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

MINISTERSTVO ŠKOLSTVÍ, MLÁDEŽE A TĚLOVÝCHOVY

Programování a algoritmizace

Insertion 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

Insertion sort

Popis implementace

Úvod

- V následujících snímcích probereme algoritmus Insertion Sort [1].
- Na těchto příkladech si demonstrujeme práci s jednorozměrným polem s pevnou délkou [2] a cyklus *for* [3].

Insertion Sort

- Algoritmus Insertion Sort představuje jednoduchý algoritmus pro seřazení prvků v poli.
- Má časovou složitosti $O(n^2)$ [1], ale může být použit při vkládání prvku do už seřazeného pole, například u spojového lineárního seznamu, který umožňuje efektivně měnit vkládat a odebírat prvky.
- Postupně procházíme pole a zařazujeme ho do již seřazené části pole.
- Na začátku má seřazené pole rozměr jeden prvek.

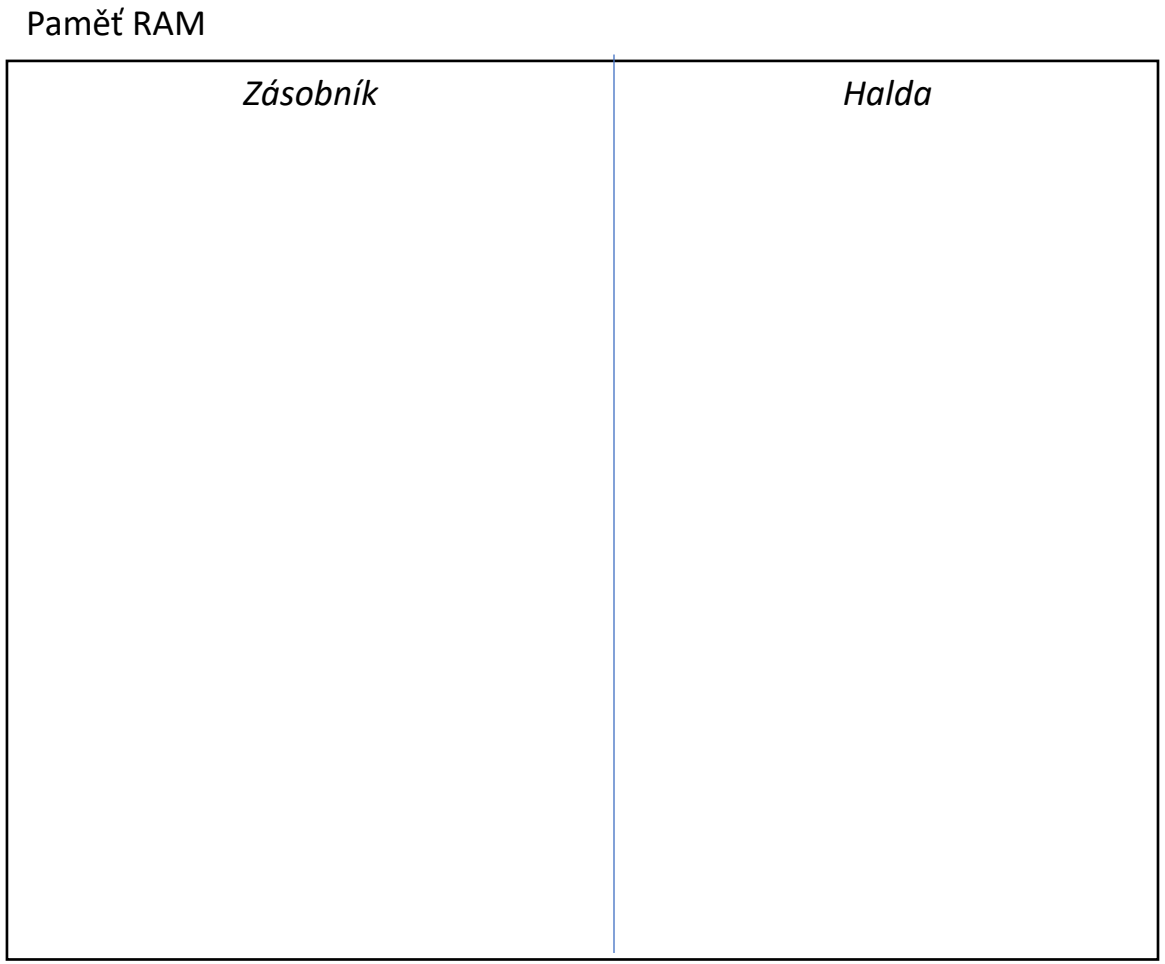
Animace Insertion Sort [4]

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.

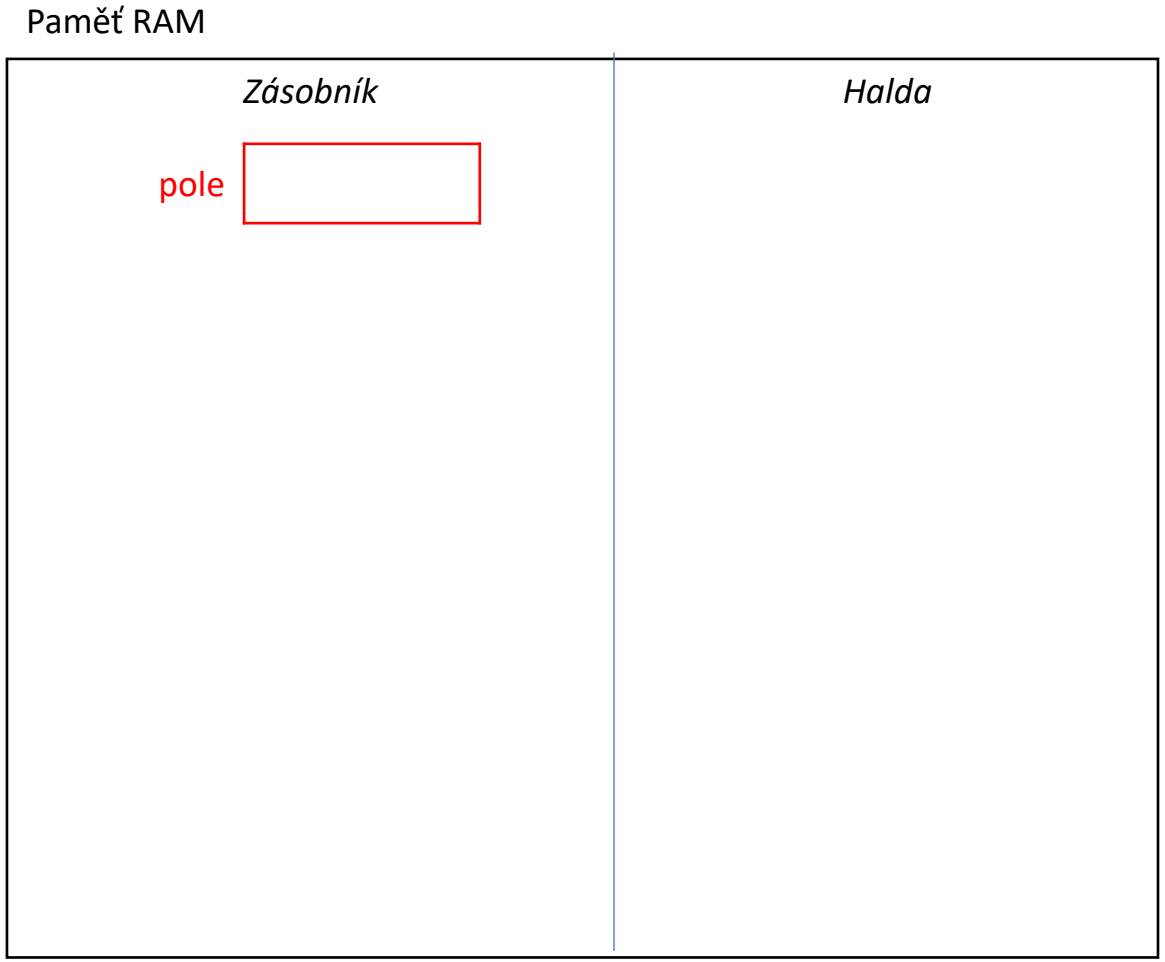
Insertion Sort

```
int[] pole = new int[]
{ 5, 9, 1, 4, 0, 8, 6, 7, 2, 3 };

int index = 0;

for (int i = index; i < pole.Length - 1; i++)
{
    int j = i + 1;
    int prvek = pole[j];

    while (j > 0 && prvek < pole[j - 1])
    {
        pole[j] = pole[j - 1];
        --j;
    }

    pole[j] = prvek;
}
```



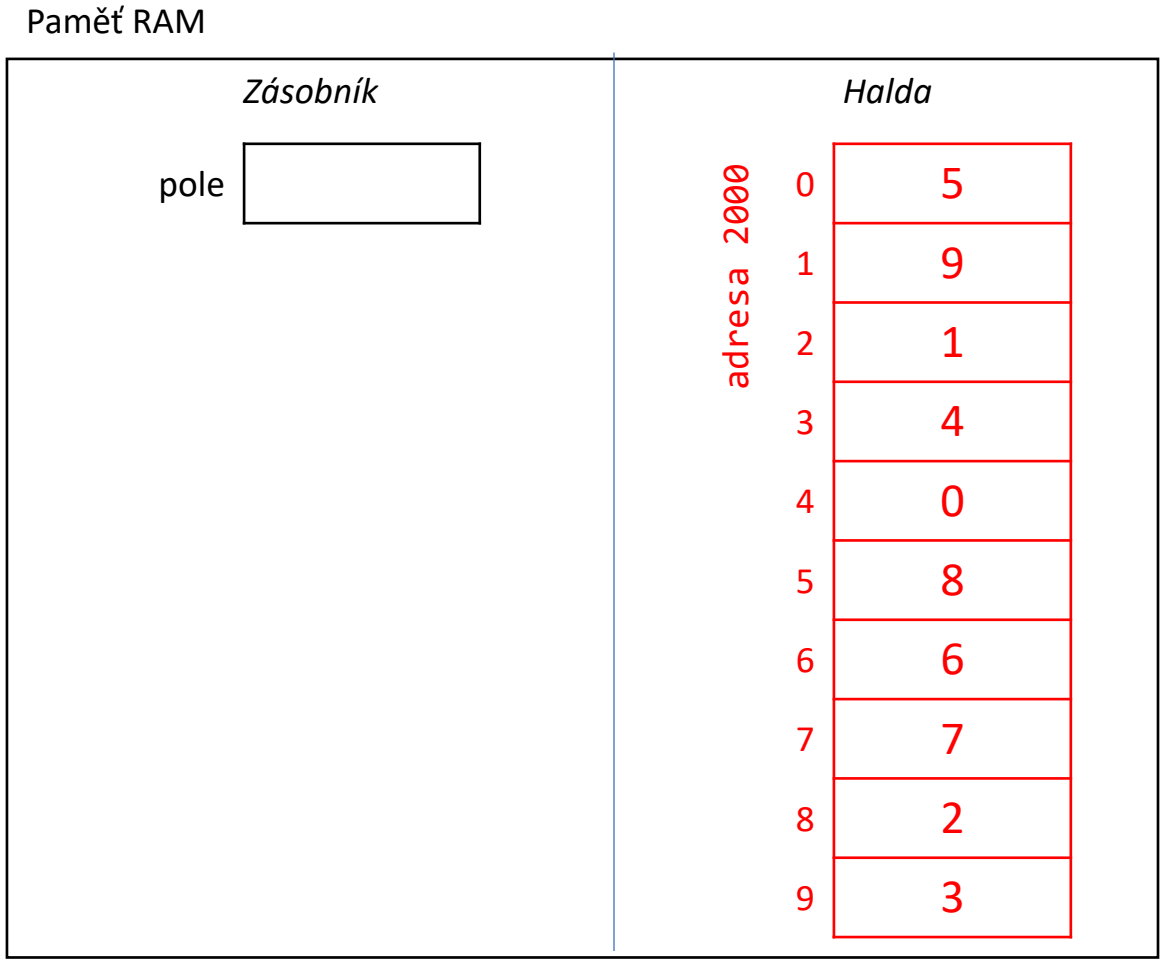
Insertion Sort

```
int[] pole = new int[]
{ 5, 9, 1, 4, 0, 8, 6, 7, 2, 3 };
```



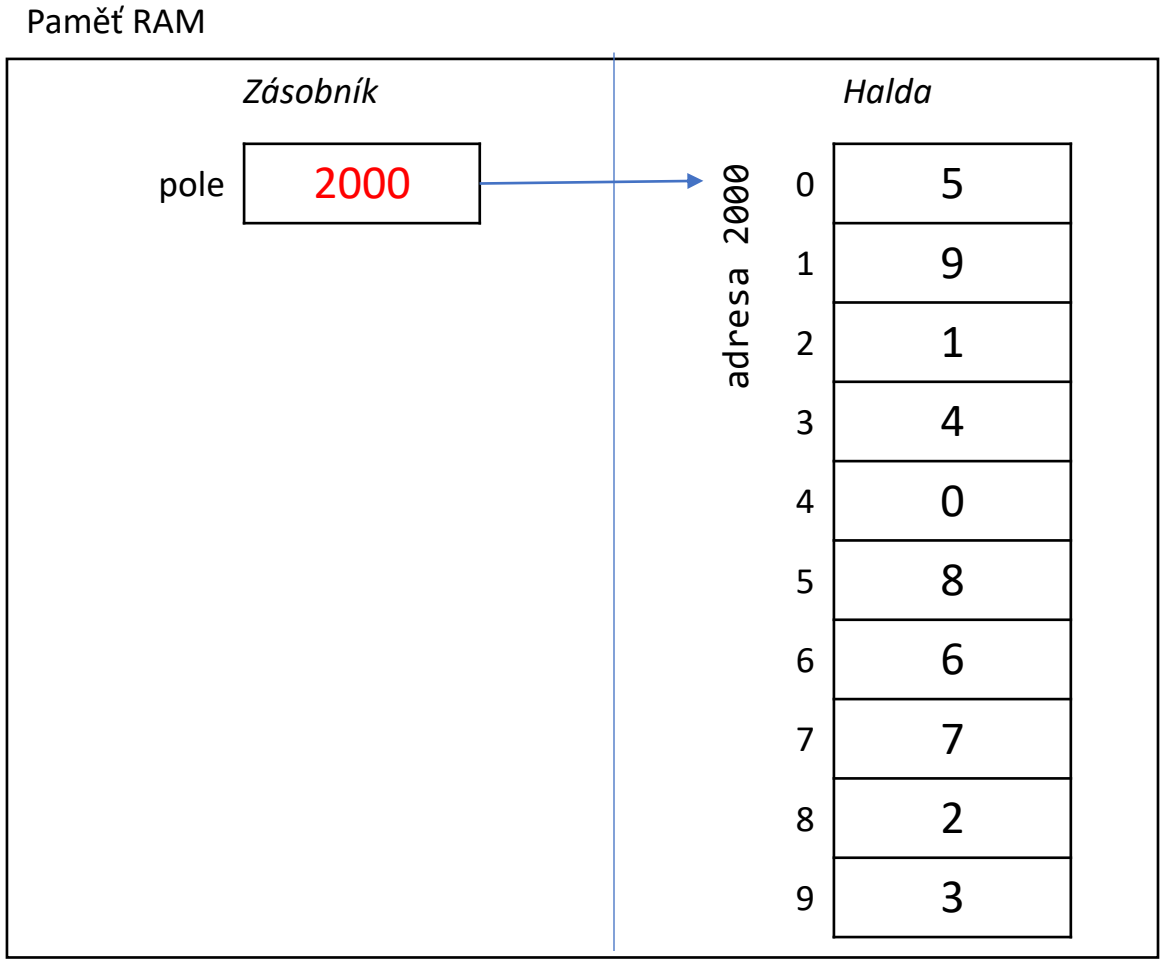
Insertion Sort

```
int[] pole = new int[]
{ 5, 9, 1, 4, 0, 8, 6, 7, 2, 3 };
```



