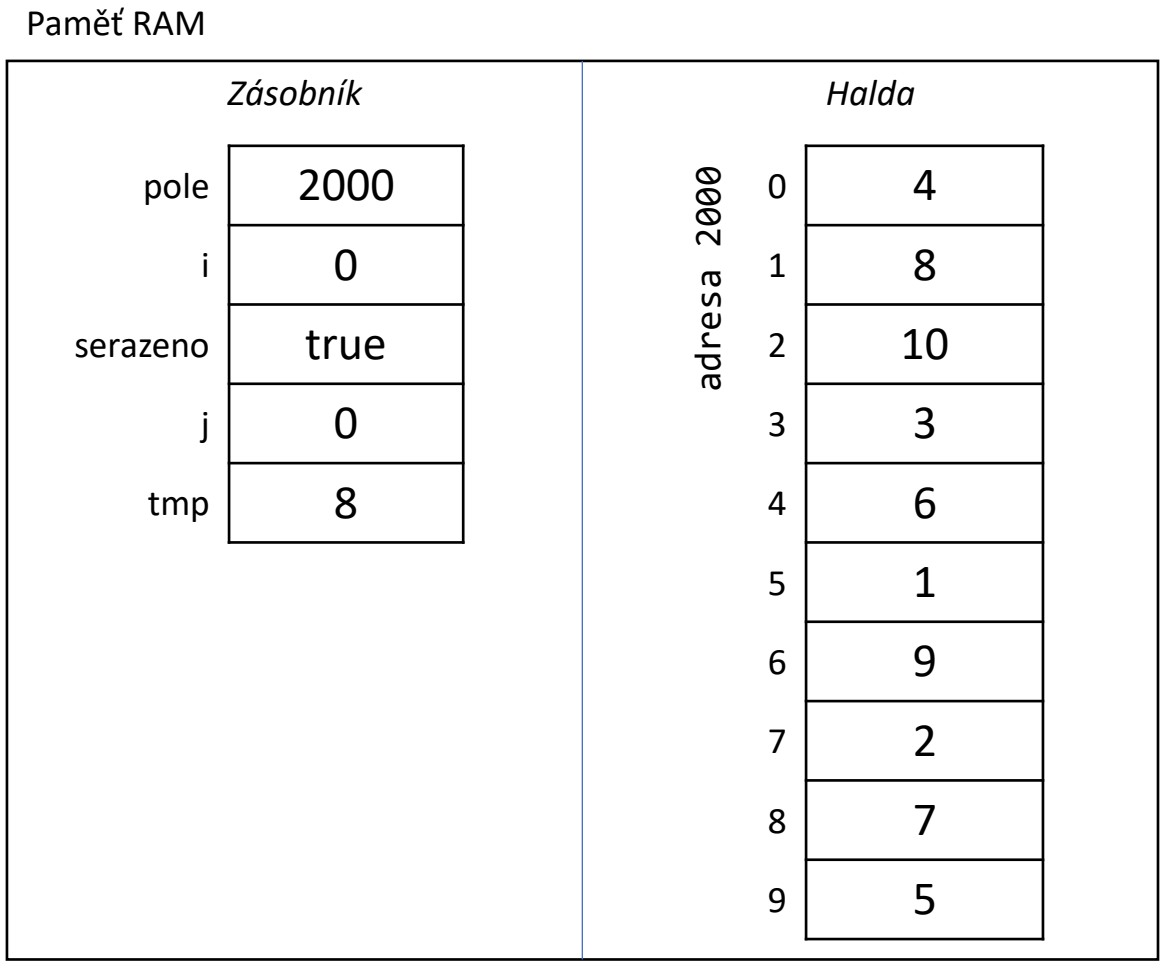
Bubble Sort

```
int[] pole = new int[]
{ 8, 4, 10, 3, 6, 1, 9, 2, 7, 5 };

for (int i = 0; i < pole.Length; i++)
{
    bool serazeno = true;

    for (int j = 0; j < pole.Length - 1 - i; j++)
    {
        if (pole[j] > pole[j + 1])
        {
            int tmp = pole[j];
            pole[j] = pole[j + 1];
            pole[j + 1] = tmp;

            serazeno = false;
        }
    }

    if (serazeno) break;
}
```



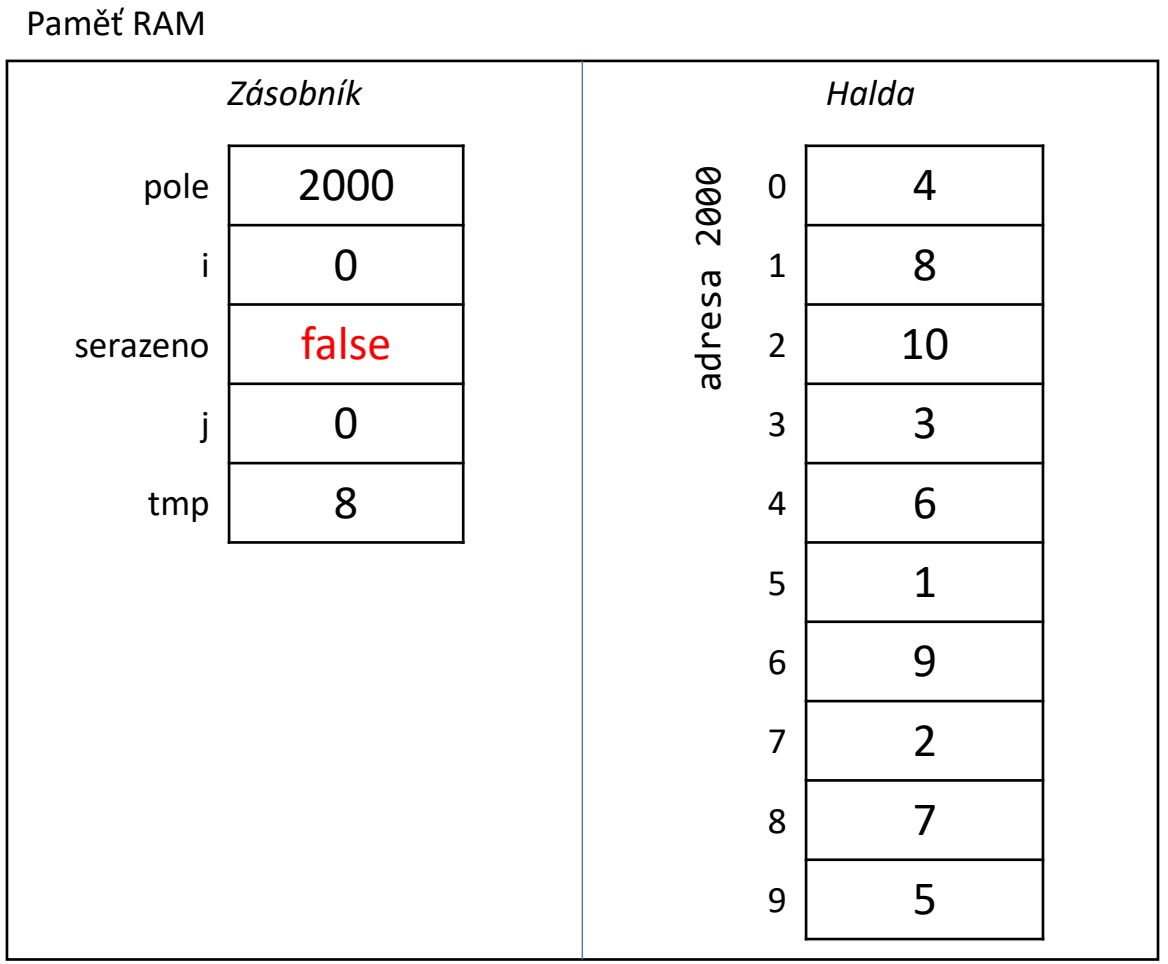
Bubble Sort

```
int[] pole = new int[]
{ 8, 4, 10, 3, 6, 1, 9, 2, 7, 5 };

for (int i = 0; i < pole.Length; i++)
{
    bool serazeno = true;

    for (int j = 0; j < pole.Length - 1 - i; j++)
    {
        if (pole[j] > pole[j + 1])
        {
            int tmp = pole[j];
            pole[j] = pole[j + 1];
            pole[j + 1] = tmp;

            serazeno = false;
        }
    }

    if (serazeno) break;
}
```



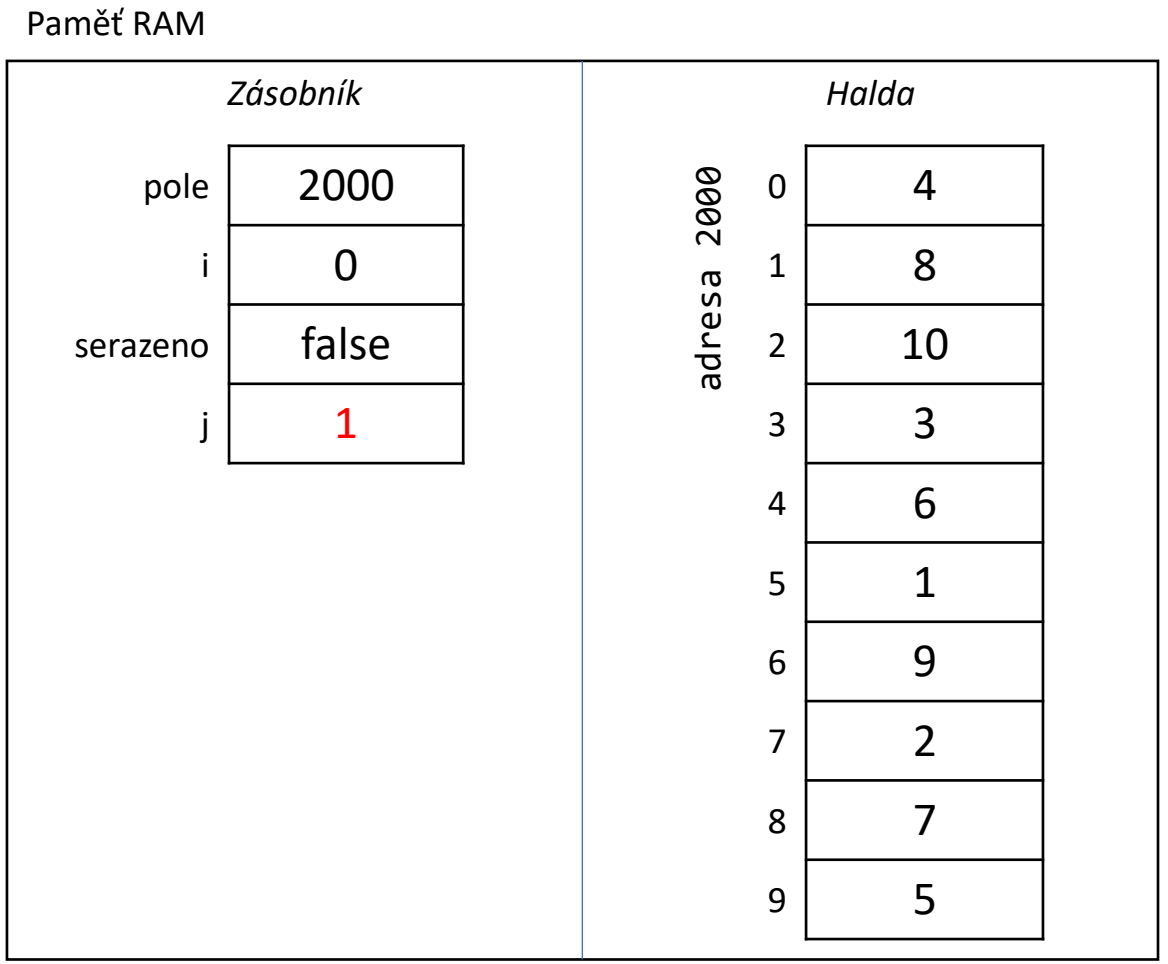
Bubble Sort

```
int[] pole = new int[]
{ 8, 4, 10, 3, 6, 1, 9, 2, 7, 5 };

for (int i = 0; i < pole.Length; i++)
{
    bool serazeno = true;

    for (int j = 0; j < pole.Length - 1 - i; j++)
    {
        if (pole[j] > pole[j + 1])
        {
            int tmp = pole[j];
            pole[j] = pole[j + 1];
            pole[j + 1] = tmp;

            serazeno = false;
        }
    }

    if (serazeno) break;
}
```



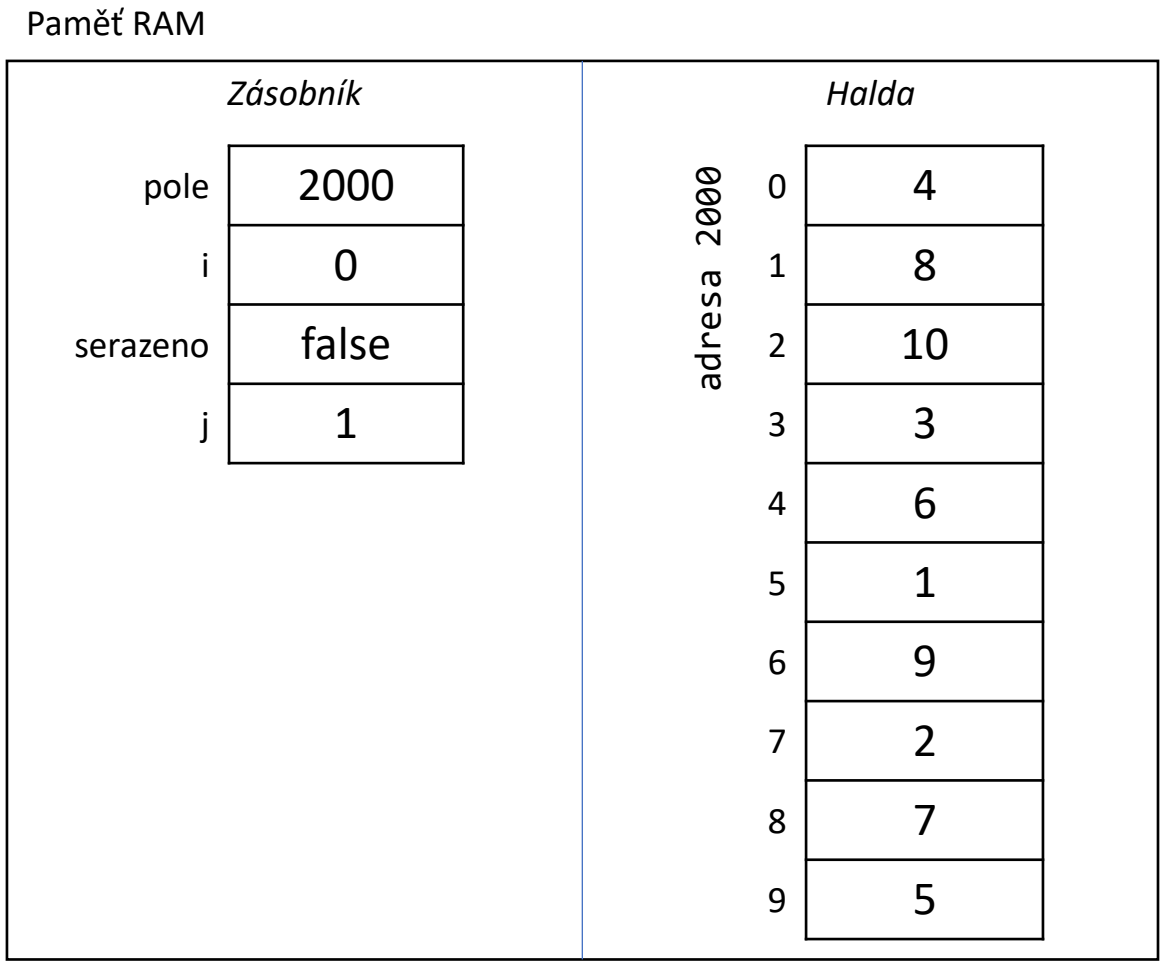
Bubble Sort

```
int[] pole = new int[]
{ 8, 4, 10, 3, 6, 1, 9, 2, 7, 5 };

for (int i = 0; i < pole.Length; i++)
{
    bool serazeno = true;

    for (int j = 0; j < pole.Length - 1 - i; j++)
    {
        if (pole[j] > pole[j + 1])
        {
            int tmp = pole[j];
            pole[j] = pole[j + 1];
            pole[j + 1] = tmp;

            serazeno = false;
        }
    }

    if (serazeno) break;
}
```