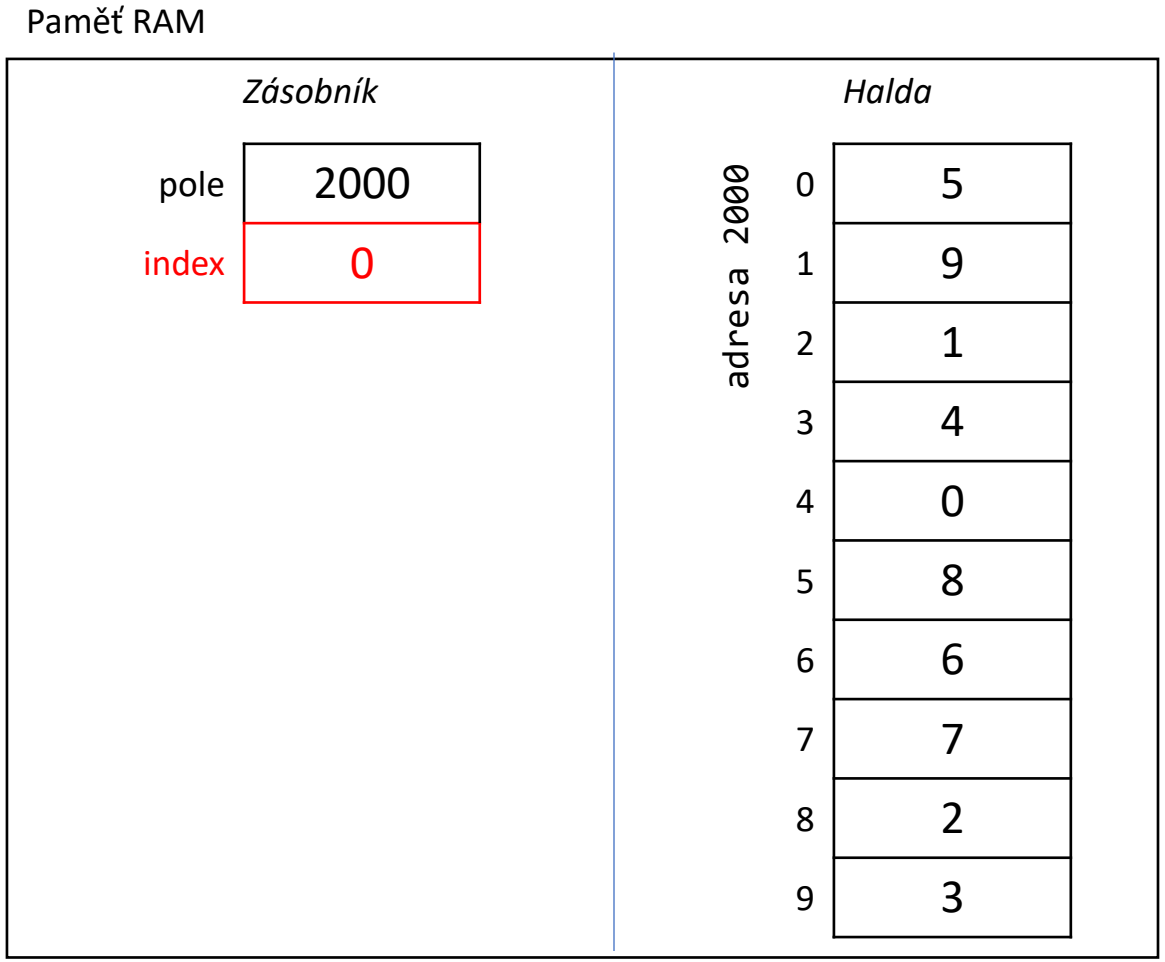
Insertion Sort

```
int[] pole = new int[]
{ 5, 9, 1, 4, 0, 8, 6, 7, 2, 3 };
```



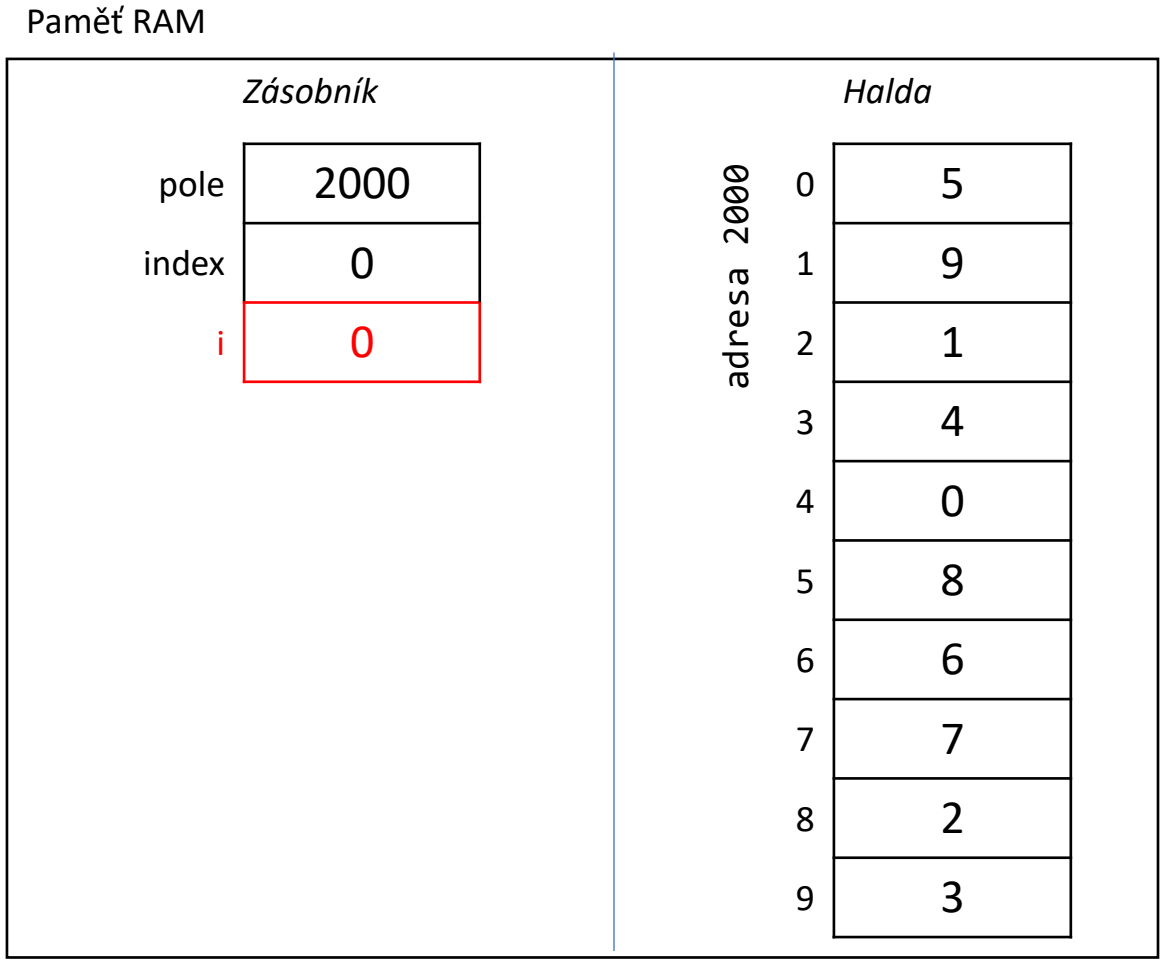
Insertion Sort

```
int[] pole = new int[]
{ 5, 9, 1, 4, 0, 8, 6, 7, 2, 3 };

int index = 0;
```



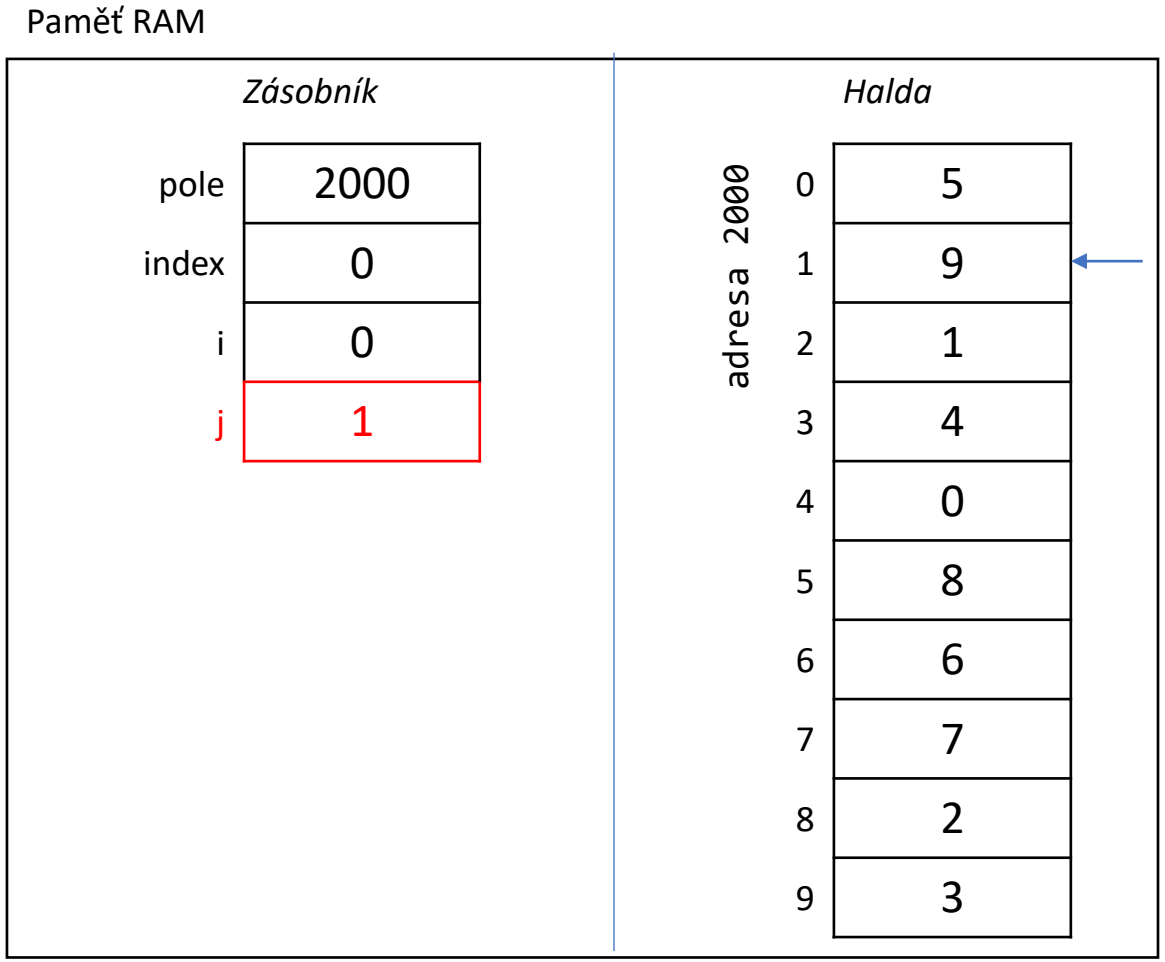
Insertion Sort

```
int[] pole = new int[]
{ 5, 9, 1, 4, 0, 8, 6, 7, 2, 3 };

int index = 0;

for (int i = index; i < pole.Length - 1; i++)
{
    int j = i + 1;
    int prvek = pole[j];

    while (j > 0 && prvek < pole[j - 1])
    {
        pole[j] = pole[j - 1];
        --j;
    }

    pole[j] = prvek;
}
```



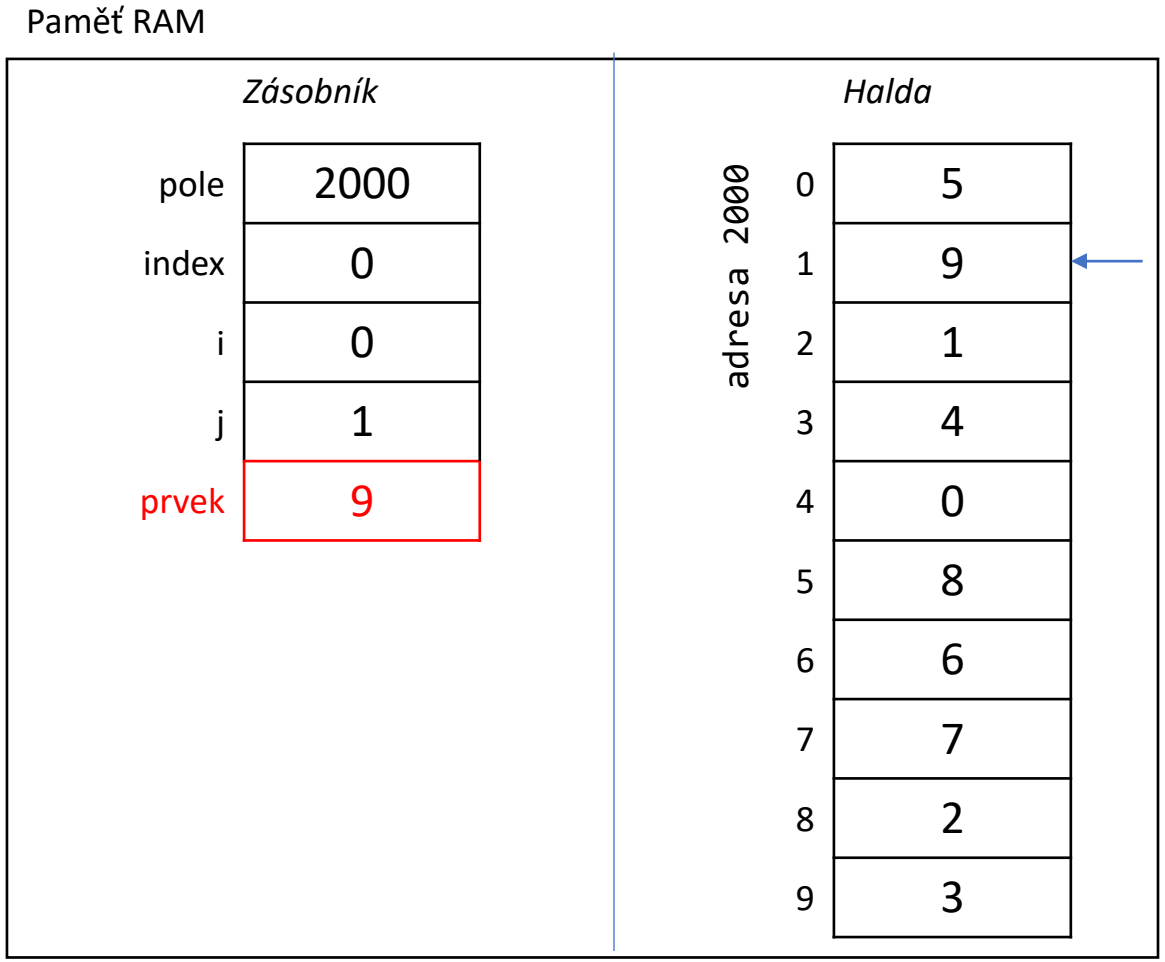
Insertion Sort

```
int[] pole = new int[]
{ 5, 9, 1, 4, 0, 8, 6, 7, 2, 3 };

int index = 0;

for (int i = index; i < pole.Length - 1; i++)
{
    int j = i + 1;
    int prvek = pole[j];

    while (j > 0 && prvek < pole[j - 1])
    {
        pole[j] = pole[j - 1];
        --j;
    }

    pole[j] = prvek;
}
```



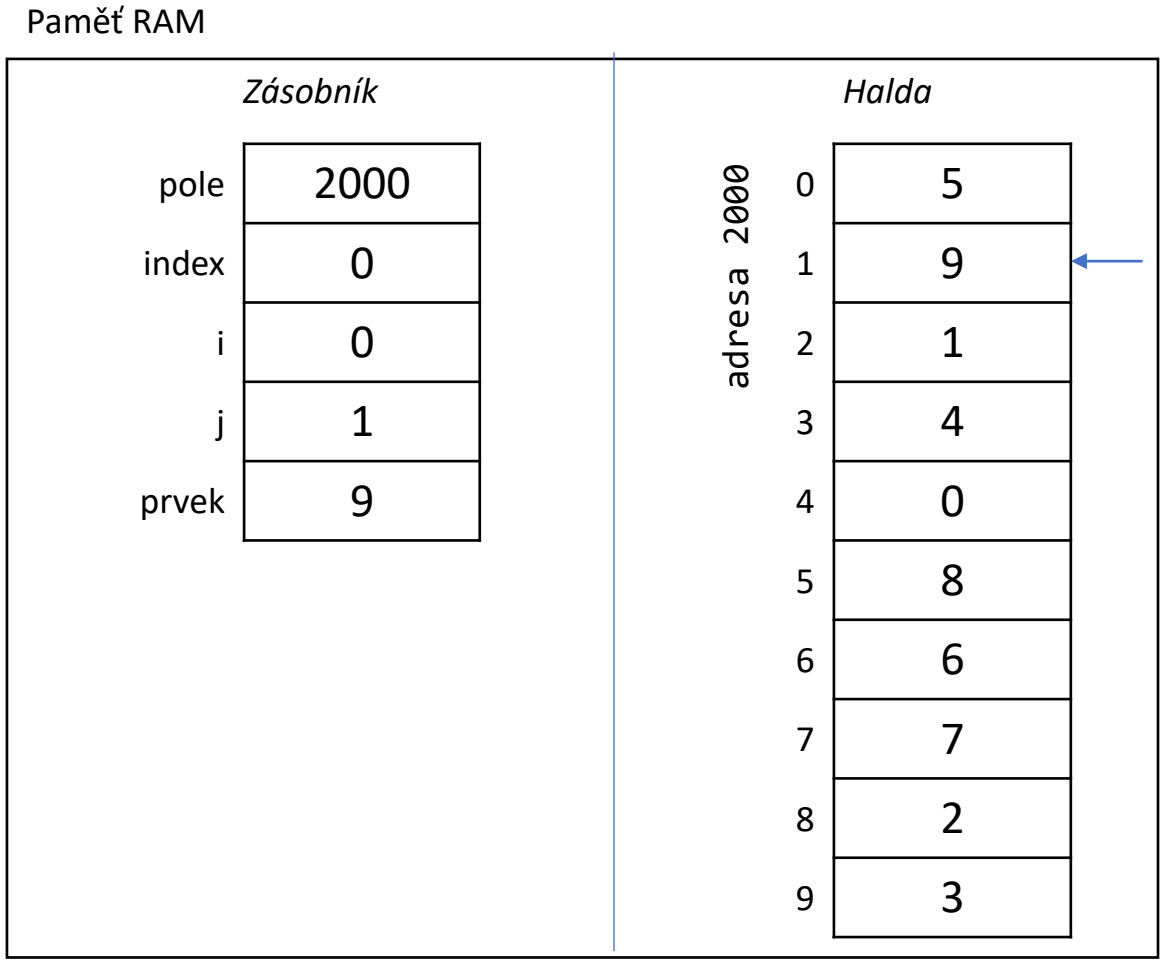
Insertion Sort

```
int[] pole = new int[]
{ 5, 9, 1, 4, 0, 8, 6, 7, 2, 3 };

int index = 0;

for (int i = index; i < pole.Length - 1; i++)
{
    int j = i + 1;
    int prvek = pole[j];

    while (j > 0 && prvek < pole[j - 1])
    {
        pole[j] = pole[j - 1];
        --j;
    }

    pole[j] = prvek;
}
```



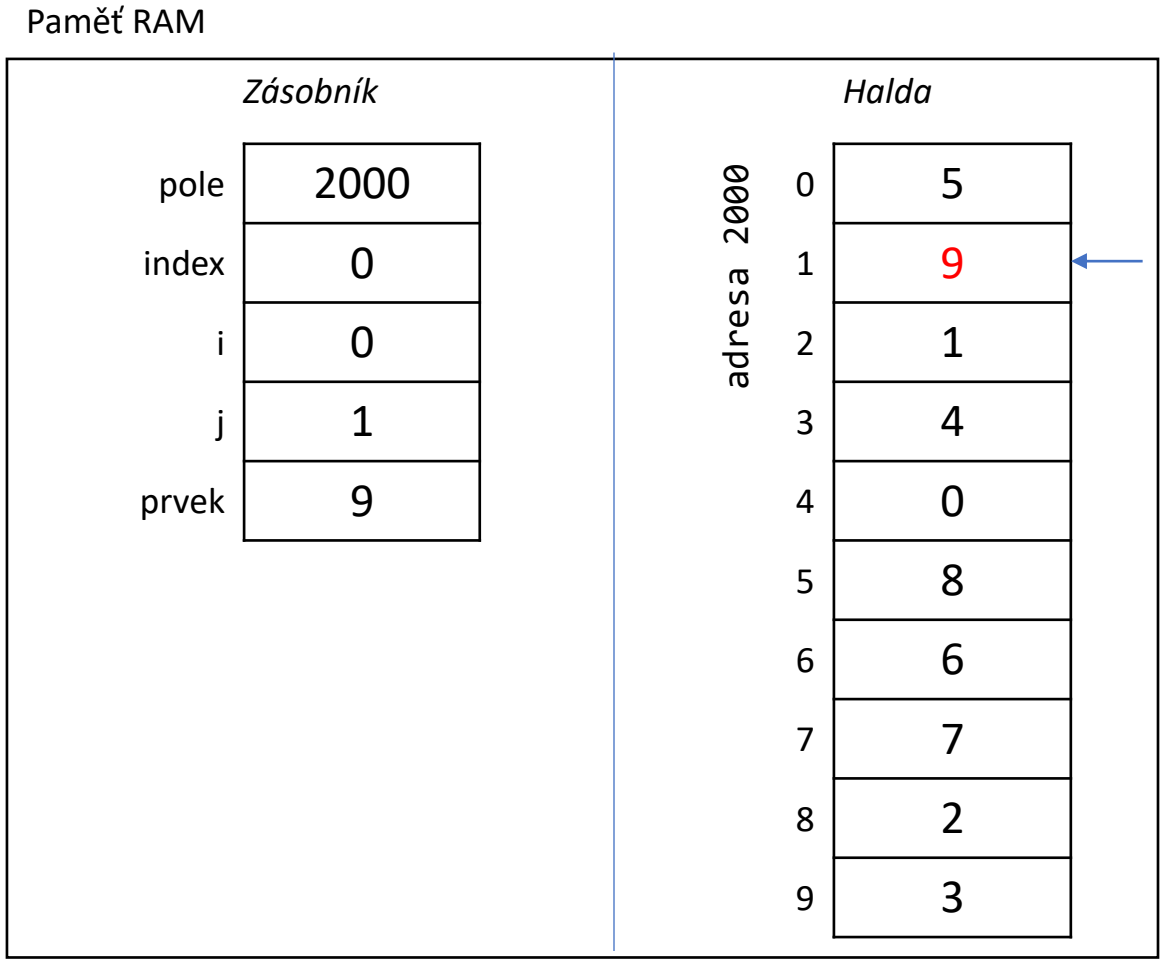
Insertion Sort

```
int[] pole = new int[]
{ 5, 9, 1, 4, 0, 8, 6, 7, 2, 3 };

int index = 0;

for (int i = index; i < pole.Length - 1; i++)
{
    int j = i + 1;
    int prvek = pole[j];
                        9       5
    while (j > 0 && prvek < pole[j - 1])
    {
        pole[j] = pole[j - 1];
        --j;
    }

    pole[j] = prvek;
}
```



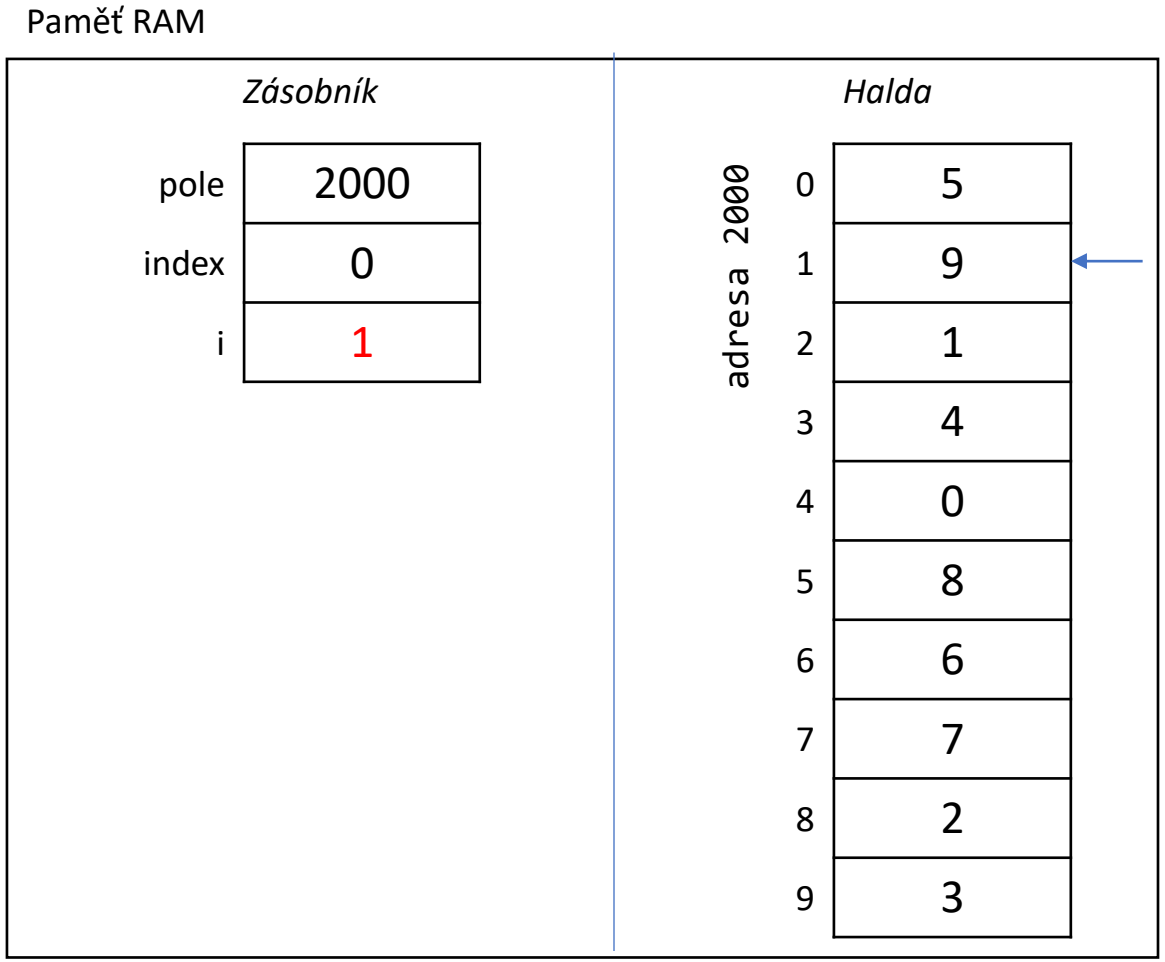
Insertion Sort

```
int[] pole = new int[]
{ 5, 9, 1, 4, 0, 8, 6, 7, 2, 3 };

int index = 0;

for (int i = index; i < pole.Length - 1; i++)
{
    int j = i + 1;
    int prvek = pole[j];

    while (j > 0 && prvek < pole[j - 1])
    {
        pole[j] = pole[j - 1];
        --j;
    }

    pole[j] = prvek;
}
```