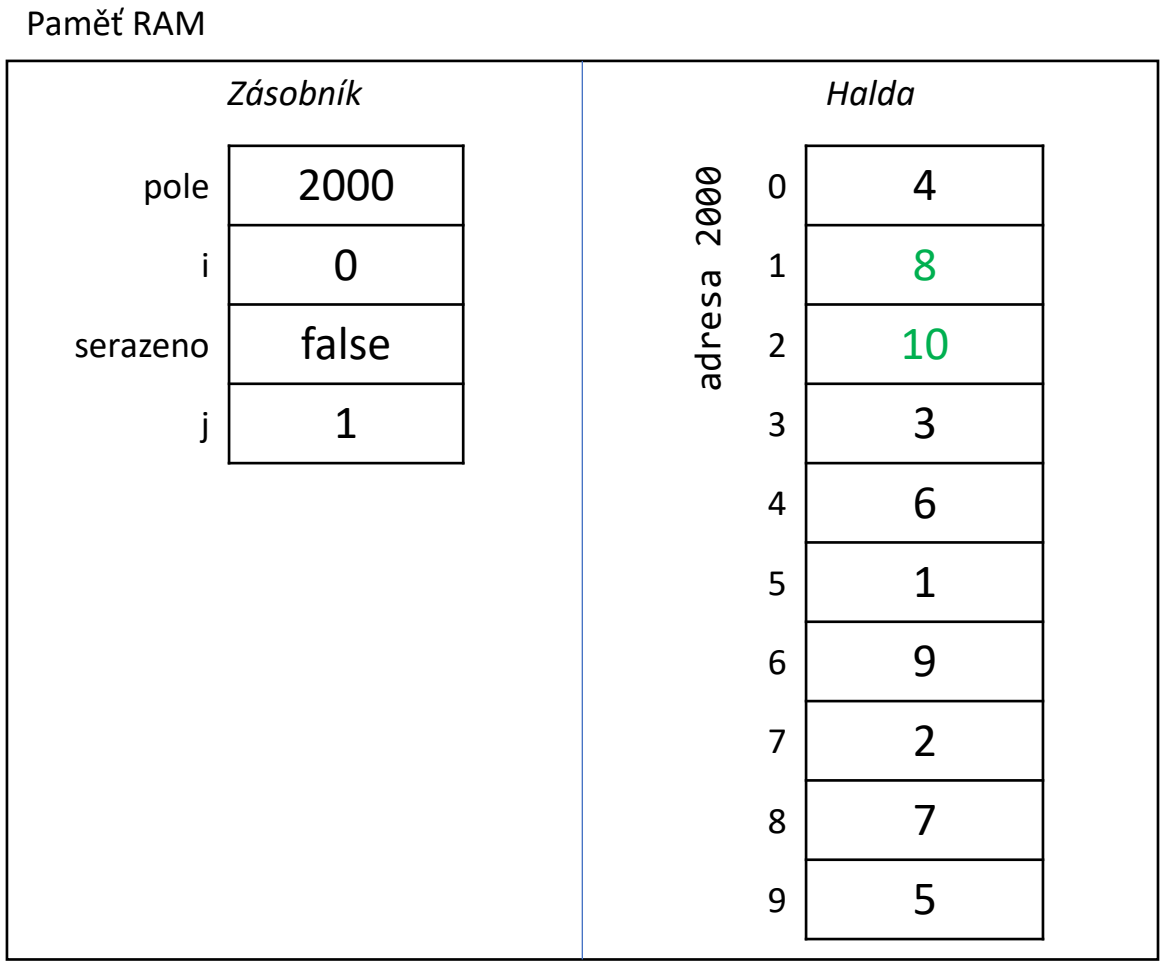
Bubble Sort

```
int[] pole = new int[]
{ 8, 4, 10, 3, 6, 1, 9, 2, 7, 5 };

for (int i = 0; i < pole.Length; i++)
{
    bool serazeno = true;

                      1          9          0
    for (int j = 0; j < pole.Length - 1 - i; j++)
    {
        if (pole[j] > pole[j + 1])
        {
            int tmp = pole[j];
            pole[j] = pole[j + 1];
            pole[j + 1] = tmp;

            serazeno = false;
        }
    }

    if (serazeno) break;
}
```



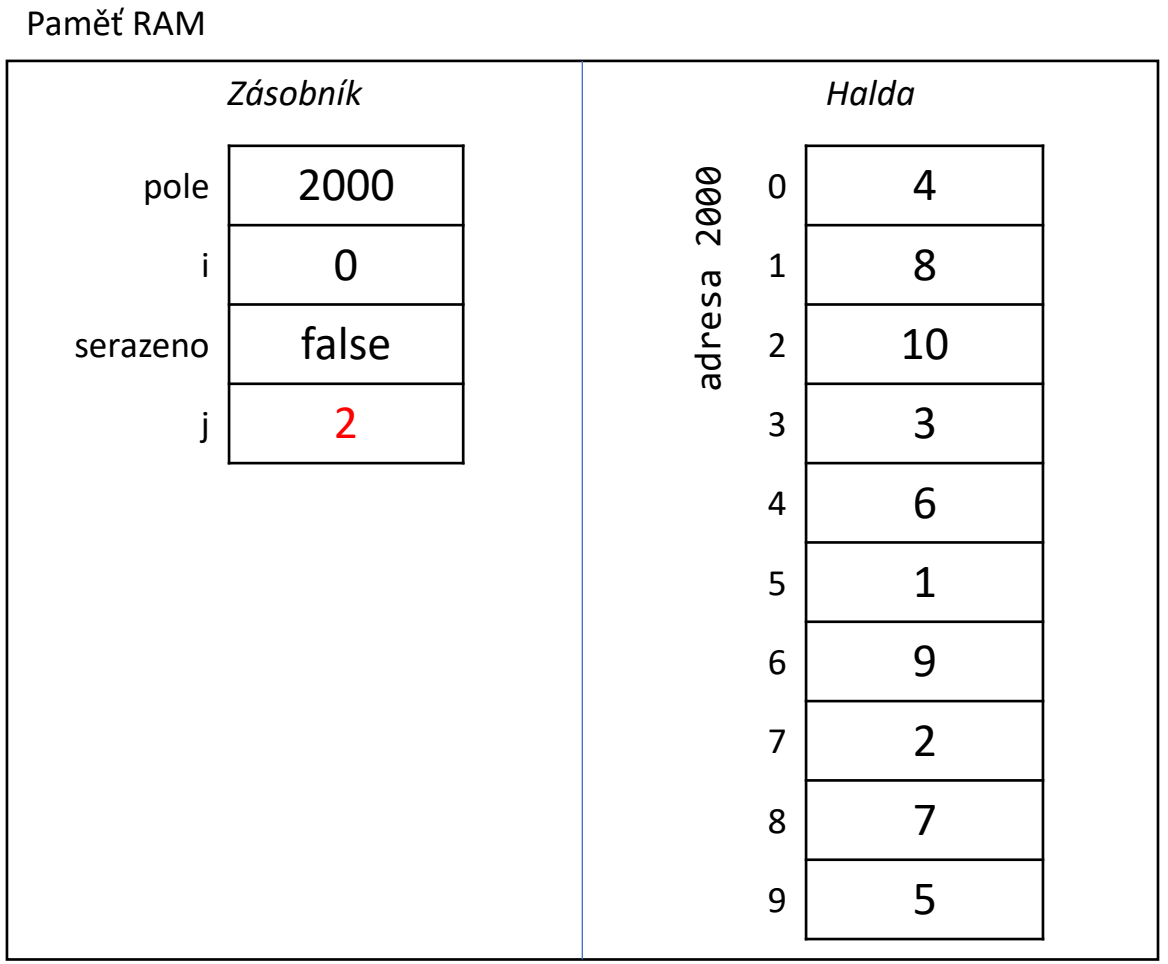
Bubble Sort

```
int[] pole = new int[]
{ 8, 4, 10, 3, 6, 1, 9, 2, 7, 5 };

for (int i = 0; i < pole.Length; i++)
{
    bool serazeno = true;

    for (int j = 0; j < pole.Length - 1 - i; j++)
    {           8        10
        if (pole[j] > pole[j + 1])
        {
            int tmp = pole[j];
            pole[j] = pole[j + 1];
            pole[j + 1] = tmp;

            serazeno = false;
        }
    }

    if (serazeno) break;
}
```



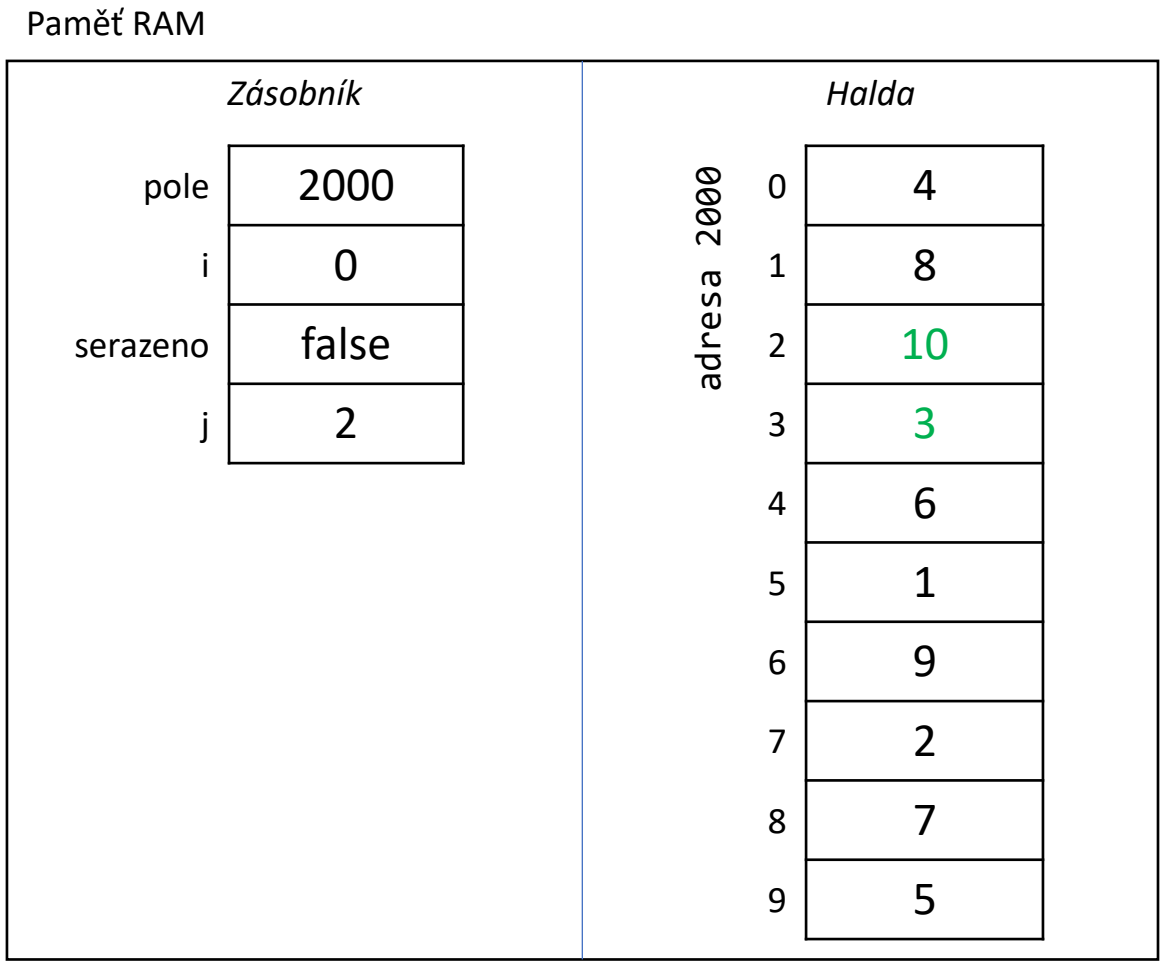
Bubble Sort

```
int[] pole = new int[]
{ 8, 4, 10, 3, 6, 1, 9, 2, 7, 5 };

for (int i = 0; i < pole.Length; i++)
{
    bool serazeno = true;

                    2          9          0
    for (int j = 0; j < pole.Length - 1 - i; j++)
    {
        if (pole[j] > pole[j + 1])
        {
            int tmp = pole[j];
            pole[j] = pole[j + 1];
            pole[j + 1] = tmp;

            serazeno = false;
        }
    }

    if (serazeno) break;
}
```