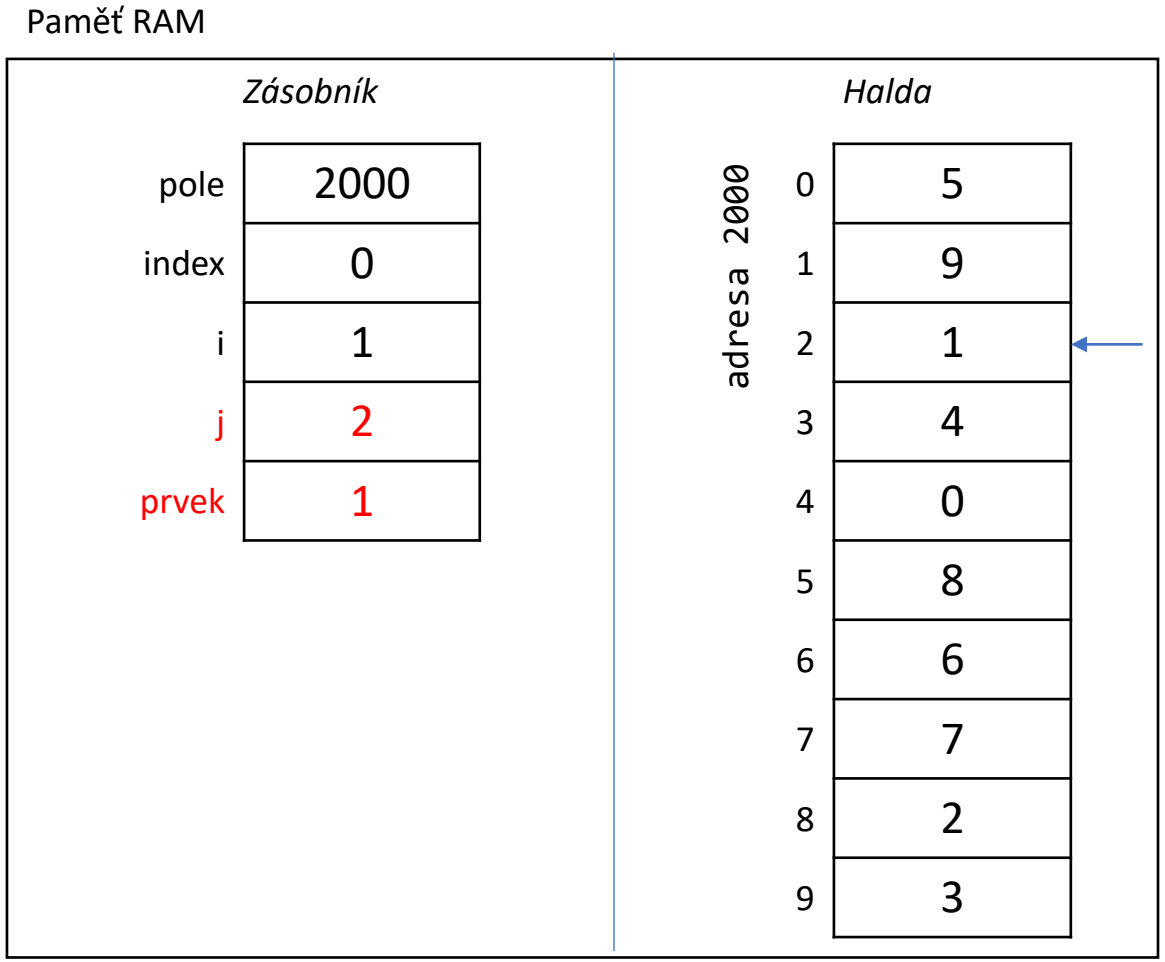
Insertion Sort

```
int[] pole = new int[]
{ 5, 9, 1, 4, 0, 8, 6, 7, 2, 3 };

int index = 0;

for (int i = index; i < pole.Length - 1; i++)
{
    int j = i + 1;
    int prvek = pole[j];

    while (j > 0 && prvek < pole[j - 1])
    {
        pole[j] = pole[j - 1];
        --j;
    }

    pole[j] = prvek;
}
```



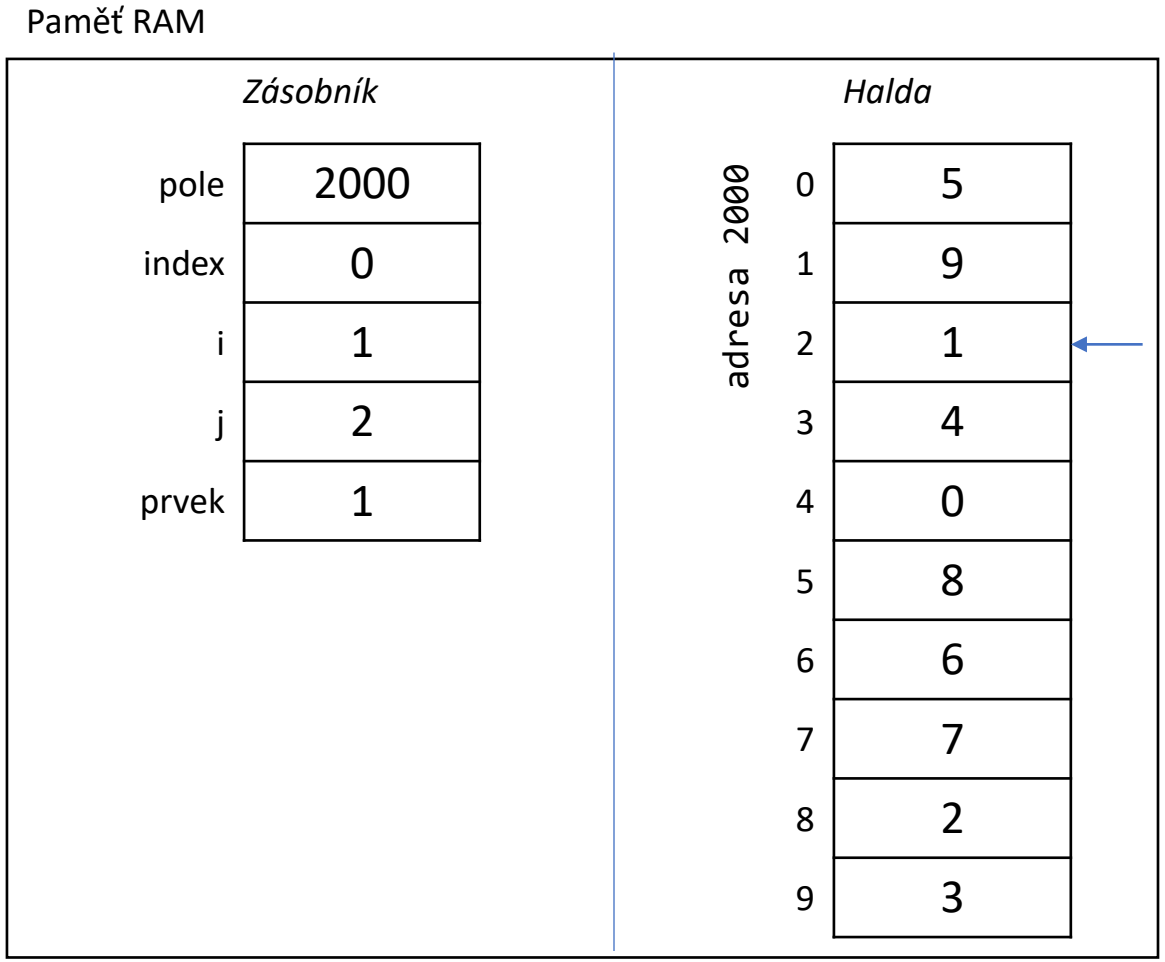
Insertion Sort

```
int[] pole = new int[]
{ 5, 9, 1, 4, 0, 8, 6, 7, 2, 3 };

int index = 0;

for (int i = index; i < pole.Length - 1; i++)
{
    int j = i + 1;
    int prvek = pole[j];

    while (j > 0 && prvek < pole[j - 1])
    {
        pole[j] = pole[j - 1];
        --j;
    }

    pole[j] = prvek;
}
```



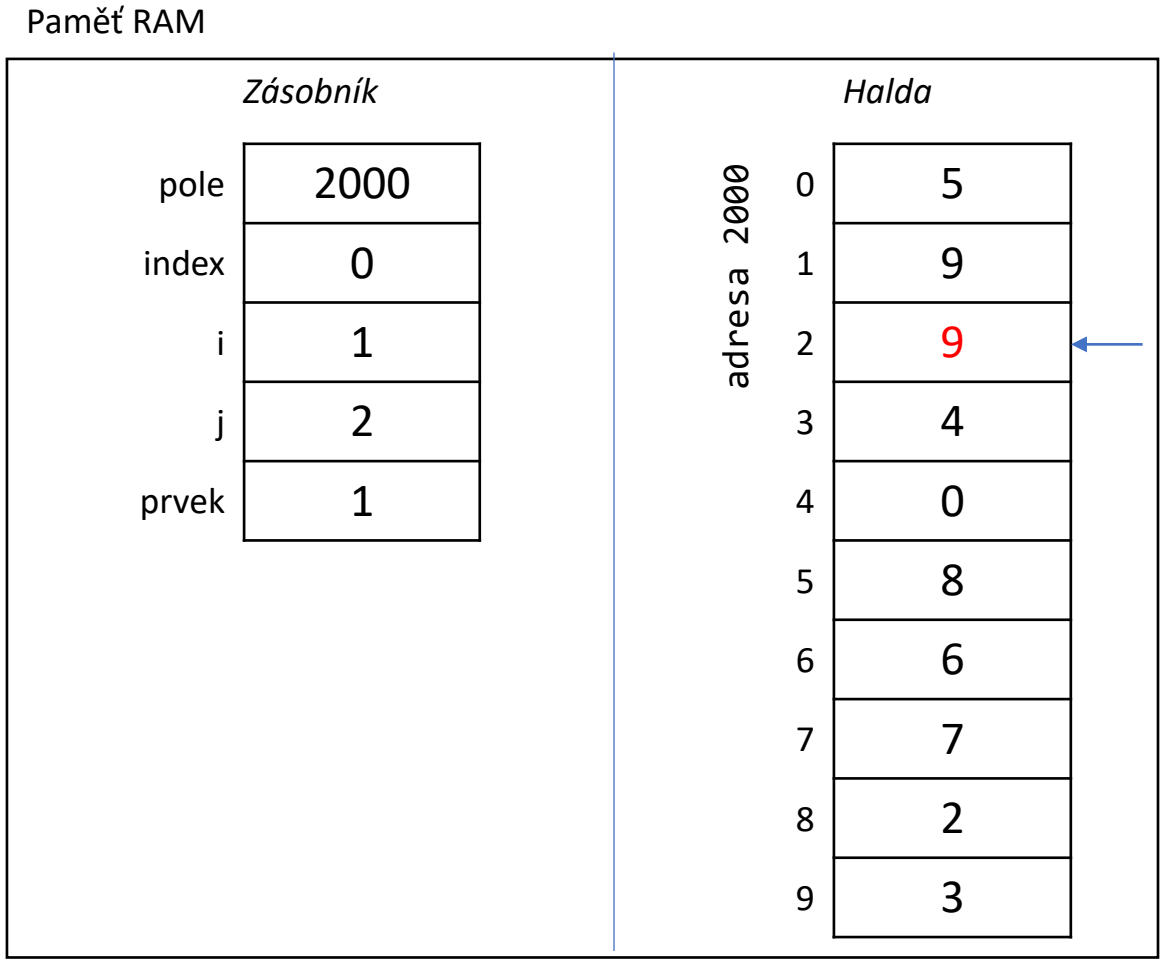
Insertion Sort

```
int[] pole = new int[]
{ 5, 9, 1, 4, 0, 8, 6, 7, 2, 3 };

int index = 0;

for (int i = index; i < pole.Length - 1; i++)
{
    int j = i + 1;
    int prvek = pole[j];
                        1       9
    while (j > 0 && prvek < pole[j - 1])
    {
        pole[j] = pole[j - 1];
        --j;
    }

    pole[j] = prvek;
}
```



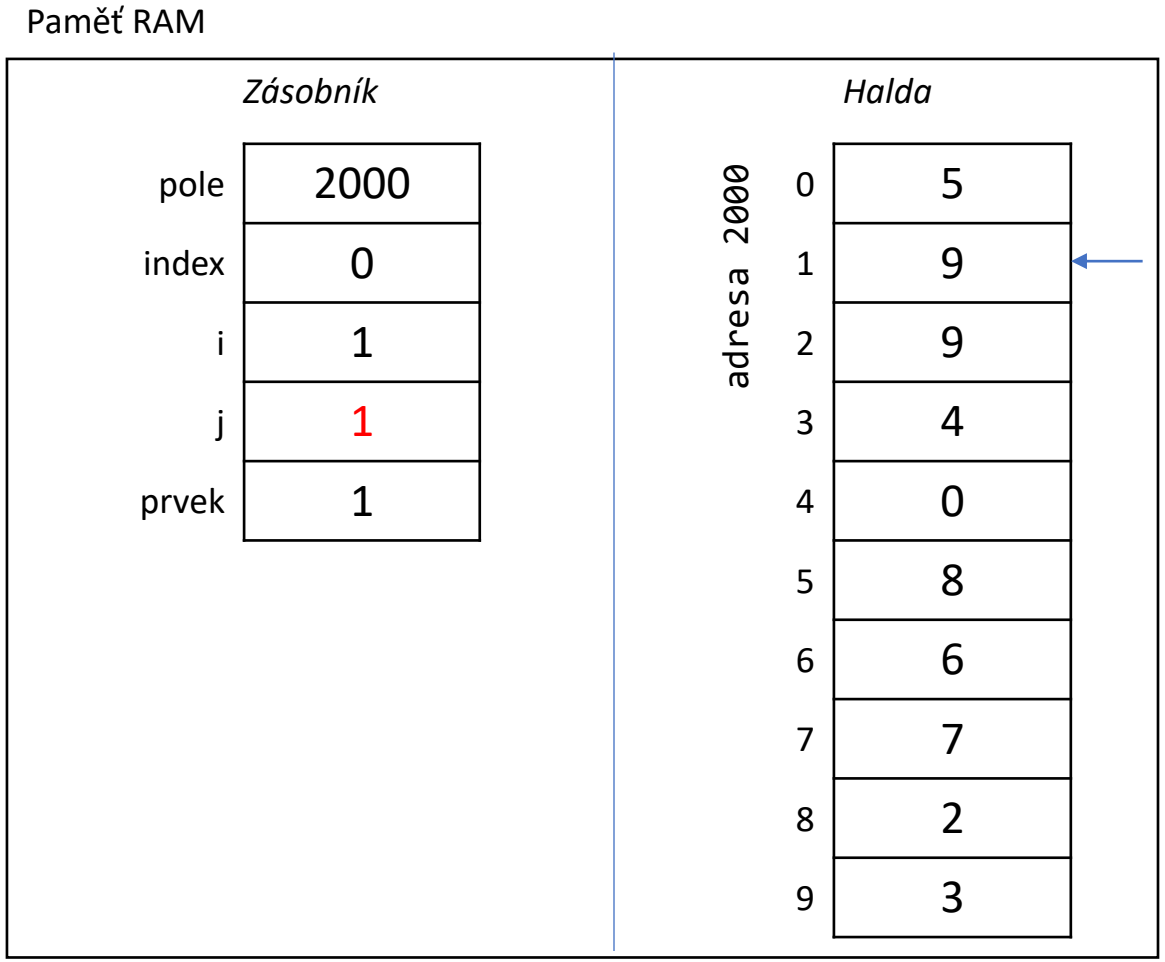
Insertion Sort

```
int[] pole = new int[]
{ 5, 9, 1, 4, 0, 8, 6, 7, 2, 3 };

int index = 0;

for (int i = index; i < pole.Length - 1; i++)
{
    int j = i + 1;
    int prvek = pole[j];

    while (j > 0 && prvek < pole[j - 1])
    {
        pole[j] = pole[j - 1];
        --j;
    }

    pole[j] = prvek;
}
```