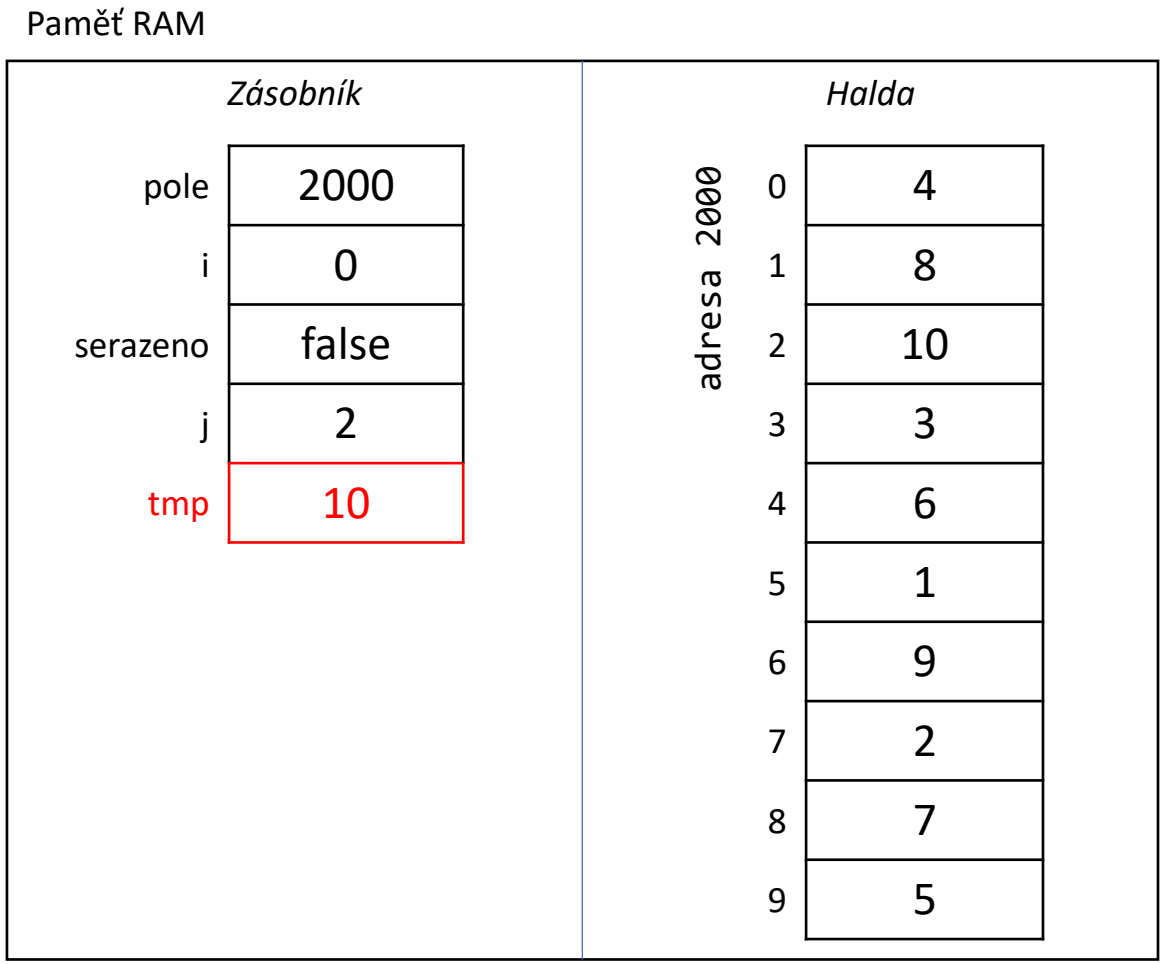
Bubble Sort

```
int[] pole = new int[]
{ 8, 4, 10, 3, 6, 1, 9, 2, 7, 5 };

for (int i = 0; i < pole.Length; i++)
{
    bool serazeno = true;

    for (int j = 0; j < pole.Length - 1 - i; j++)
    {          10       3
        if (pole[j] > pole[j + 1])
        {
            int tmp = pole[j];
            pole[j] = pole[j + 1];
            pole[j + 1] = tmp;

            serazeno = false;
        }
    }

    if (serazeno) break;
}
```



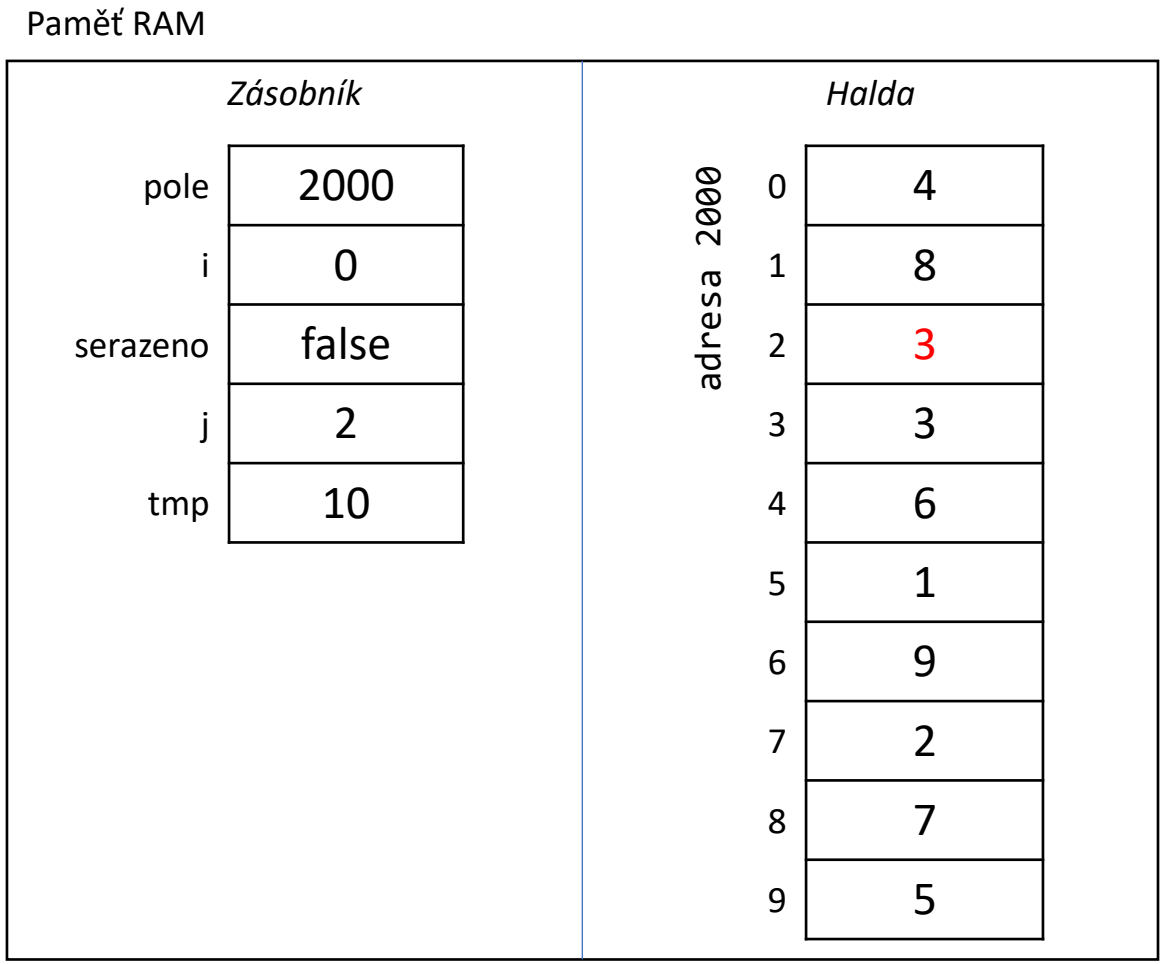
Bubble Sort

```
int[] pole = new int[]
{ 8, 4, 10, 3, 6, 1, 9, 2, 7, 5 };

for (int i = 0; i < pole.Length; i++)
{
    bool serazeno = true;

    for (int j = 0; j < pole.Length - 1 - i; j++)
    {
        if (pole[j] > pole[j + 1])
        {
            int tmp = pole[j];
            pole[j] = pole[j + 1];
            pole[j + 1] = tmp;

            serazeno = false;
        }
    }

    if (serazeno) break;
}
```



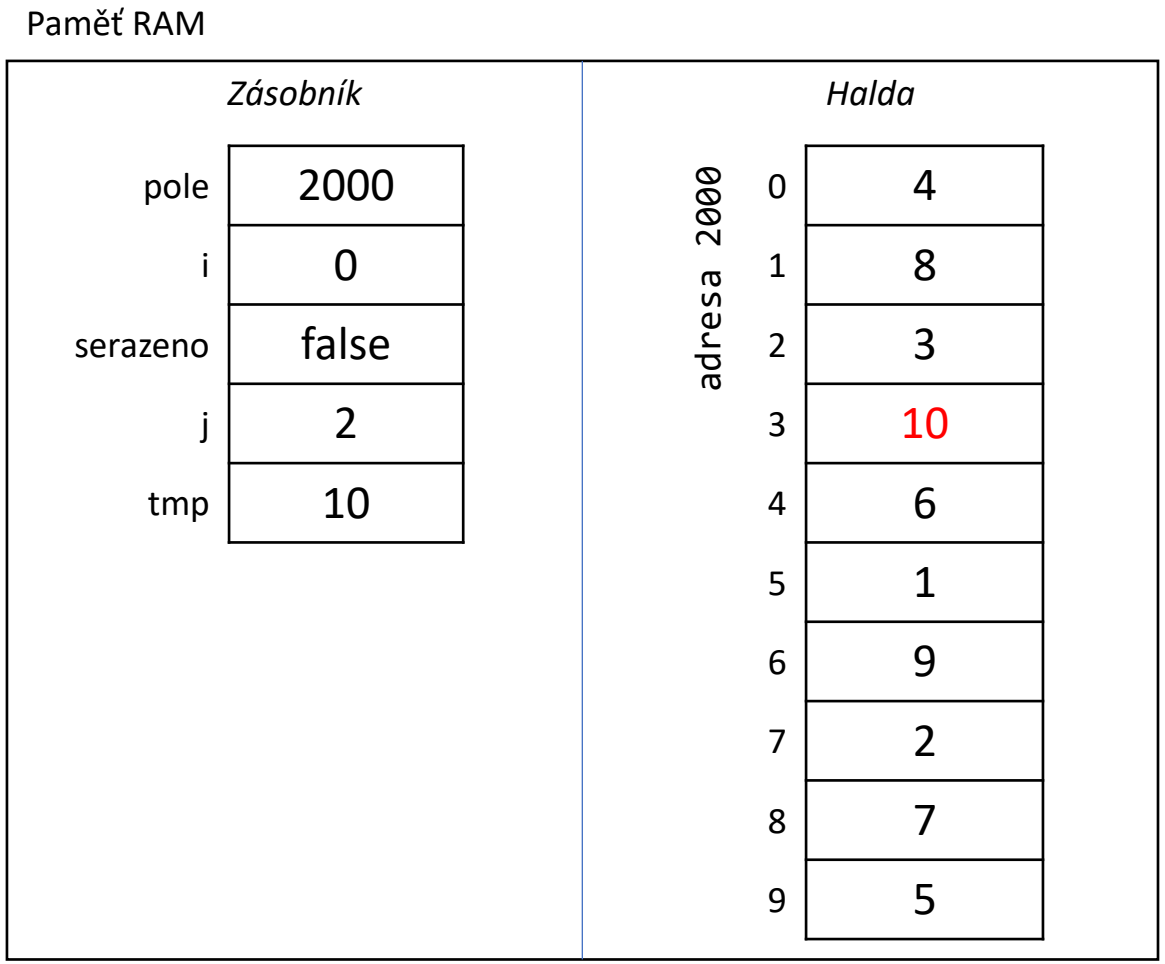
Bubble Sort

```
int[] pole = new int[]
{ 8, 4, 10, 3, 6, 1, 9, 2, 7, 5 };

for (int i = 0; i < pole.Length; i++)
{
    bool serazeno = true;

    for (int j = 0; j < pole.Length - 1 - i; j++)
    {
        if (pole[j] > pole[j + 1])
        {
            int tmp = pole[j];
            pole[j] = pole[j + 1];
            pole[j + 1] = tmp;

            serazeno = false;
        }
    }

    if (serazeno) break;
}
```



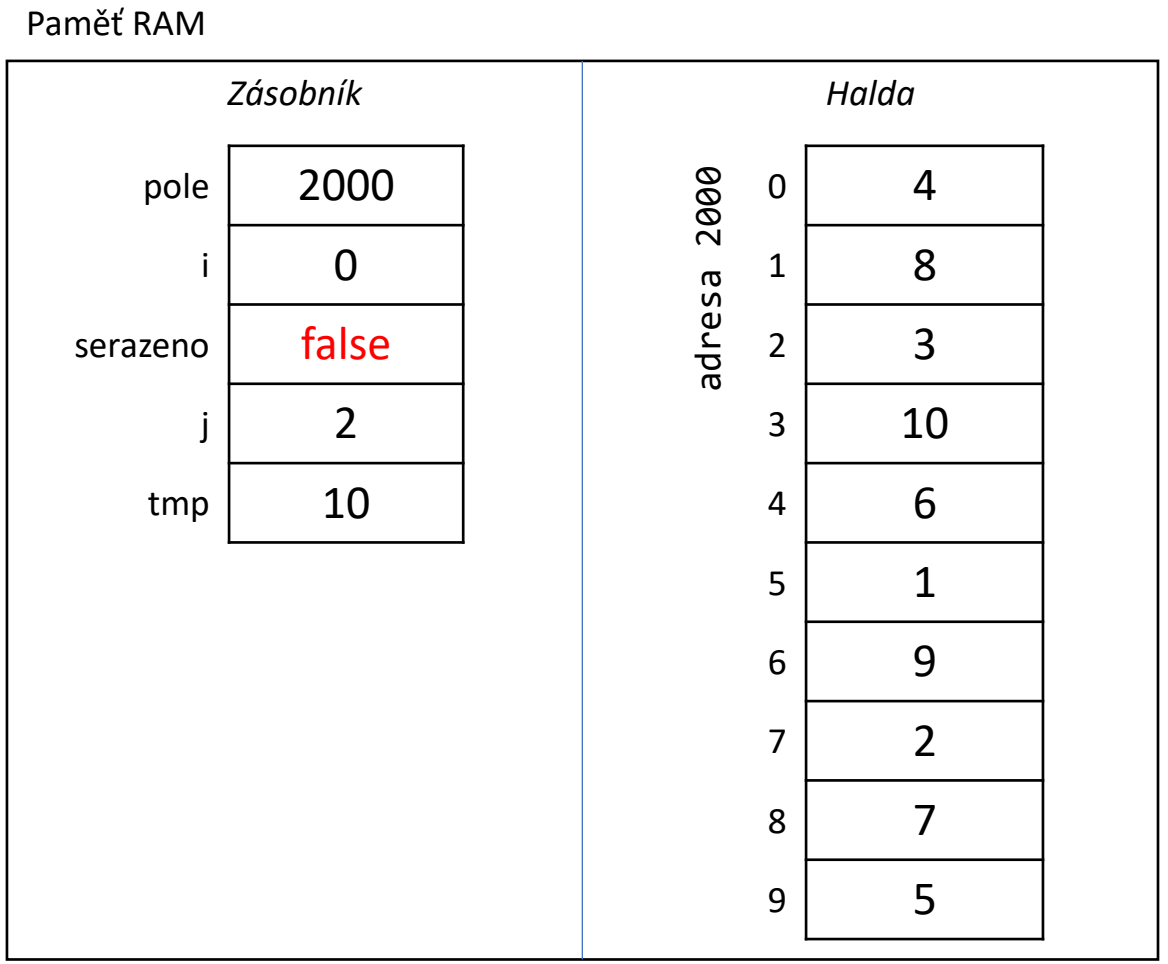
Bubble Sort

```
int[] pole = new int[]
{ 8, 4, 10, 3, 6, 1, 9, 2, 7, 5 };

for (int i = 0; i < pole.Length; i++)
{
    bool serazeno = true;

    for (int j = 0; j < pole.Length - 1 - i; j++)
    {
        if (pole[j] > pole[j + 1])
        {
            int tmp = pole[j];
            pole[j] = pole[j + 1];
            pole[j + 1] = tmp;

            serazeno = false;
        }
    }

    if (serazeno) break;
}
```