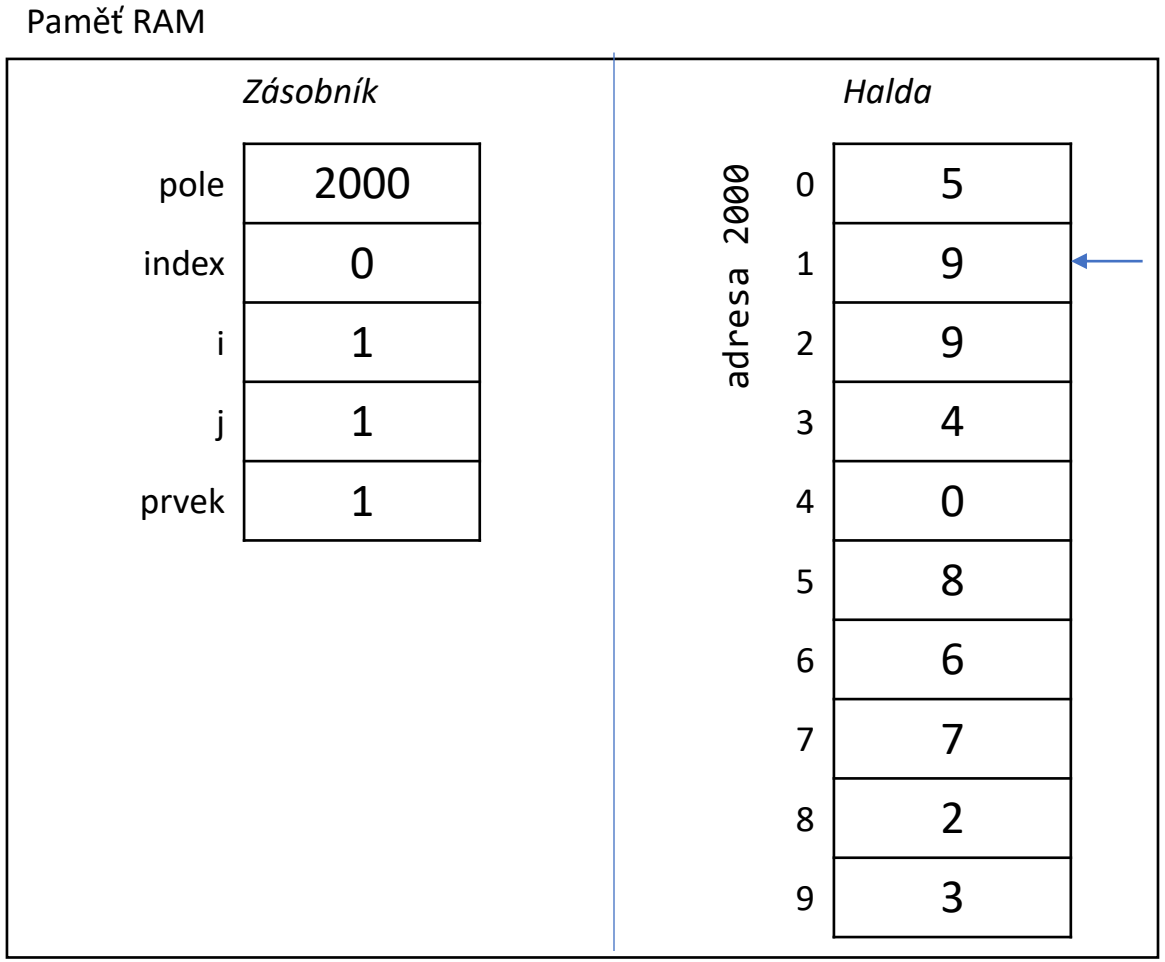
Insertion Sort

```
int[] pole = new int[]
{ 5, 9, 1, 4, 0, 8, 6, 7, 2, 3 };

int index = 0;

for (int i = index; i < pole.Length - 1; i++)
{
    int j = i + 1;
    int prvek = pole[j];

    while (j > 0 && prvek < pole[j - 1])
    {
        pole[j] = pole[j - 1];
        --j;
    }

    pole[j] = prvek;
}
```



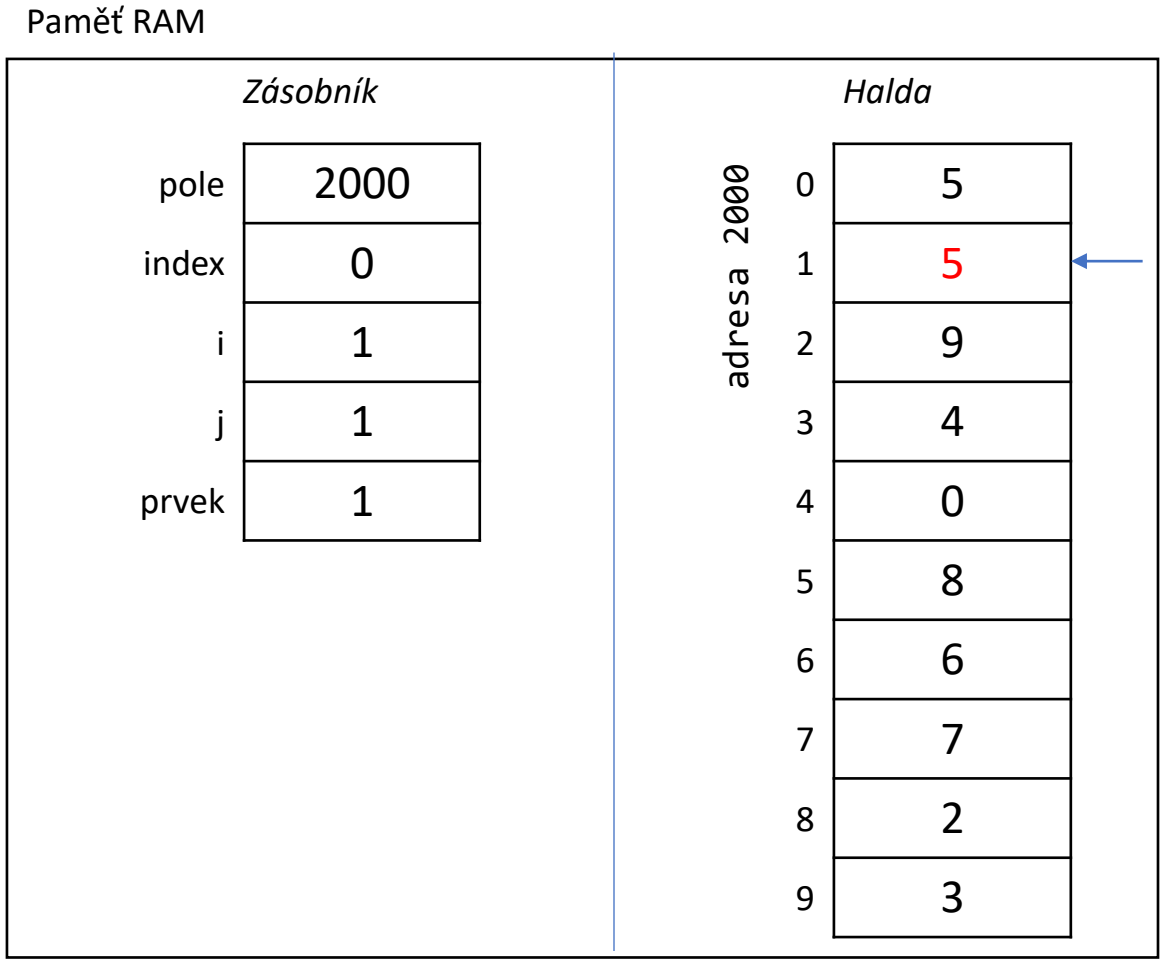
Insertion Sort

```
int[] pole = new int[]
{ 5, 9, 1, 4, 0, 8, 6, 7, 2, 3 };

int index = 0;

for (int i = index; i < pole.Length - 1; i++)
{
    int j = i + 1;
    int prvek = pole[j];
                        1       5
    while (j > 0 && prvek < pole[j - 1])
    {
        pole[j] = pole[j - 1];
        --j;
    }

    pole[j] = prvek;
}
```



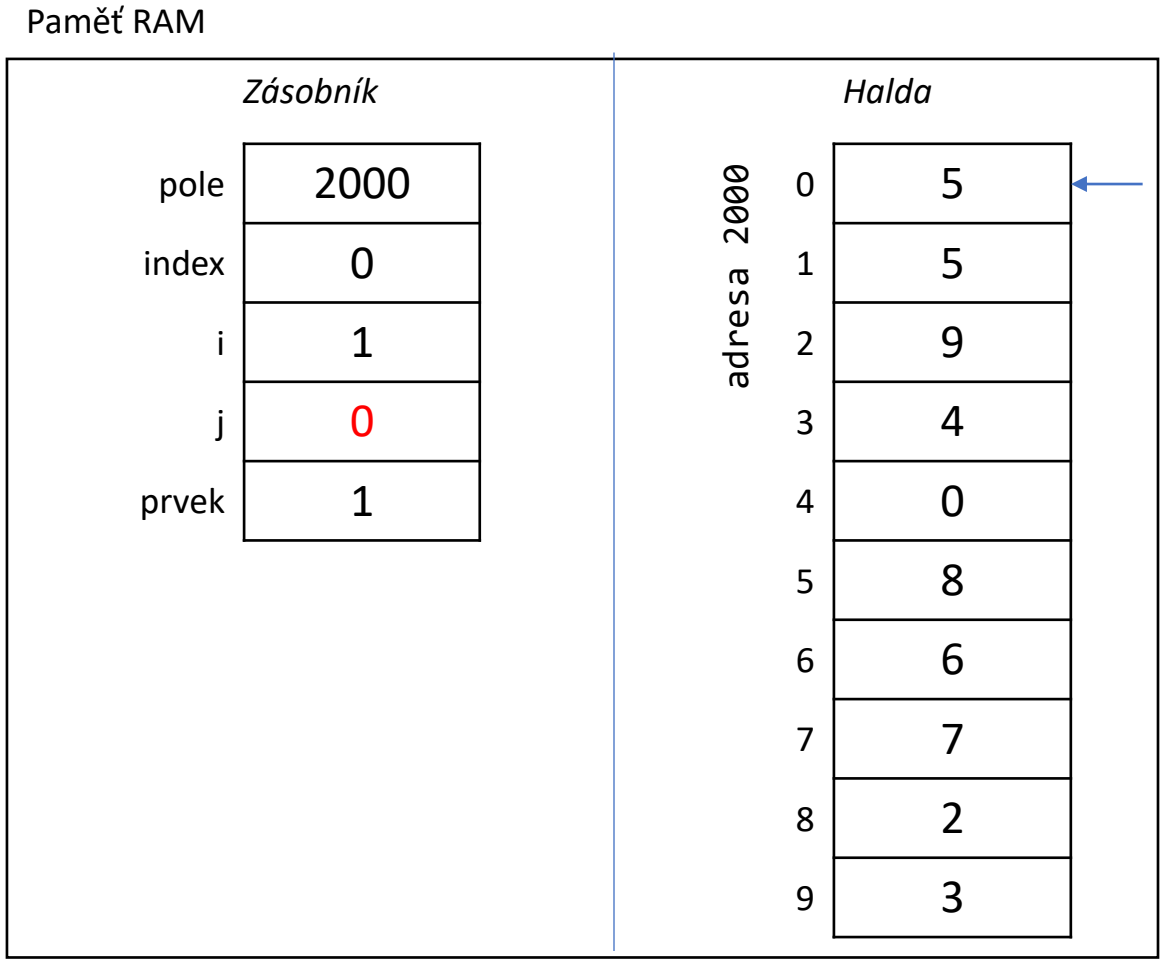
Insertion Sort

```
int[] pole = new int[]
{ 5, 9, 1, 4, 0, 8, 6, 7, 2, 3 };

int index = 0;

for (int i = index; i < pole.Length - 1; i++)
{
    int j = i + 1;
    int prvek = pole[j];

    while (j > 0 && prvek < pole[j - 1])
    {
        pole[j] = pole[j - 1];
        --j;
    }

    pole[j] = prvek;
}
```



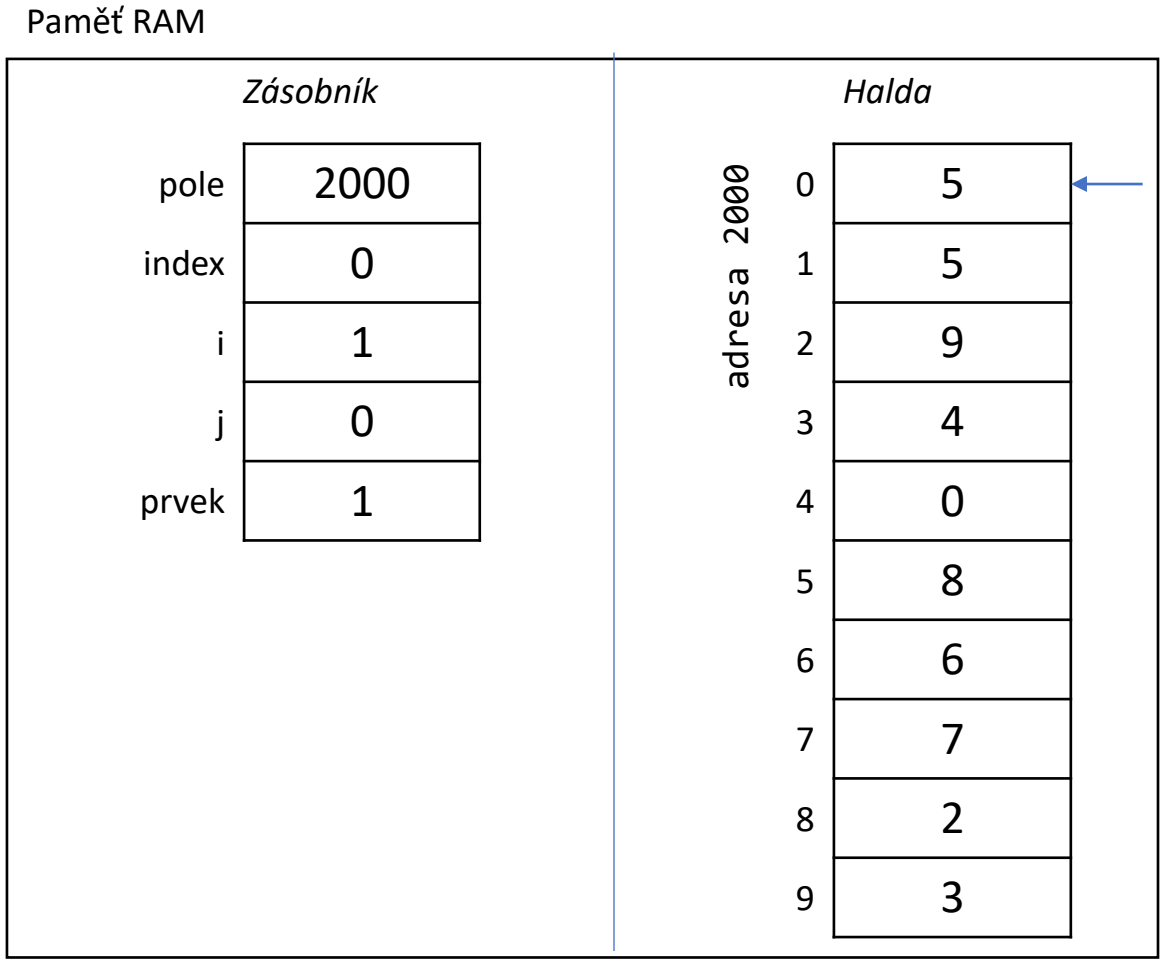
Insertion Sort

```
int[] pole = new int[]
{ 5, 9, 1, 4, 0, 8, 6, 7, 2, 3 };

int index = 0;

for (int i = index; i < pole.Length - 1; i++)
{
    int j = i + 1;
    int prvek = pole[j];

    while (j > 0 && prvek < pole[j - 1])
    {
        pole[j] = pole[j - 1];
        --j;
    }

    pole[j] = prvek;
}
```



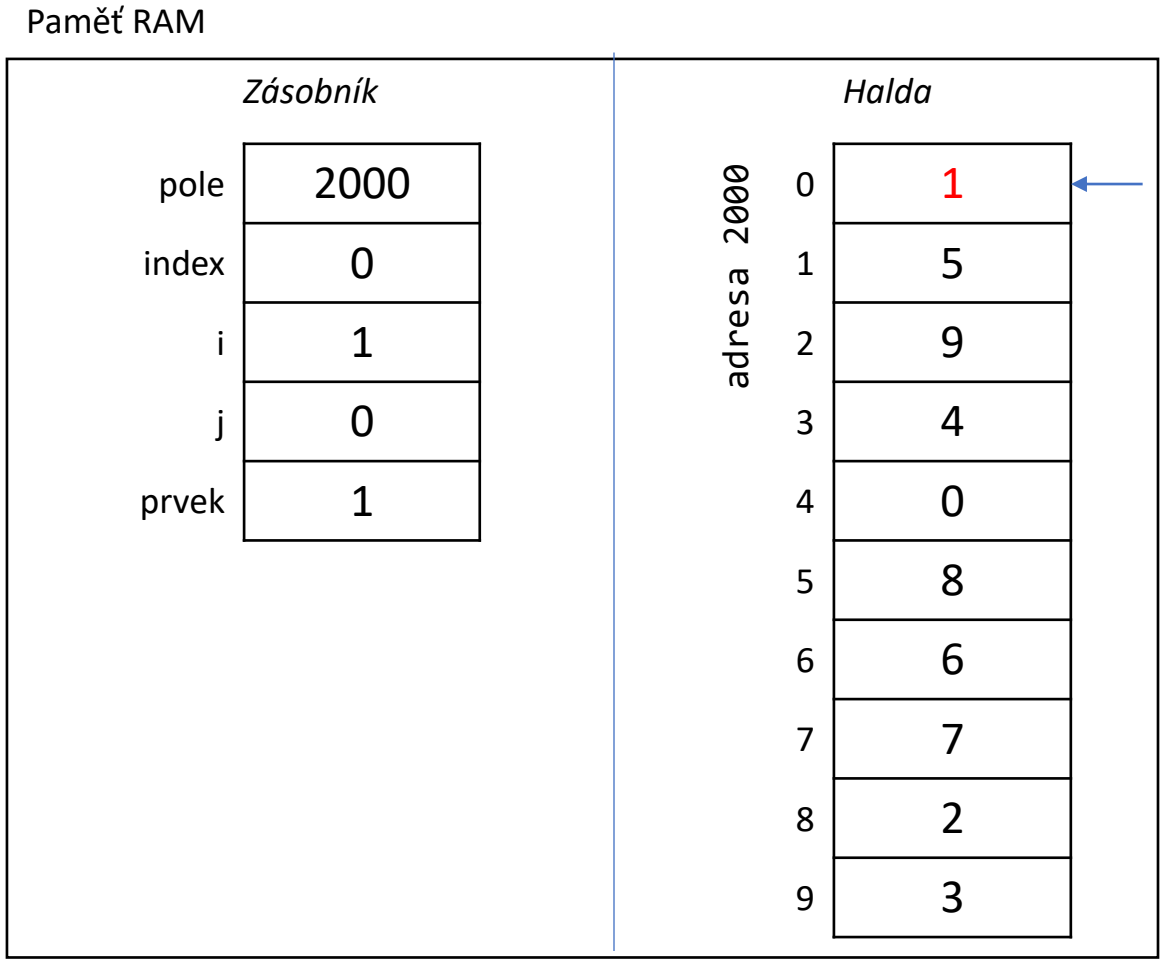
Insertion Sort

```
int[] pole = new int[]
{ 5, 9, 1, 4, 0, 8, 6, 7, 2, 3 };

int index = 0;

for (int i = index; i < pole.Length - 1; i++)
{
    int j = i + 1;
    int prvek = pole[j];
          0
    while (j > 0 && prvek < pole[j - 1])
    {
        pole[j] = pole[j - 1];
        --j;
    }

    pole[j] = prvek;
}
```



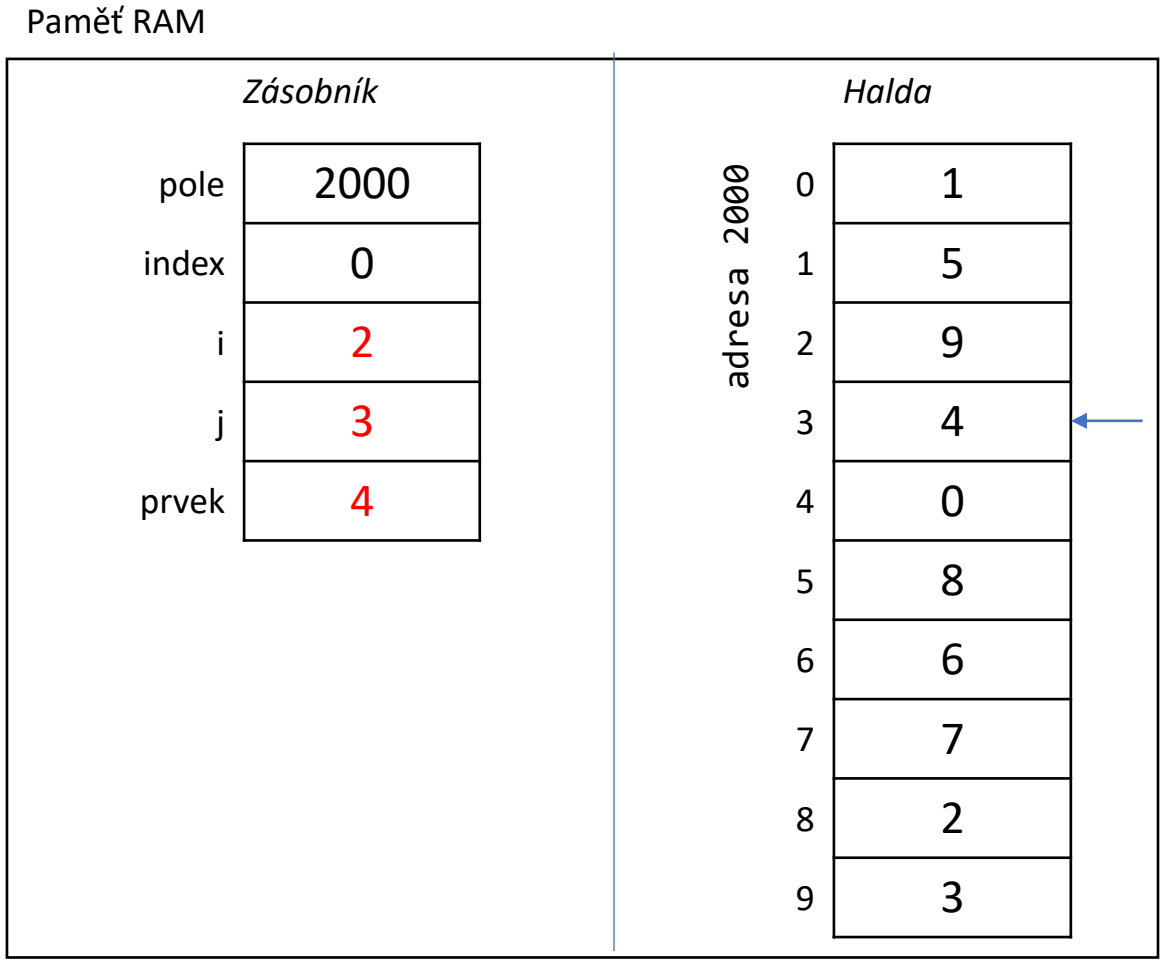
Insertion Sort

```
int[] pole = new int[]
{ 5, 9, 1, 4, 0, 8, 6, 7, 2, 3 };

int index = 0;

for (int i = index; i < pole.Length - 1; i++)
{
    int j = i + 1;
    int prvek = pole[j];

    while (j > 0 && prvek < pole[j - 1])
    {
        pole[j] = pole[j - 1];
        --j;
    }

    pole[j] = prvek;
}
```



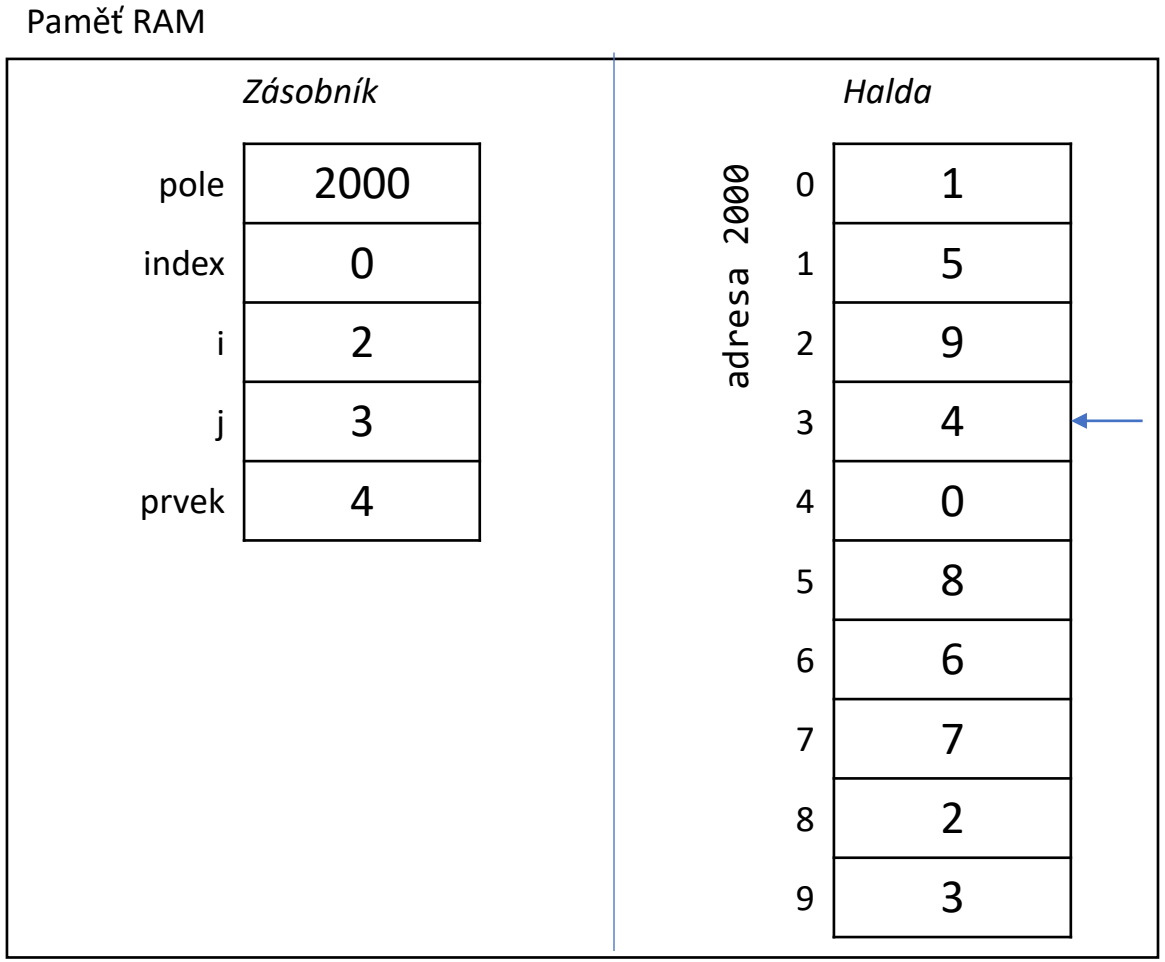
Insertion Sort

```
int[] pole = new int[]
{ 5, 9, 1, 4, 0, 8, 6, 7, 2, 3 };

int index = 0;

for (int i = index; i < pole.Length - 1; i++)
{
    int j = i + 1;
    int prvek = pole[j];

    while (j > 0 && prvek < pole[j - 1])
    {
        pole[j] = pole[j - 1];
        --j;
    }

    pole[j] = prvek;
}
```