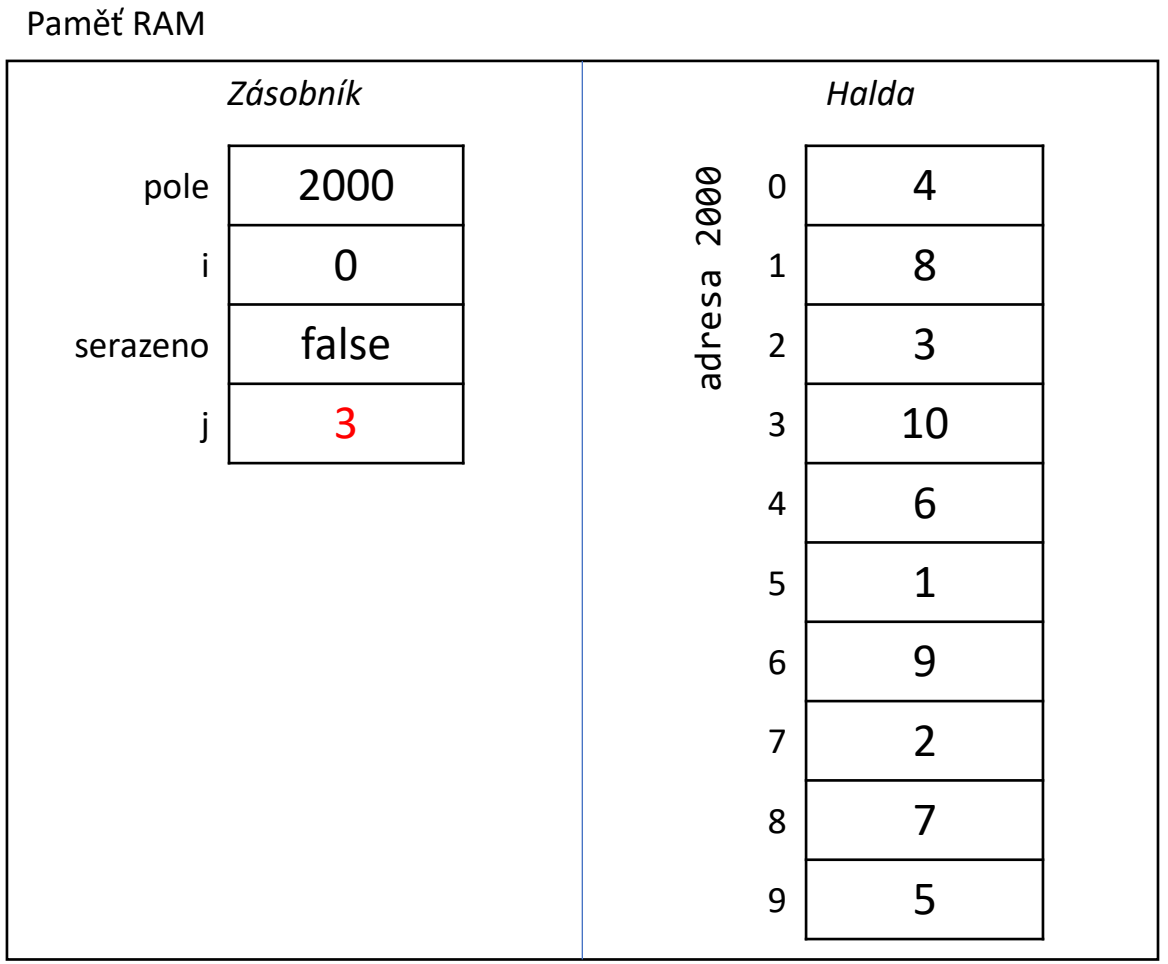
Bubble Sort

```
int[] pole = new int[]
{ 8, 4, 10, 3, 6, 1, 9, 2, 7, 5 };

for (int i = 0; i < pole.Length; i++)
{
    bool serazeno = true;

    for (int j = 0; j < pole.Length - 1 - i; j++)
    {
        if (pole[j] > pole[j + 1])
        {
            int tmp = pole[j];
            pole[j] = pole[j + 1];
            pole[j + 1] = tmp;

            serazeno = false;
        }
    }

    if (serazeno) break;
}
```



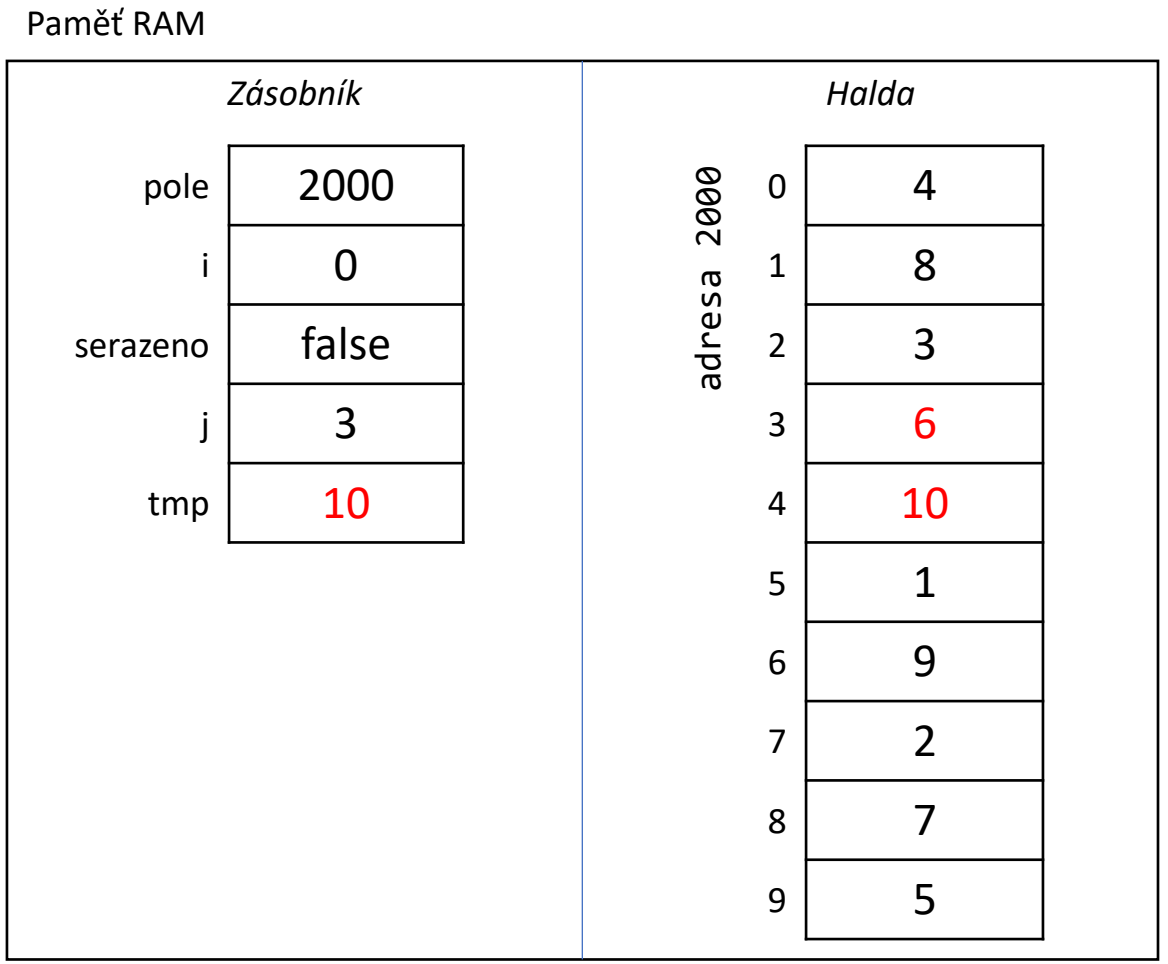
Bubble Sort

```
int[] pole = new int[]
{ 8, 4, 10, 3, 6, 1, 9, 2, 7, 5 };

for (int i = 0; i < pole.Length; i++)
{
    bool serazeno = true;

    for (int j = 0; j < pole.Length - 1 - i; j++)
    {          10       6
        if (pole[j] > pole[j + 1])
        {
            int tmp = pole[j];
            pole[j] = pole[j + 1];
            pole[j + 1] = tmp;

            serazeno = false;
        }
    }

    if (serazeno) break;
}
```



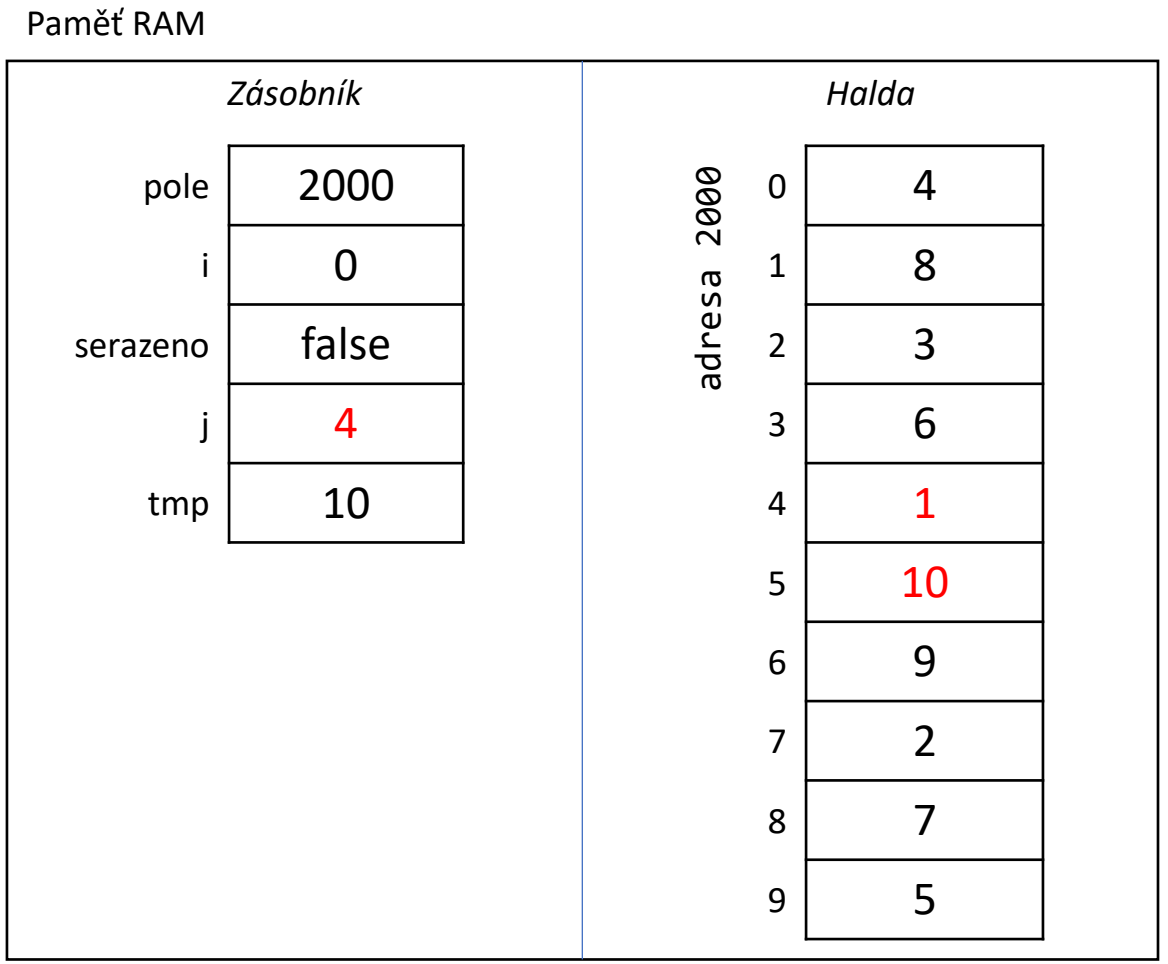
Bubble Sort

```
int[] pole = new int[]
{ 8, 4, 10, 3, 6, 1, 9, 2, 7, 5 };

for (int i = 0; i < pole.Length; i++)
{
    bool serazeno = true;

    for (int j = 0; j < pole.Length - 1 - i; j++)
    {
        if (pole[j] > pole[j + 1])
        {
            int tmp = pole[j];
            pole[j] = pole[j + 1];
            pole[j + 1] = tmp;

            serazeno = false;
        }
    }

    if (serazeno) break;
}
```



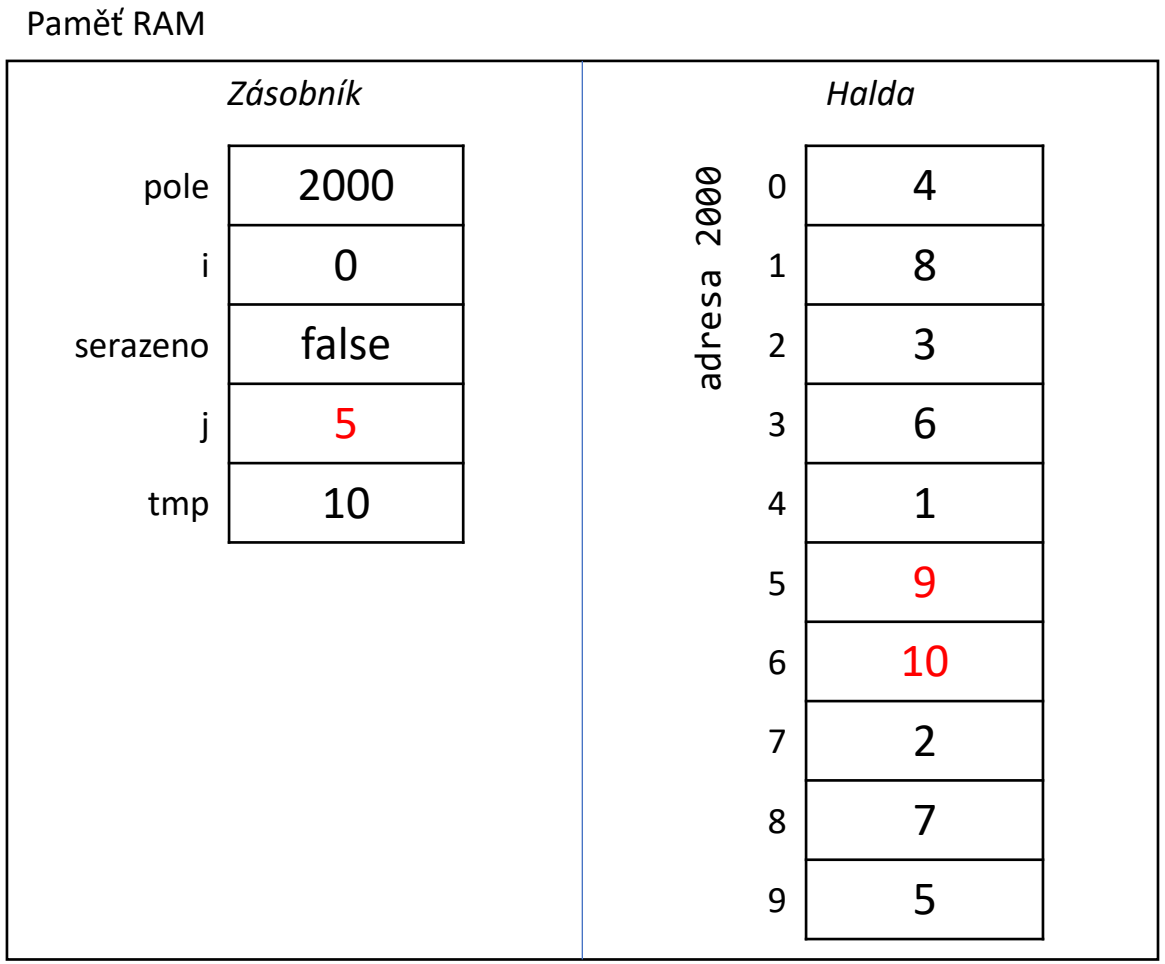
Bubble Sort

```
int[] pole = new int[]
{ 8, 4, 10, 3, 6, 1, 9, 2, 7, 5 };

for (int i = 0; i < pole.Length; i++)
{
    bool serazeno = true;

    for (int j = 0; j < pole.Length - 1 - i; j++)
    {          10       1
        if (pole[j] > pole[j + 1])
        {
            int tmp = pole[j];
            pole[j] = pole[j + 1];
            pole[j + 1] = tmp;

            serazeno = false;
        }
    }

    if (serazeno) break;
}
```



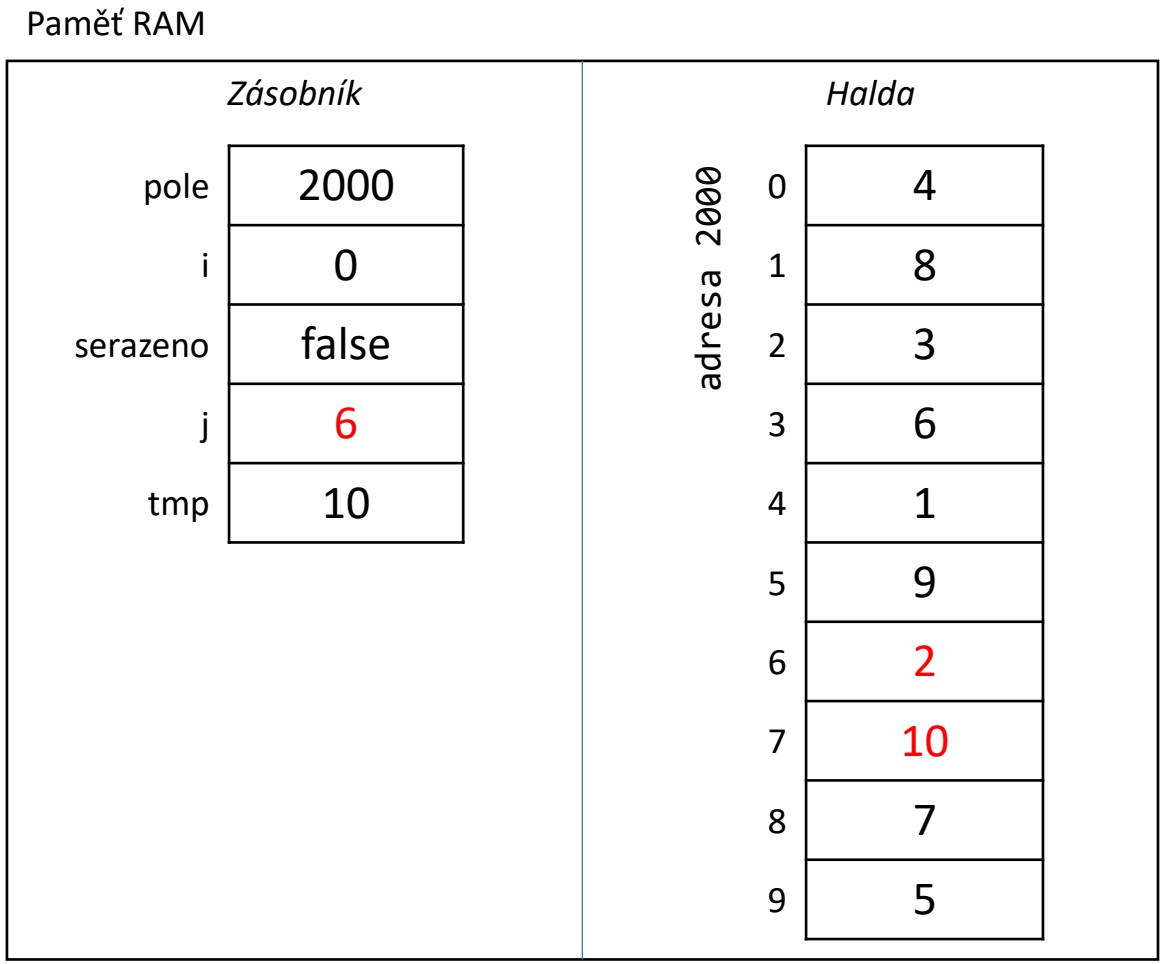
Bubble Sort

```
int[] pole = new int[]
{ 8, 4, 10, 3, 6, 1, 9, 2, 7, 5 };

for (int i = 0; i < pole.Length; i++)
{
    bool serazeno = true;

    for (int j = 0; j < pole.Length - 1 - i; j++)
    {          10       9
        if (pole[j] > pole[j + 1])
        {
            int tmp = pole[j];
            pole[j] = pole[j + 1];
            pole[j + 1] = tmp;

            serazeno = false;
        }
    }

    if (serazeno) break;
}
```



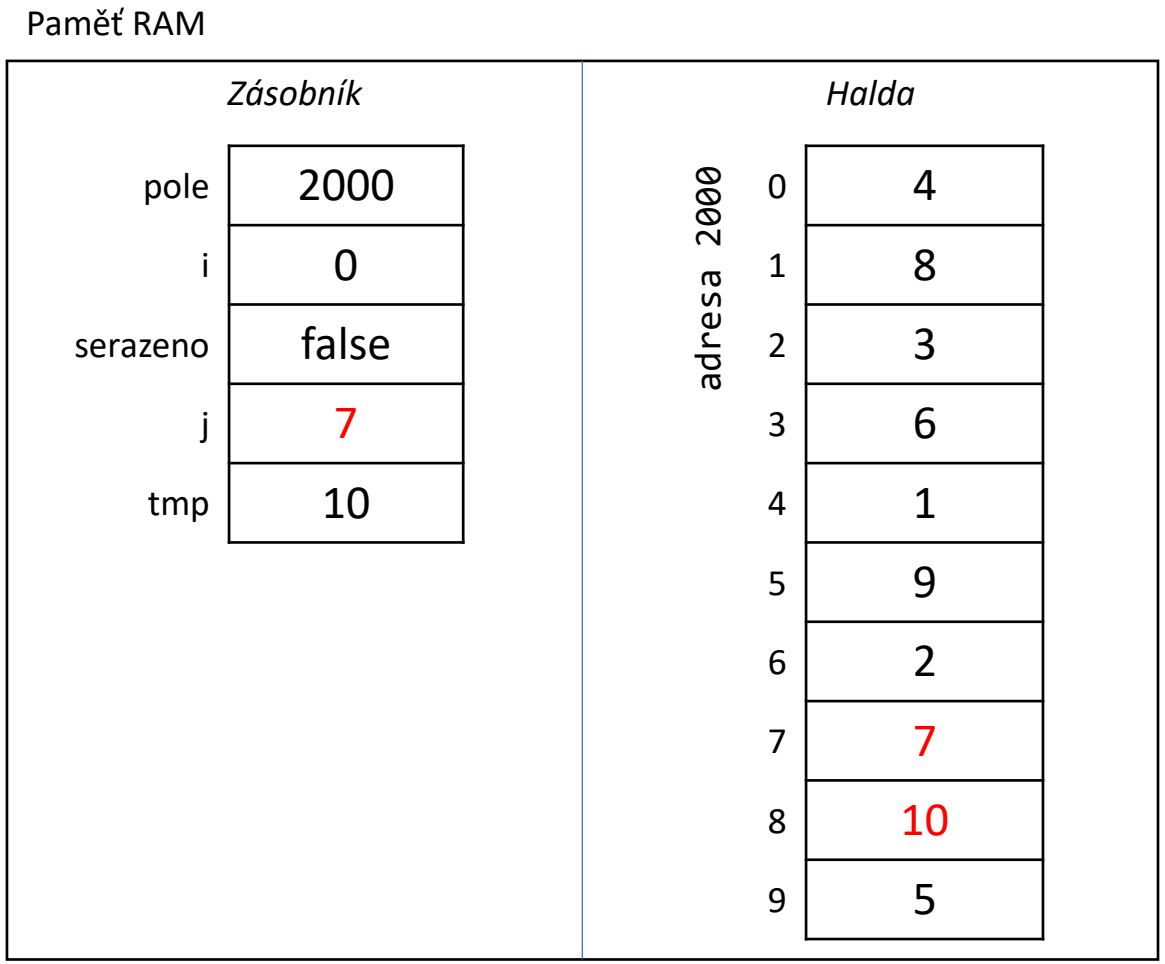
Bubble Sort

```
int[] pole = new int[]
{ 8, 4, 10, 3, 6, 1, 9, 2, 7, 5 };

for (int i = 0; i < pole.Length; i++)
{
    bool serazeno = true;

    for (int j = 0; j < pole.Length - 1 - i; j++)
    {          10       2
        if (pole[j] > pole[j + 1])
        {
            int tmp = pole[j];
            pole[j] = pole[j + 1];
            pole[j + 1] = tmp;

            serazeno = false;
        }
    }

    if (serazeno) break;
}
```



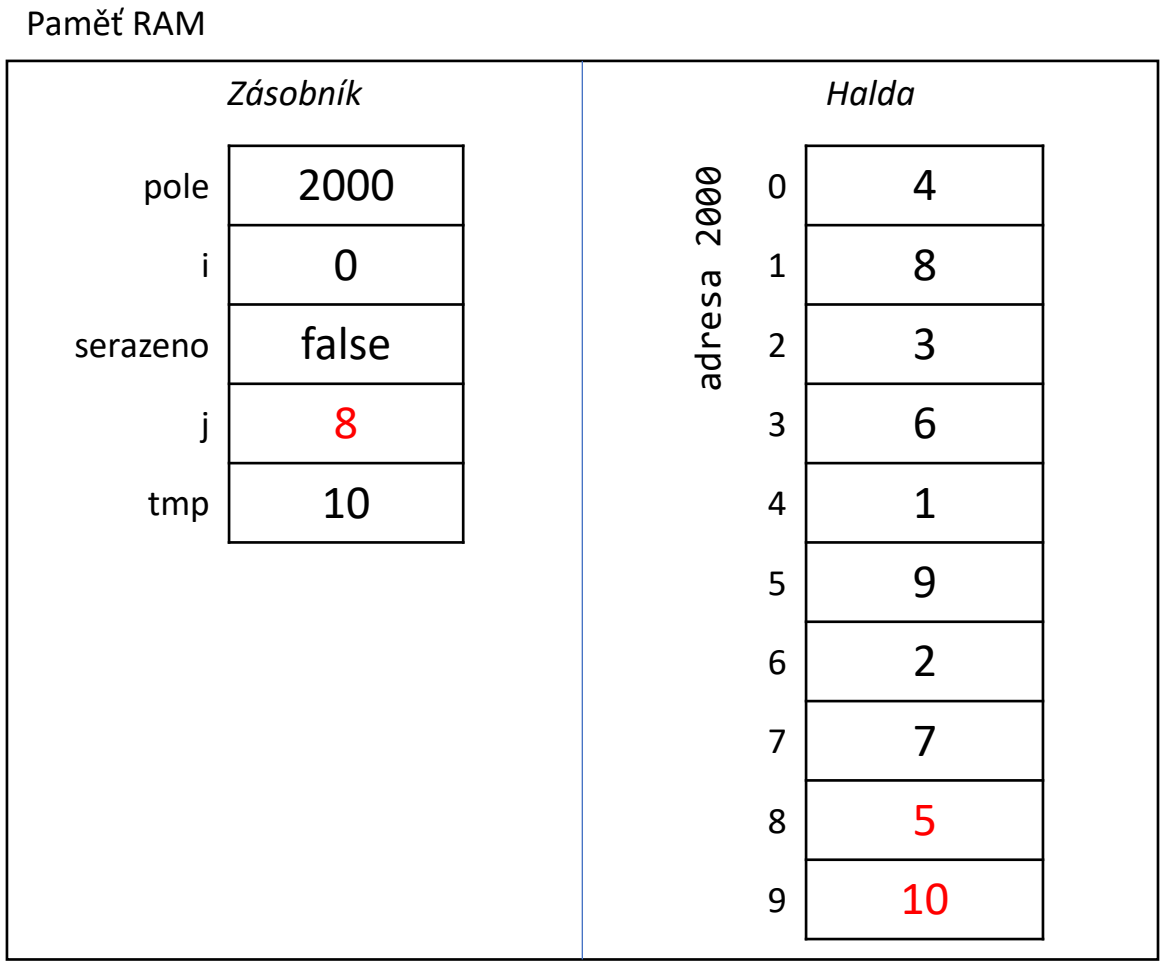
Bubble Sort

```
int[] pole = new int[]
{ 8, 4, 10, 3, 6, 1, 9, 2, 7, 5 };

for (int i = 0; i < pole.Length; i++)
{
    bool serazeno = true;

    for (int j = 0; j < pole.Length - 1 - i; j++)
    {          10       7
        if (pole[j] > pole[j + 1])
        {
            int tmp = pole[j];
            pole[j] = pole[j + 1];
            pole[j + 1] = tmp;

            serazeno = false;
        }
    }

    if (serazeno) break;
}
```