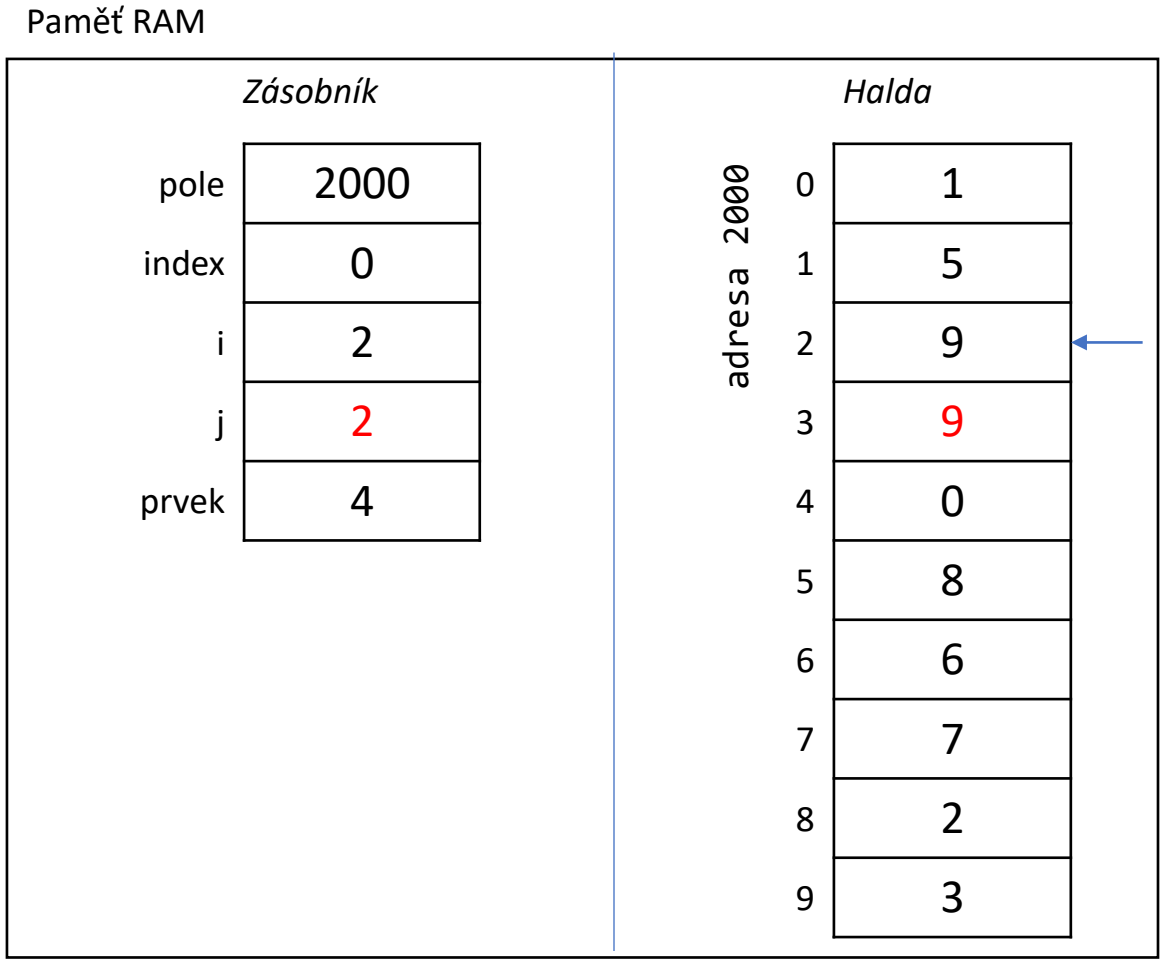
Insertion Sort

```
int[] pole = new int[]
{ 5, 9, 1, 4, 0, 8, 6, 7, 2, 3 };

int index = 0;

for (int i = index; i < pole.Length - 1; i++)
{
    int j = i + 1;
    int prvek = pole[j];
                        4       9
    while (j > 0 && prvek < pole[j - 1])
    {
        pole[j] = pole[j - 1];
        --j;
    }

    pole[j] = prvek;
}
```



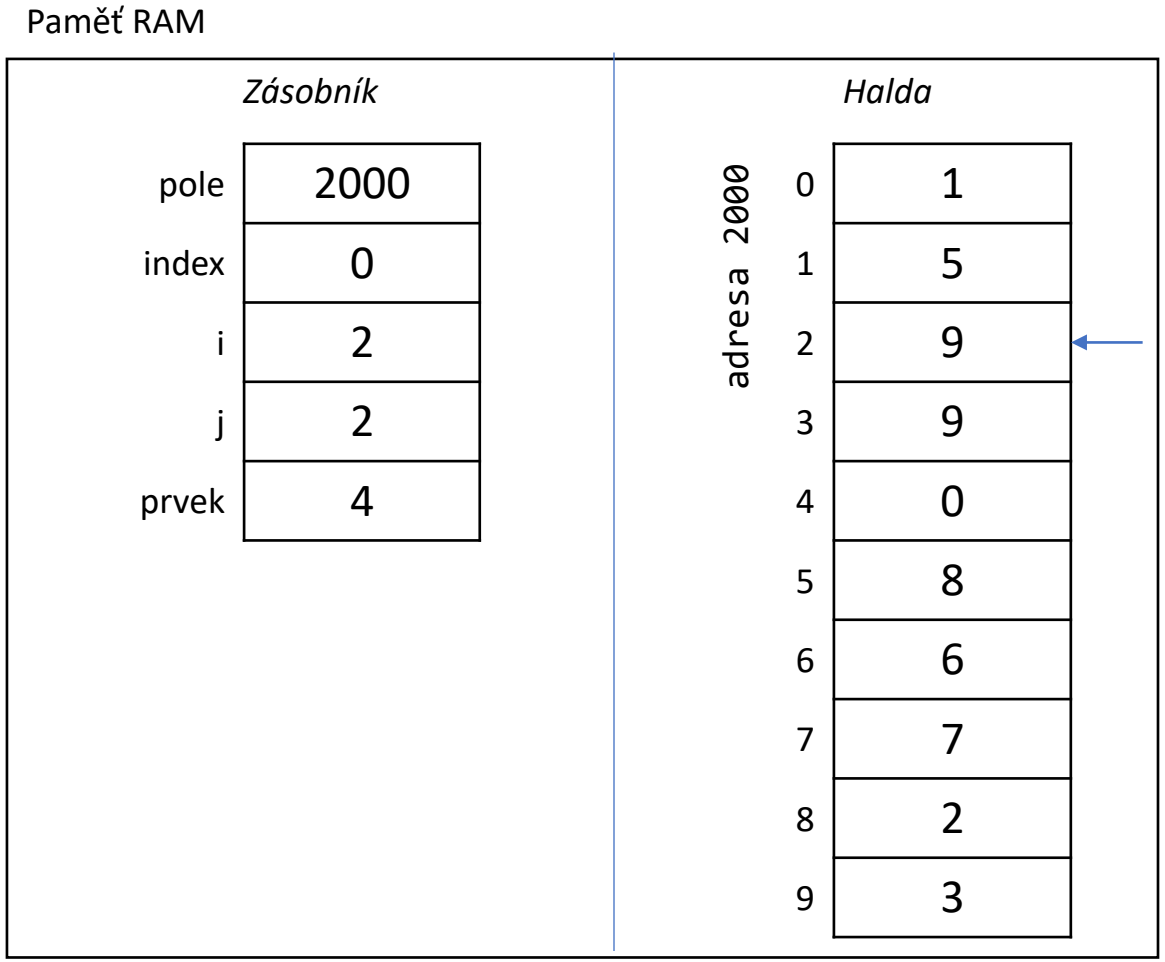
Insertion Sort

```
int[] pole = new int[]
{ 5, 9, 1, 4, 0, 8, 6, 7, 2, 3 };

int index = 0;

for (int i = index; i < pole.Length - 1; i++)
{
    int j = i + 1;
    int prvek = pole[j];

    while (j > 0 && prvek < pole[j - 1])
    {
        pole[j] = pole[j - 1];
        --j;
    }

    pole[j] = prvek;
}
```



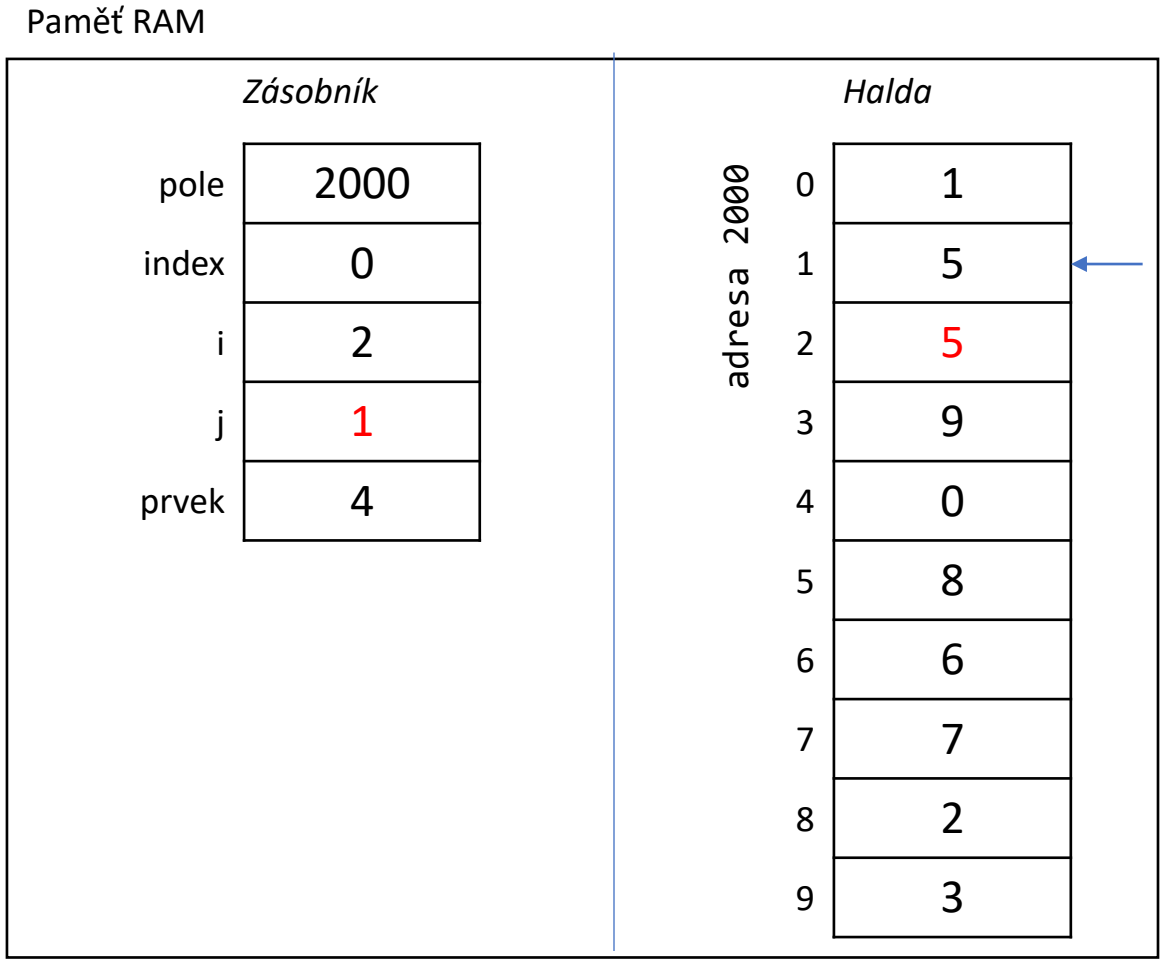
Insertion Sort

```
int[] pole = new int[]
{ 5, 9, 1, 4, 0, 8, 6, 7, 2, 3 };

int index = 0;

for (int i = index; i < pole.Length - 1; i++)
{
    int j = i + 1;
    int prvek = pole[j];
                        4        5
    while (j > 0 && prvek < pole[j - 1])
    {
        pole[j] = pole[j - 1];
        --j;
    }

    pole[j] = prvek;
}
```