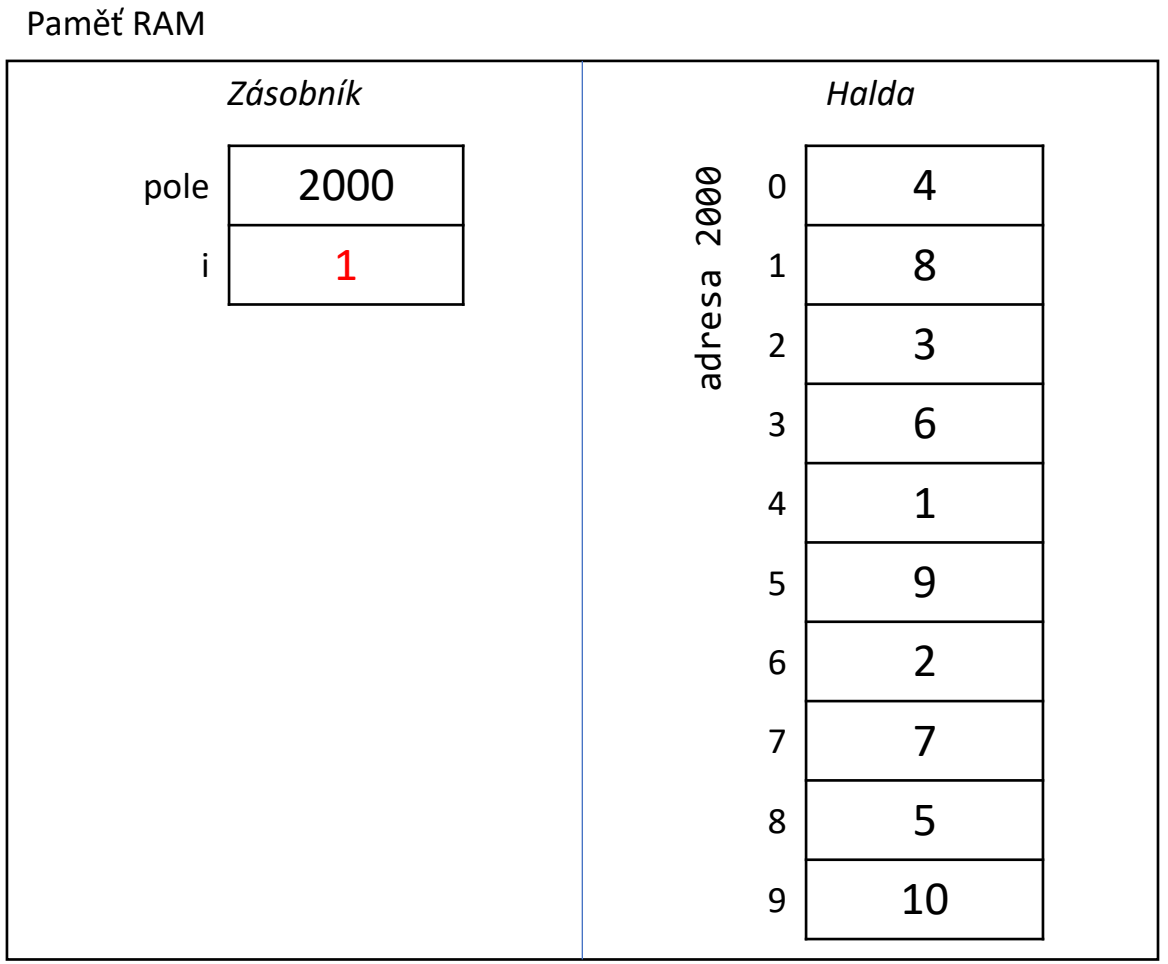
Bubble Sort

```
int[] pole = new int[]
{ 8, 4, 10, 3, 6, 1, 9, 2, 7, 5 };

for (int i = 0; i < pole.Length; i++)
{
    bool serazeno = true;

    for (int j = 0; j < pole.Length - 1 - i; j++)
    {          10       5
        if (pole[j] > pole[j + 1])
        {
            int tmp = pole[j];
            pole[j] = pole[j + 1];
            pole[j + 1] = tmp;

            serazeno = false;
        }
    }

    if (serazeno) break;
}
```



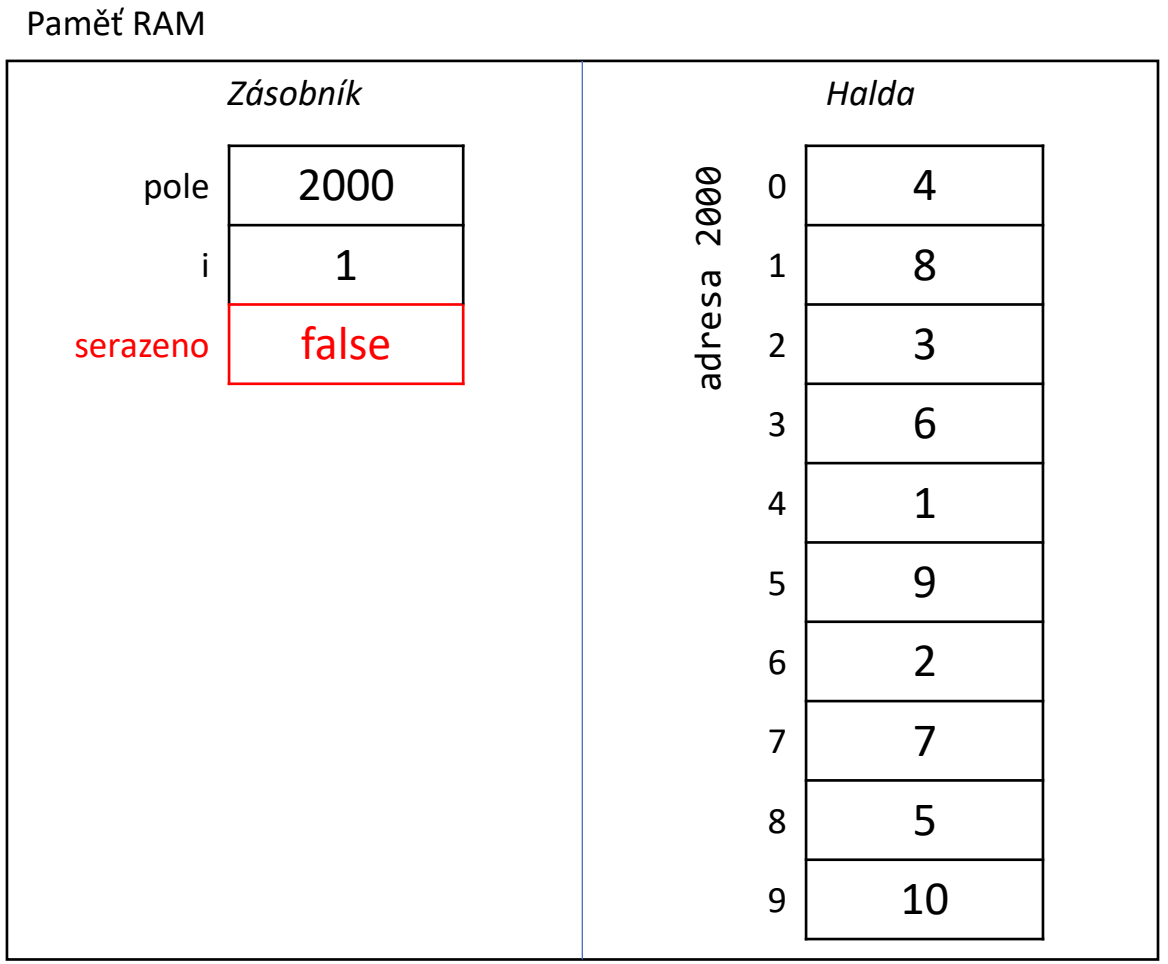
Bubble Sort

```
int[] pole = new int[]
{ 8, 4, 10, 3, 6, 1, 9, 2, 7, 5 };

for (int i = 0; i < pole.Length; i++)
{
    bool serazeno = true;

    for (int j = 0; j < pole.Length - 1 - i; j++)
    {
        if (pole[j] > pole[j + 1])
        {
            int tmp = pole[j];
            pole[j] = pole[j + 1];
            pole[j + 1] = tmp;

            serazeno = false;
        }
    }

    if (serazeno) break;
}
```



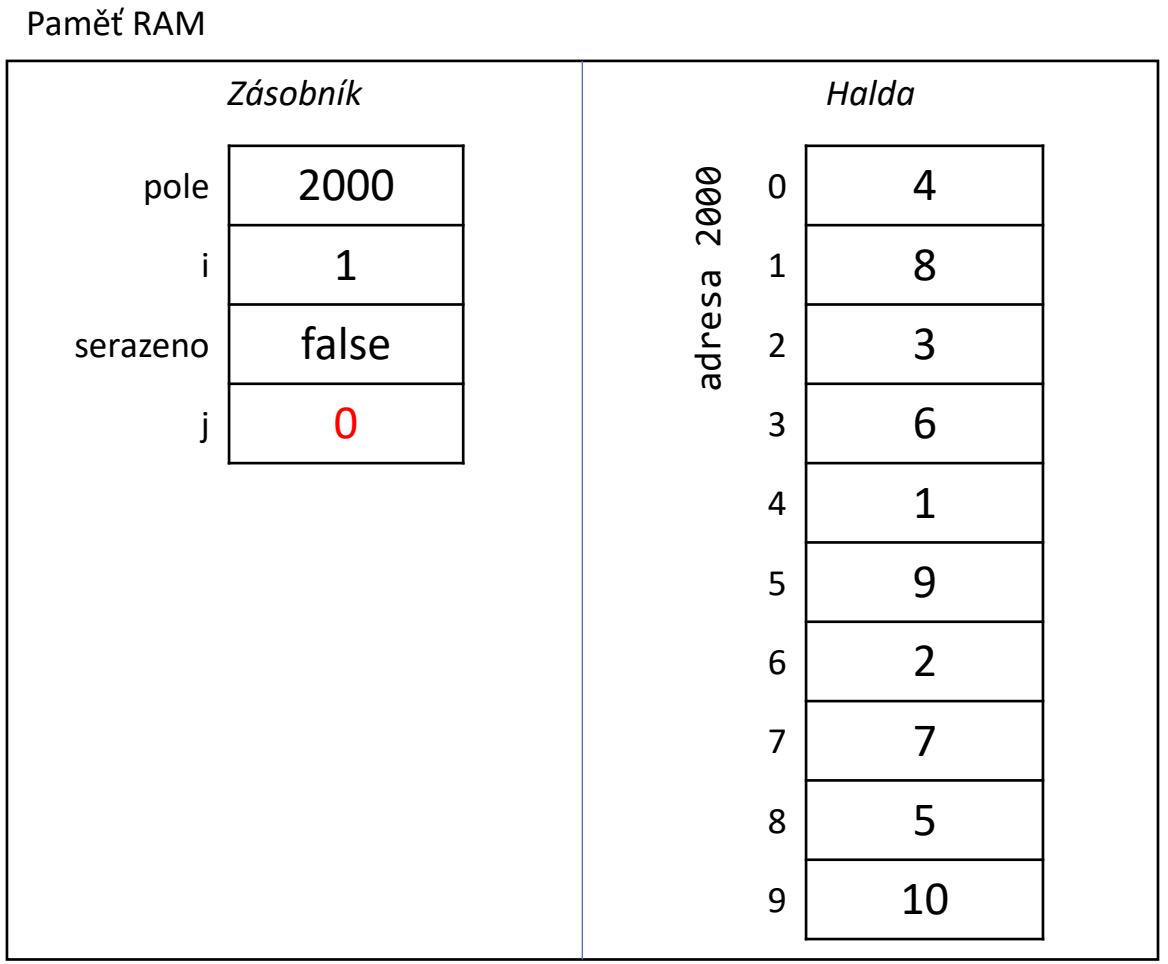
Bubble Sort

```
int[] pole = new int[]
{ 8, 4, 10, 3, 6, 1, 9, 2, 7, 5 };

for (int i = 0; i < pole.Length; i++)
{
    bool serazeno = true;

    for (int j = 0; j < pole.Length - 1 - i; j++)
    {
        if (pole[j] > pole[j + 1])
        {
            int tmp = pole[j];
            pole[j] = pole[j + 1];
            pole[j + 1] = tmp;

            serazeno = false;
        }
    }

    if (serazeno) break;
}
```



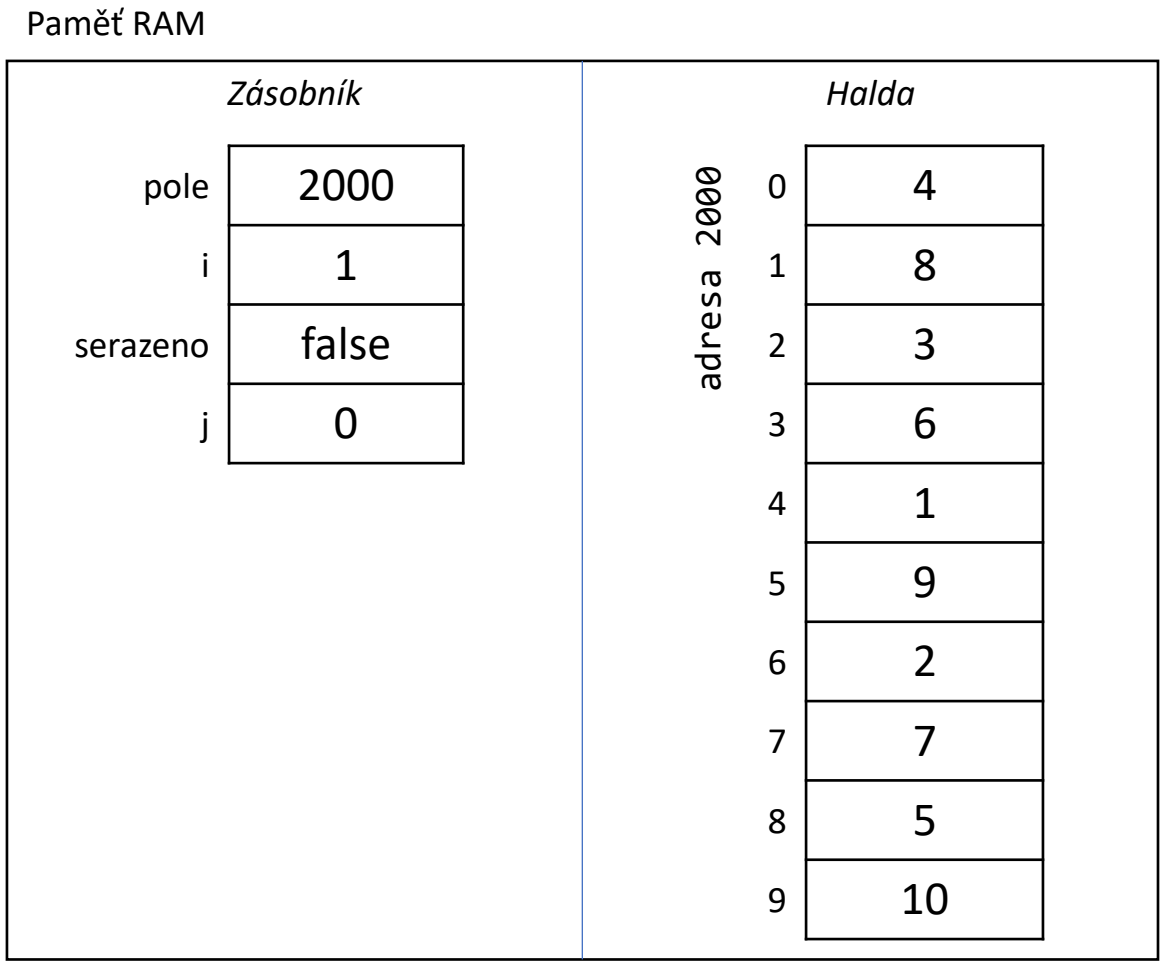
Bubble Sort

```
int[] pole = new int[]
{ 8, 4, 10, 3, 6, 1, 9, 2, 7, 5 };

for (int i = 0; i < pole.Length; i++)
{
    bool serazeno = true;

    for (int j = 0; j < pole.Length - 1 - i; j++)
    {
        if (pole[j] > pole[j + 1])
        {
            int tmp = pole[j];
            pole[j] = pole[j + 1];
            pole[j + 1] = tmp;

            serazeno = false;
        }
    }

    if (serazeno) break;
}
```



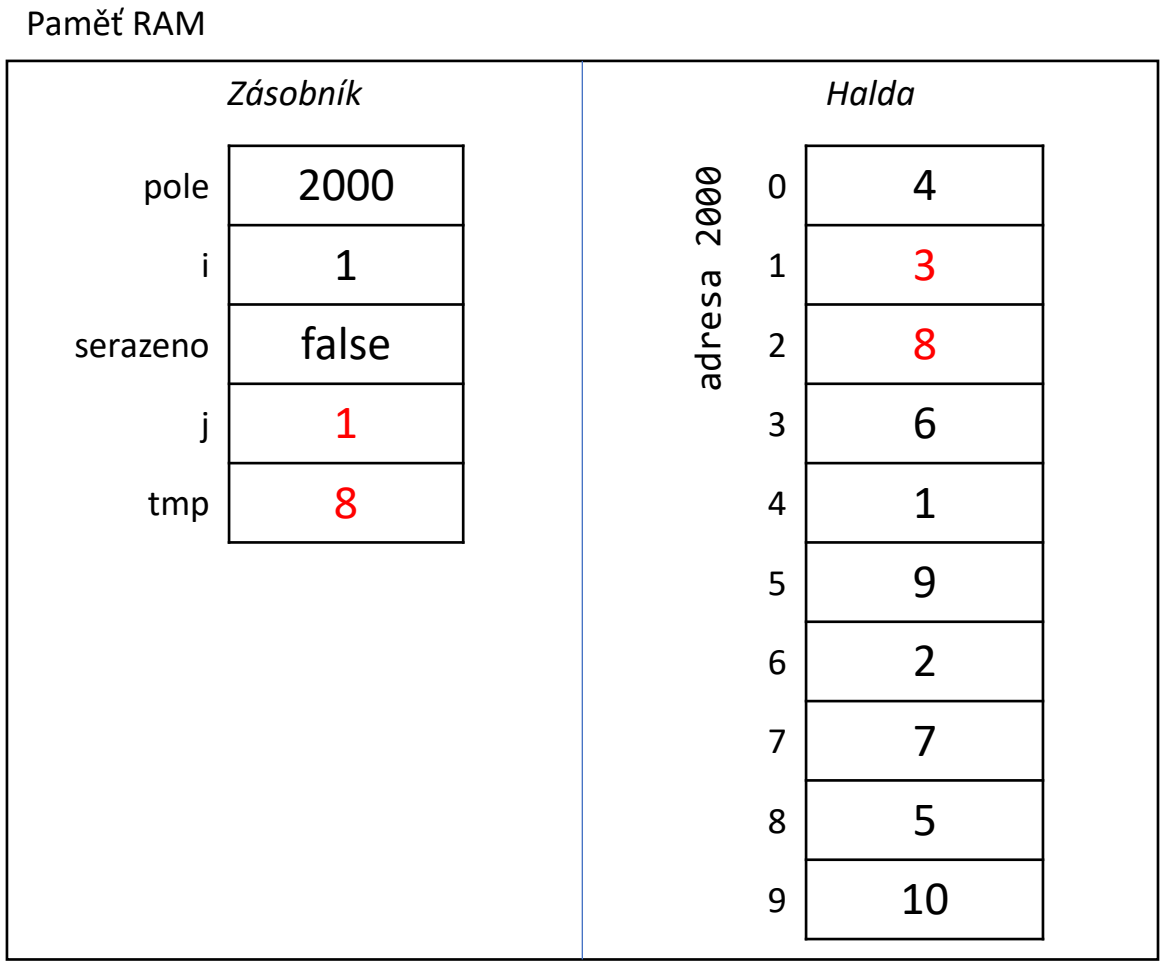
Bubble Sort

```
int[] pole = new int[]
{ 8, 4, 10, 3, 6, 1, 9, 2, 7, 5 };

for (int i = 0; i < pole.Length; i++)
{
    bool serazeno = true;

    for (int j = 0; j < pole.Length - 1 - i; j++)
    {           4        8
        if (pole[j] > pole[j + 1])
        {
            int tmp = pole[j];
            pole[j] = pole[j + 1];
            pole[j + 1] = tmp;

            serazeno = false;
        }
    }

    if (serazeno) break;
}
```



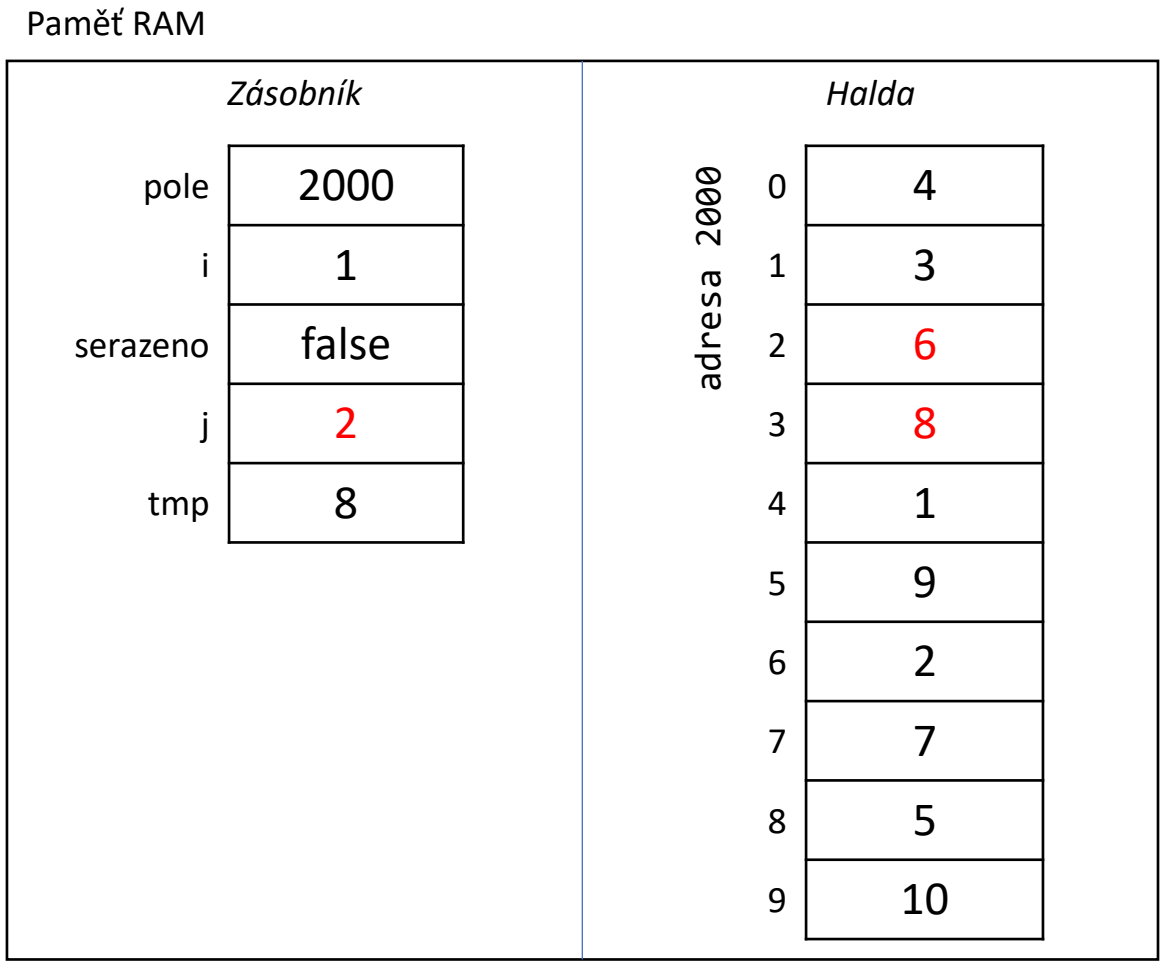
Bubble Sort

```
int[] pole = new int[]
{ 8, 4, 10, 3, 6, 1, 9, 2, 7, 5 };

for (int i = 0; i < pole.Length; i++)
{
    bool serazeno = true;

    for (int j = 0; j < pole.Length - 1 - i; j++)
    {          8        3
        if (pole[j] > pole[j + 1])
        {
            int tmp = pole[j];
            pole[j] = pole[j + 1];
            pole[j + 1] = tmp;

            serazeno = false;
        }
    }

    if (serazeno) break;
}
```



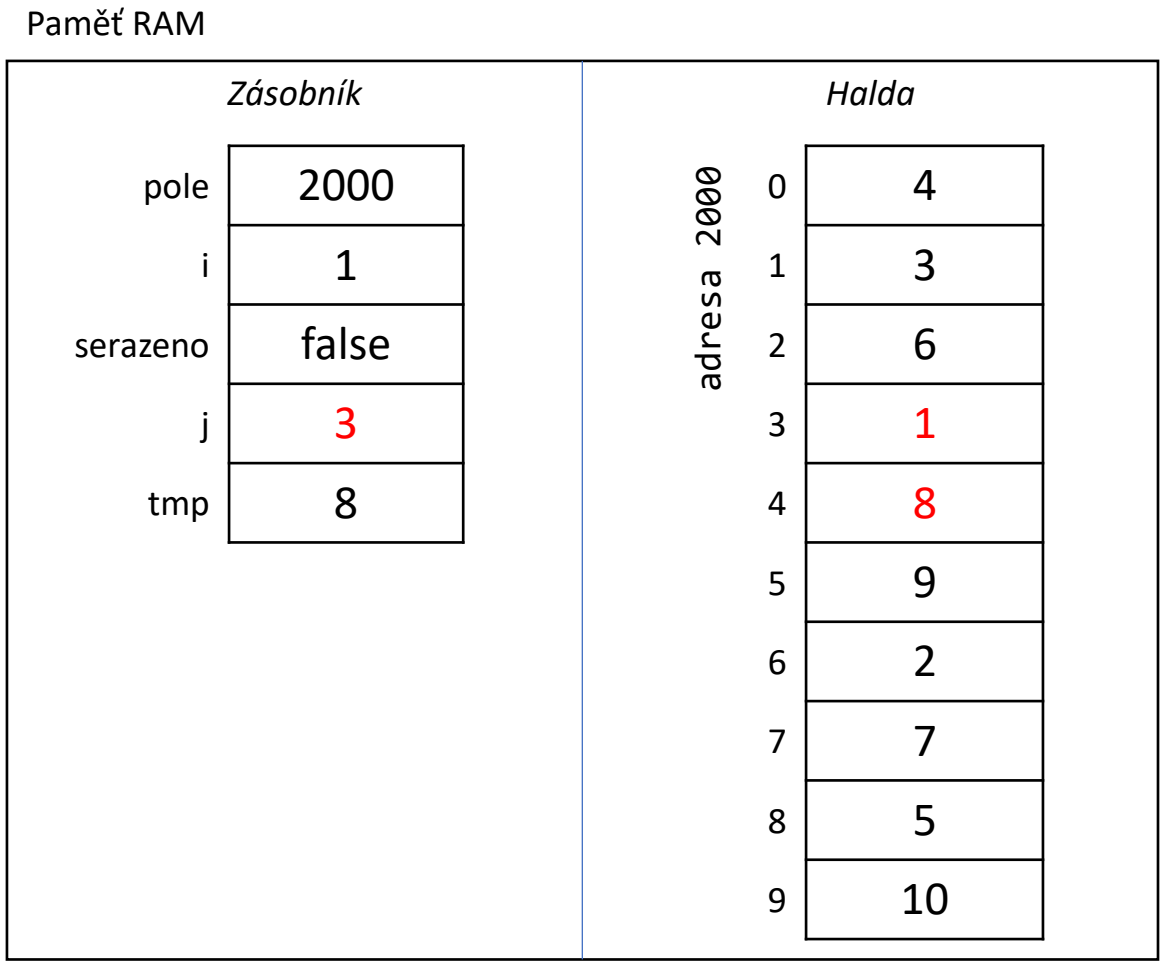
Bubble Sort

```
int[] pole = new int[]
{ 8, 4, 10, 3, 6, 1, 9, 2, 7, 5 };

for (int i = 0; i < pole.Length; i++)
{
    bool serazeno = true;

    for (int j = 0; j < pole.Length - 1 - i; j++)
    {          8        6
        if (pole[j] > pole[j + 1])
        {
            int tmp = pole[j];
            pole[j] = pole[j + 1];
            pole[j + 1] = tmp;

            serazeno = false;
        }
    }

    if (serazeno) break;
}
```



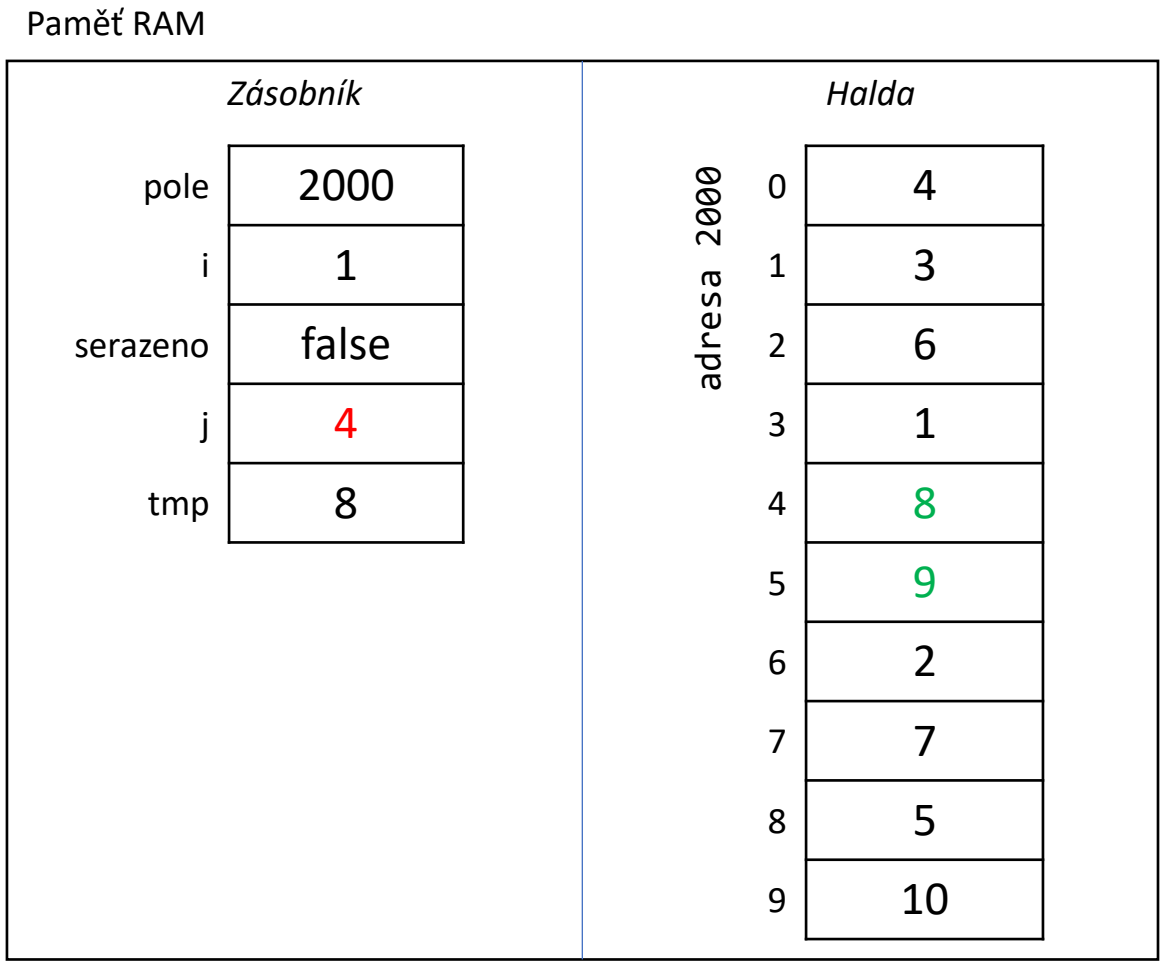
Bubble Sort

```
int[] pole = new int[]
{ 8, 4, 10, 3, 6, 1, 9, 2, 7, 5 };

for (int i = 0; i < pole.Length; i++)
{
    bool serazeno = true;

    for (int j = 0; j < pole.Length - 1 - i; j++)
    {           8        1
        if (pole[j] > pole[j + 1])
        {
            int tmp = pole[j];
            pole[j] = pole[j + 1];
            pole[j + 1] = tmp;

            serazeno = false;
        }
    }

    if (serazeno) break;
}
```



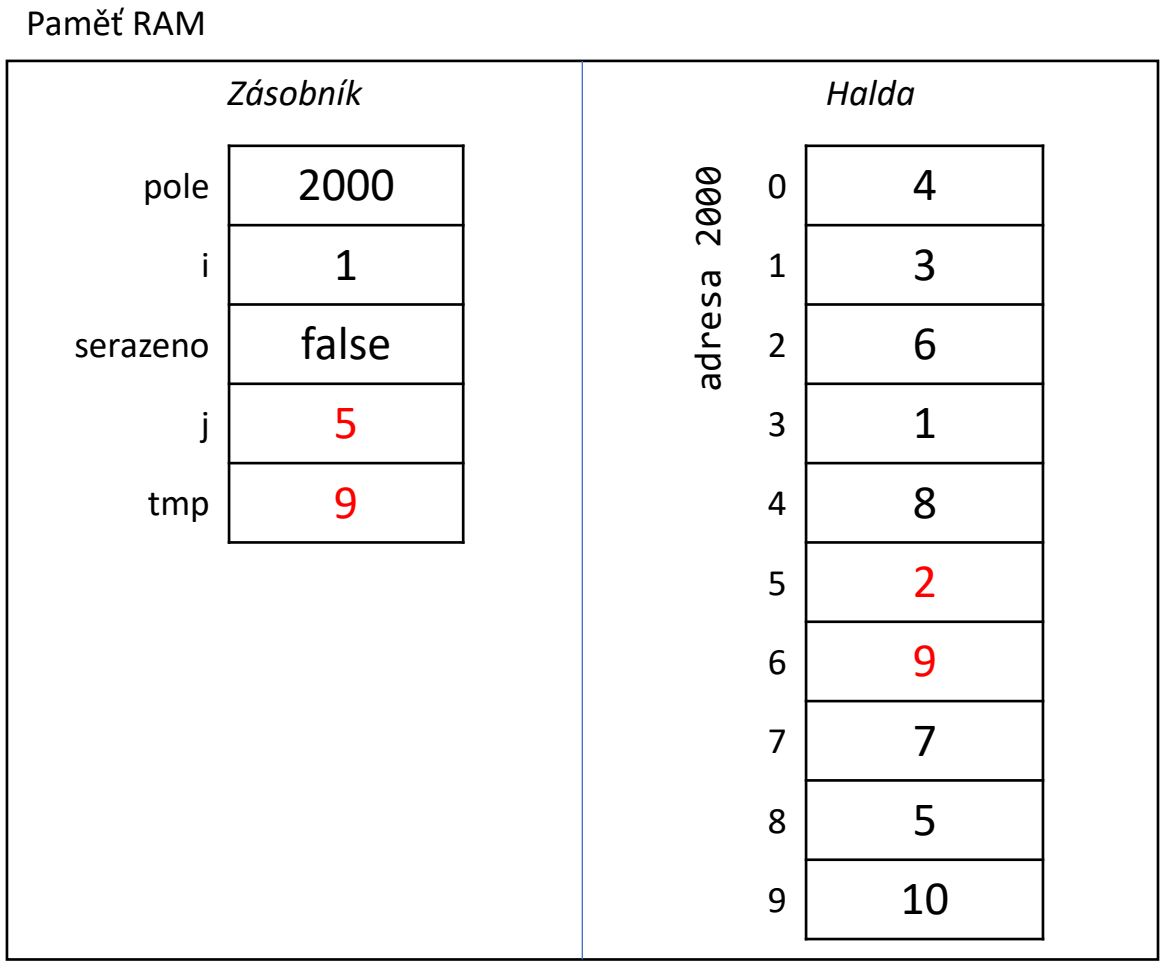
Bubble Sort

```
int[] pole = new int[]
{ 8, 4, 10, 3, 6, 1, 9, 2, 7, 5 };

for (int i = 0; i < pole.Length; i++)
{
    bool serazeno = true;

    for (int j = 0; j < pole.Length - 1 - i; j++)
    {           8        9
        if (pole[j] > pole[j + 1])
        {
            int tmp = pole[j];
            pole[j] = pole[j + 1];
            pole[j + 1] = tmp;

            serazeno = false;
        }
    }

    if (serazeno) break;
}
```



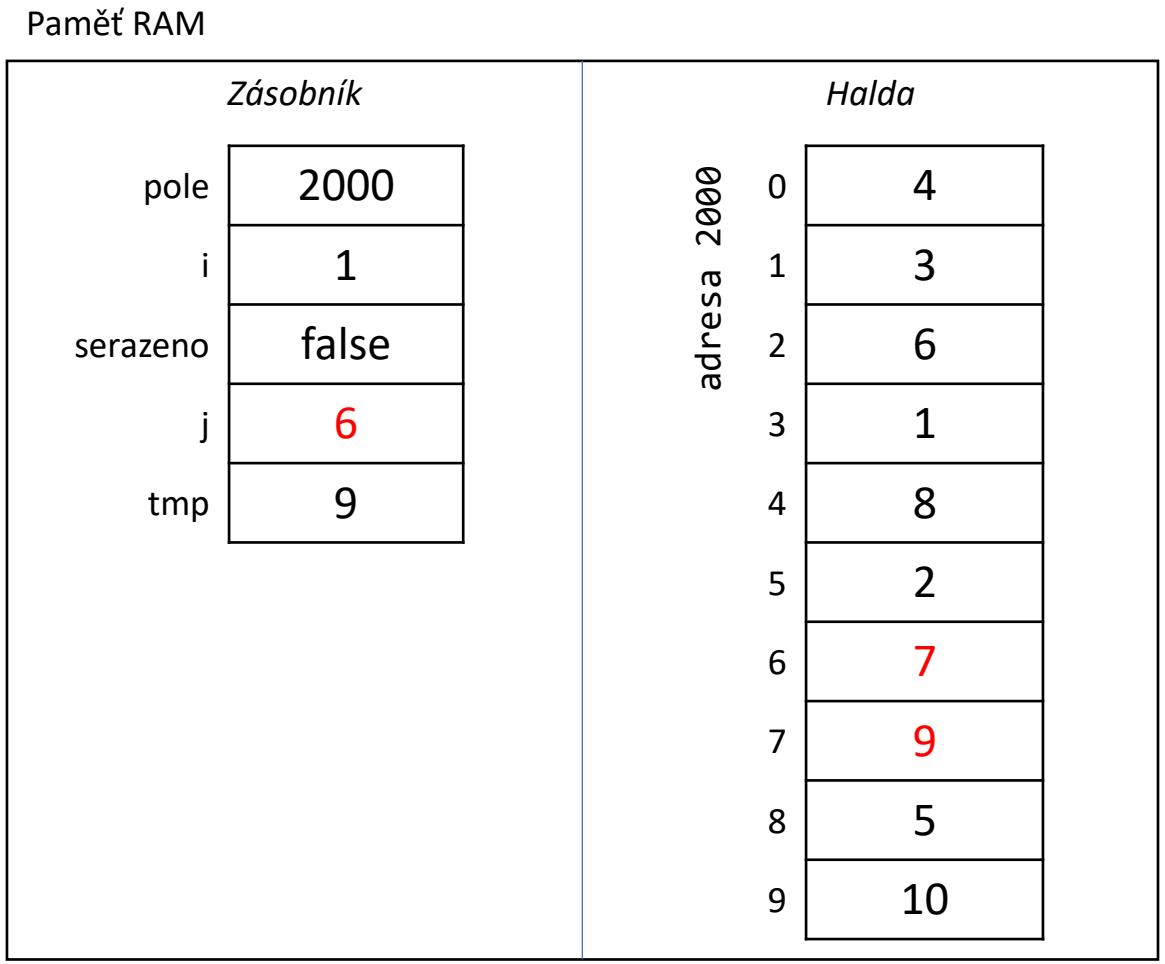
Bubble Sort

```
int[] pole = new int[]
{ 8, 4, 10, 3, 6, 1, 9, 2, 7, 5 };

for (int i = 0; i < pole.Length; i++)
{
    bool serazeno = true;

    for (int j = 0; j < pole.Length - 1 - i; j++)
    {          9        2
        if (pole[j] > pole[j + 1])
        {
            int tmp = pole[j];
            pole[j] = pole[j + 1];
            pole[j + 1] = tmp;

            serazeno = false;
        }
    }

    if (serazeno) break;
}
```



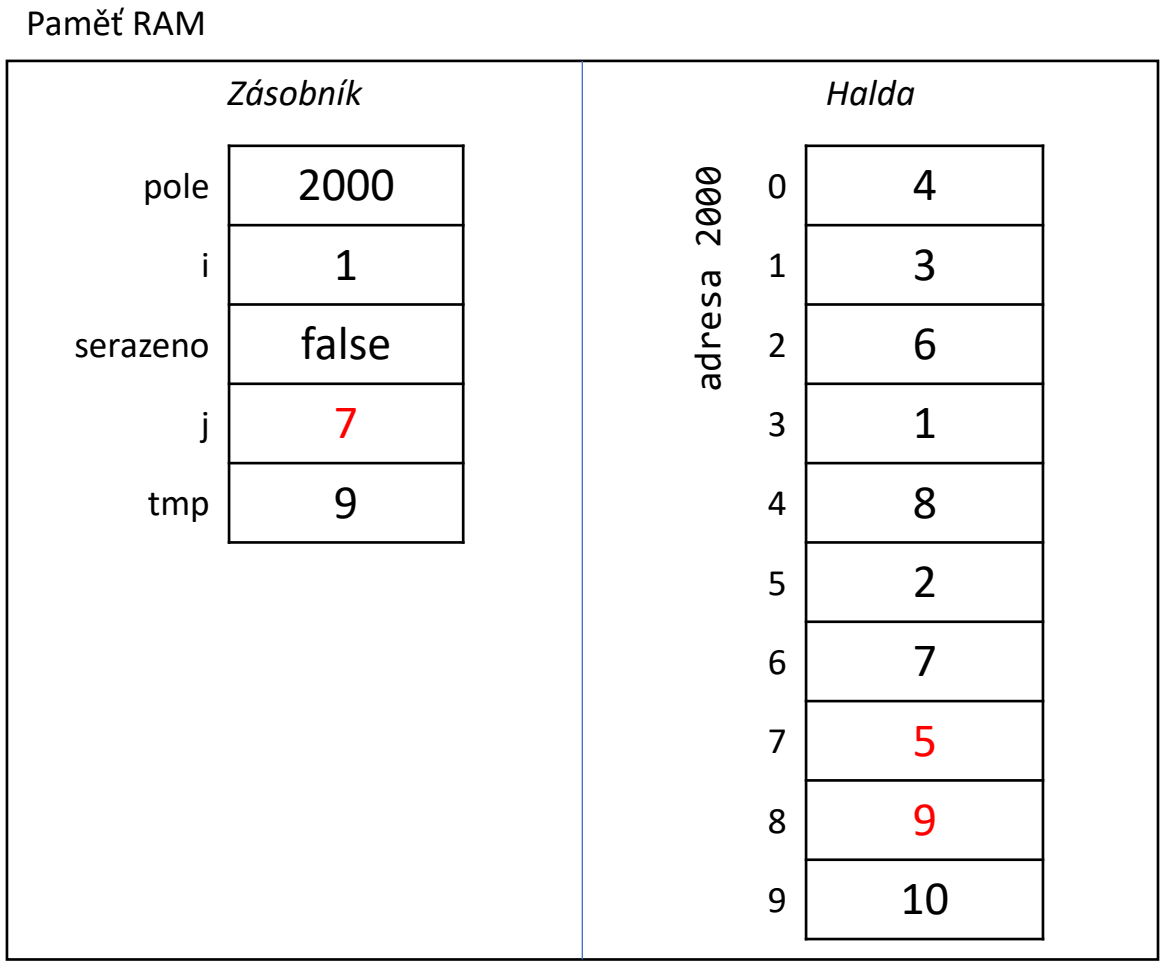
Bubble Sort

```
int[] pole = new int[]
{ 8, 4, 10, 3, 6, 1, 9, 2, 7, 5 };

for (int i = 0; i < pole.Length; i++)
{
    bool serazeno = true;

    for (int j = 0; j < pole.Length - 1 - i; j++)
    {          9        7
        if (pole[j] > pole[j + 1])
        {
            int tmp = pole[j];
            pole[j] = pole[j + 1];
            pole[j + 1] = tmp;

            serazeno = false;
        }
    }

    if (serazeno) break;
}
```



Bubble Sort

```
int[] pole = new int[]
{ 8, 4, 10, 3, 6, 1, 9, 2, 7, 5 };

for (int i = 0; i < pole.Length; i++)
{
    bool serazeno = true;

    for (int j = 0; j < pole.Length - 1 - i; j++)
    {          9        5
        if (pole[j] > pole[j + 1])
        {
            int tmp = pole[j];
            pole[j] = pole[j + 1];
            pole[j + 1] = tmp;

            serazeno = false;
        }
    }

    if (serazeno) break;
}
```



Bubble Sort – ukončení algoritmu

- Vzhledem k množství iterací nebude procházet další kroky.
- Algoritmus se ukončí až se ani jednou neprovede prohození prvků a nebo když byly provedeny všechny iterace.

Použité zdroje

[1] Single-Dimensional Arrays - C# Programming Guide | Microsoft Docs. [online]. Copyright © Microsoft 2021 [cit. 02.02.2021]. Dostupné z: https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/arrays/single-dimensional-arrays

[2] break statement - C# Reference | Microsoft Docs. [online]. Copyright © Microsoft 2021 [cit. 25.02.2021]. Dostupné z: https://docs.microsoft.com/en-us/dotnet/csharp/language-reference/keywords/break

[3] Bubble sort. *Algoritmus* [online]. Copyright © 2015 [cit. 25.02.2021]. Dostupné z: https://www.algoritmy.net/article/3/Bubble-sort

[4] Lego Bubble Sort - YouTube. *YouTube* [online]. Copyright © 2021 Google LLC [cit. 25.02.2021]. Dostupné z: https://www.youtube.com/watch?v=MtcrEhrt_K0

EVROPSKÁ UNIE
Evropské strukturální a investiční fondy
Operační program Výzkum, vývoj a vzdělávání

MINISTERSTVO ŠKOLSTVÍ, MLÁDEŽE A TĚLOVÝCHOVY

Programování a algoritmizace

Děkuji za pozornost

Strategický projekt UTB ve Zlíně, reg. č. CZ.02.2.69/0.0/0.0/16_015/0002204

Ing. et Ing. Erik Král, Ph.D.
FAI, ÚPKS