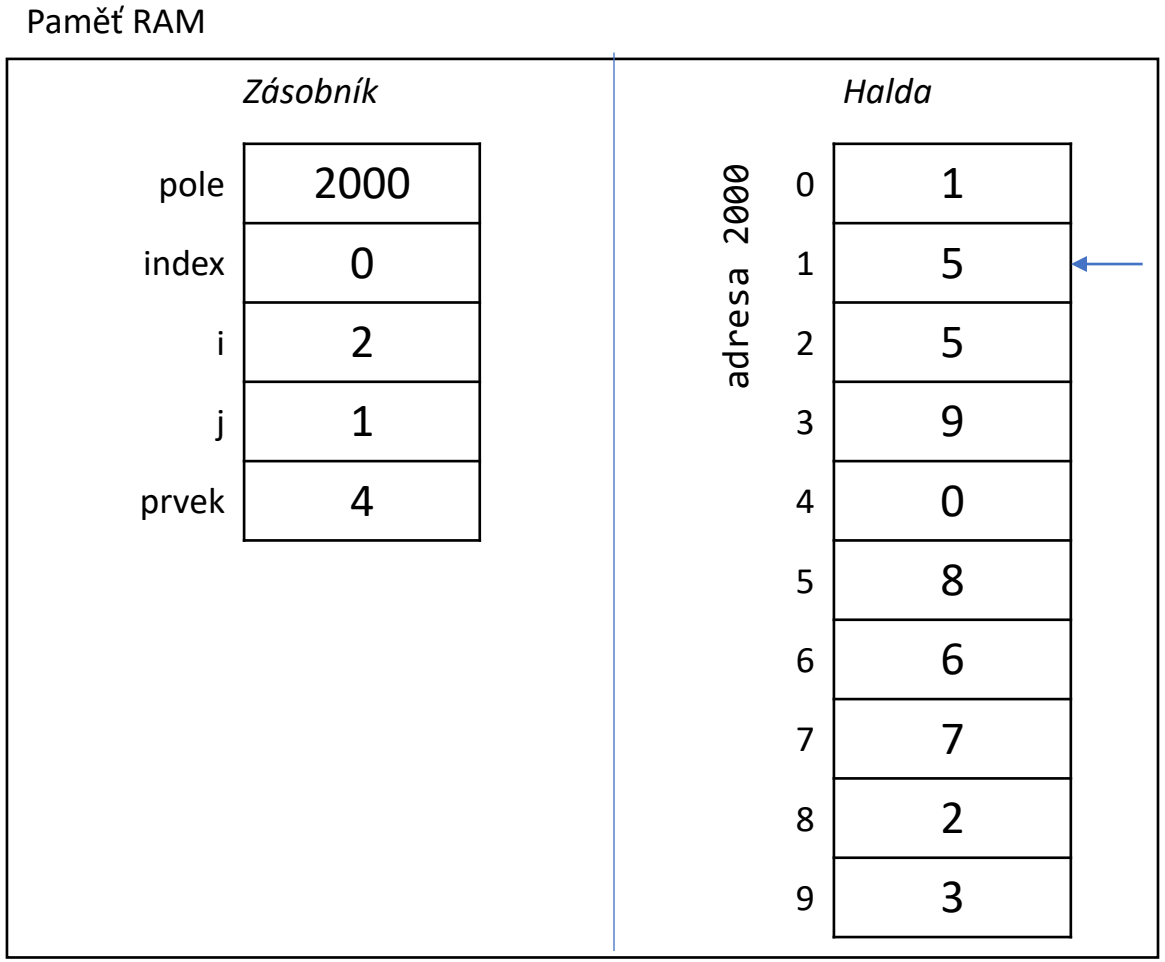
Insertion Sort

```
int[] pole = new int[]
{ 5, 9, 1, 4, 0, 8, 6, 7, 2, 3 };

int index = 0;

for (int i = index; i < pole.Length - 1; i++)
{
    int j = i + 1;
    int prvek = pole[j];

    while (j > 0 && prvek < pole[j - 1])
    {
        pole[j] = pole[j - 1];
        --j;
    }

    pole[j] = prvek;
}
```



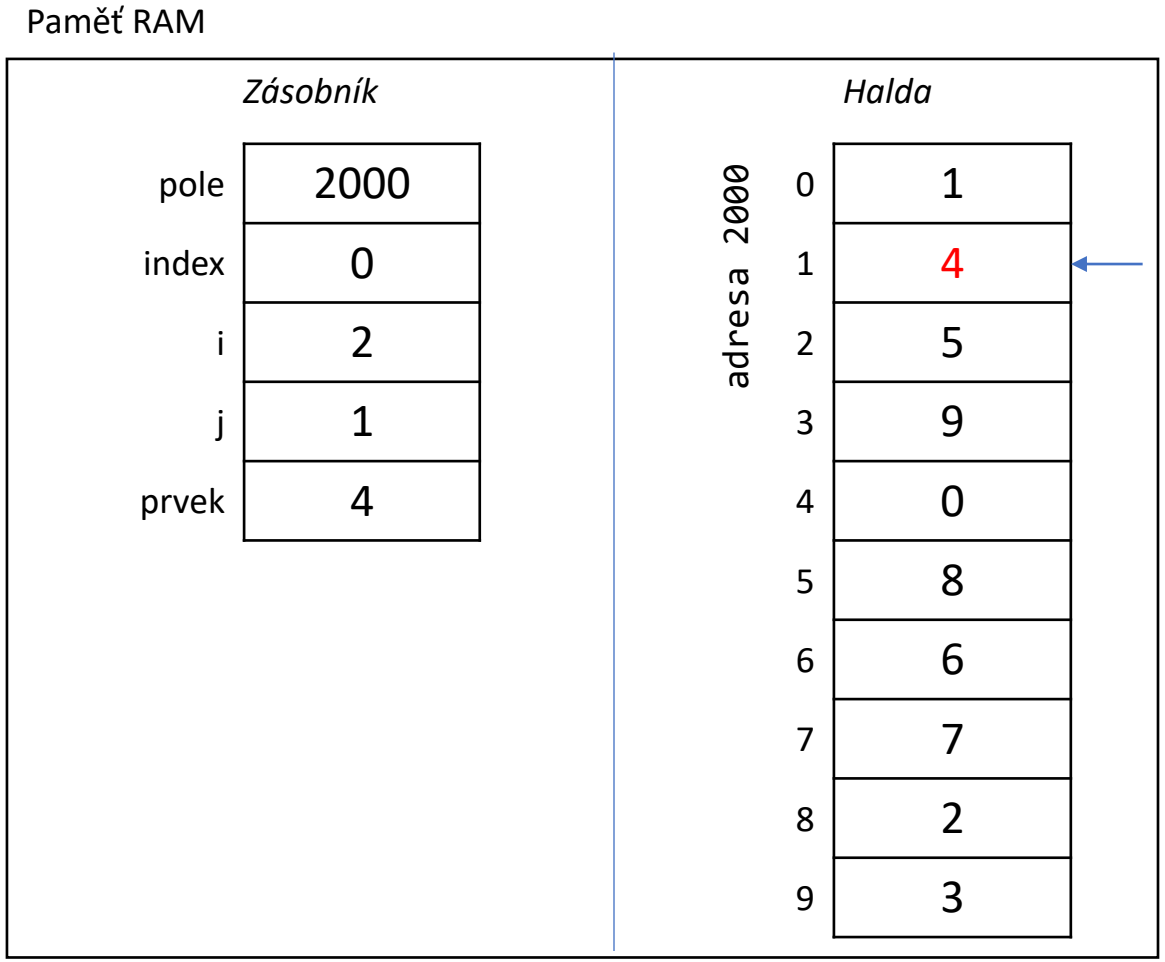
Insertion Sort

```
int[] pole = new int[]
{ 5, 9, 1, 4, 0, 8, 6, 7, 2, 3 };

int index = 0;

for (int i = index; i < pole.Length - 1; i++)
{
    int j = i + 1;
    int prvek = pole[j];
                        4        1
    while (j > 0 && prvek < pole[j - 1])
    {
        pole[j] = pole[j - 1];
        --j;
    }

    pole[j] = prvek;
}
```



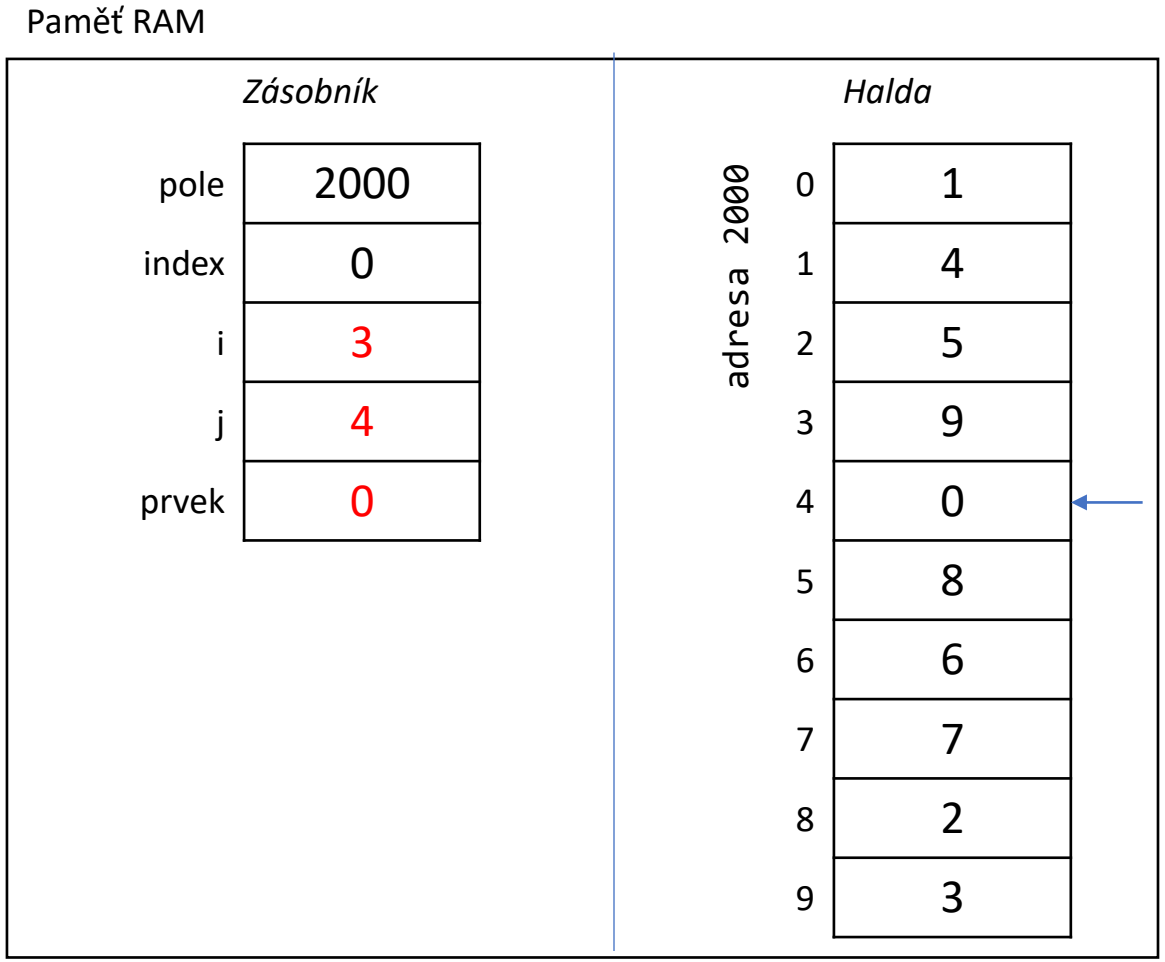
Insertion Sort

```
int[] pole = new int[]
{ 5, 9, 1, 4, 0, 8, 6, 7, 2, 3 };

int index = 0;

for (int i = index; i < pole.Length - 1; i++)
{
    int j = i + 1;
    int prvek = pole[j];

    while (j > 0 && prvek < pole[j - 1])
    {
        pole[j] = pole[j - 1];
        --j;
    }

    pole[j] = prvek;
}
```



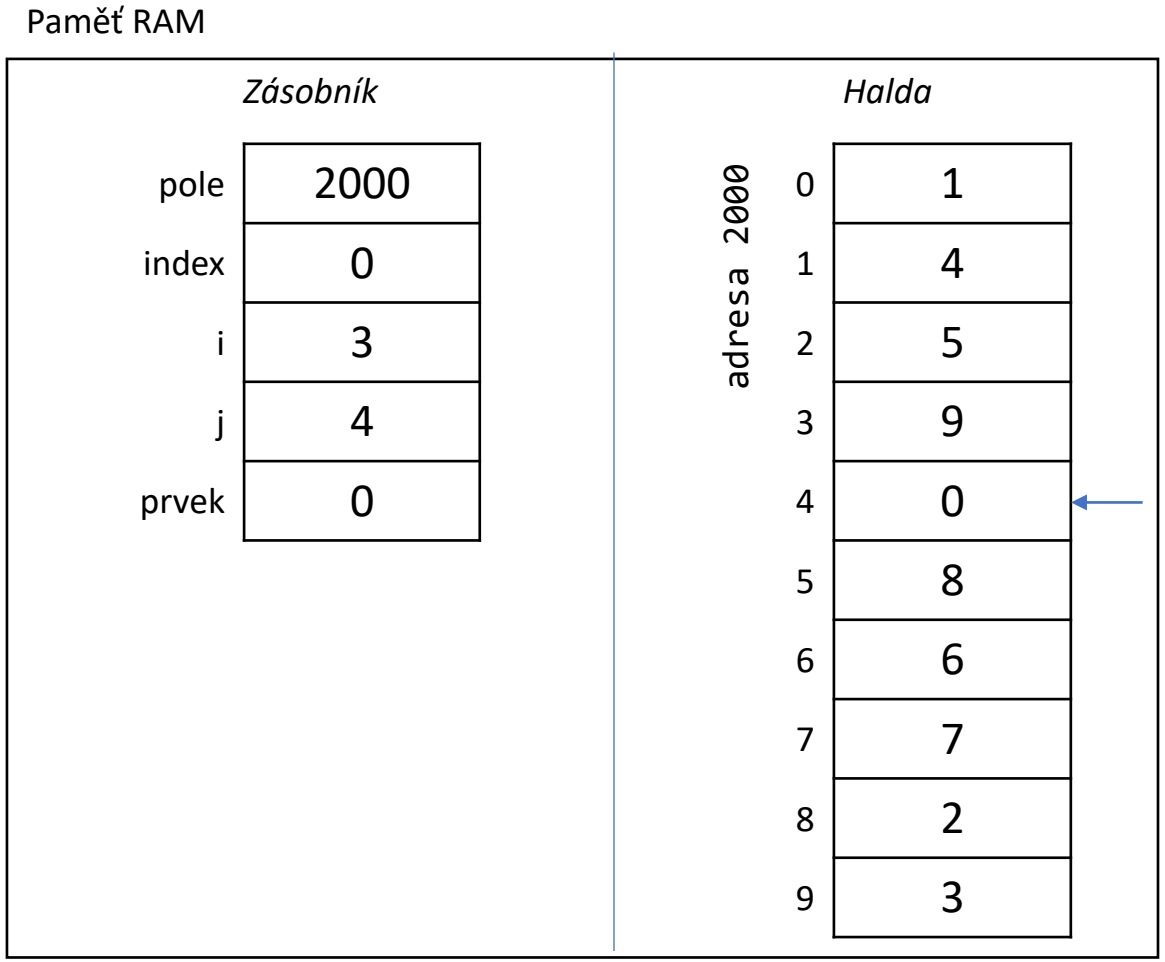
Insertion Sort

```
int[] pole = new int[]
{ 5, 9, 1, 4, 0, 8, 6, 7, 2, 3 };

int index = 0;

for (int i = index; i < pole.Length - 1; i++)
{
    int j = i + 1;
    int prvek = pole[j];

    while (j > 0 && prvek < pole[j - 1])
    {
        pole[j] = pole[j - 1];
        --j;
    }

    pole[j] = prvek;
}
```



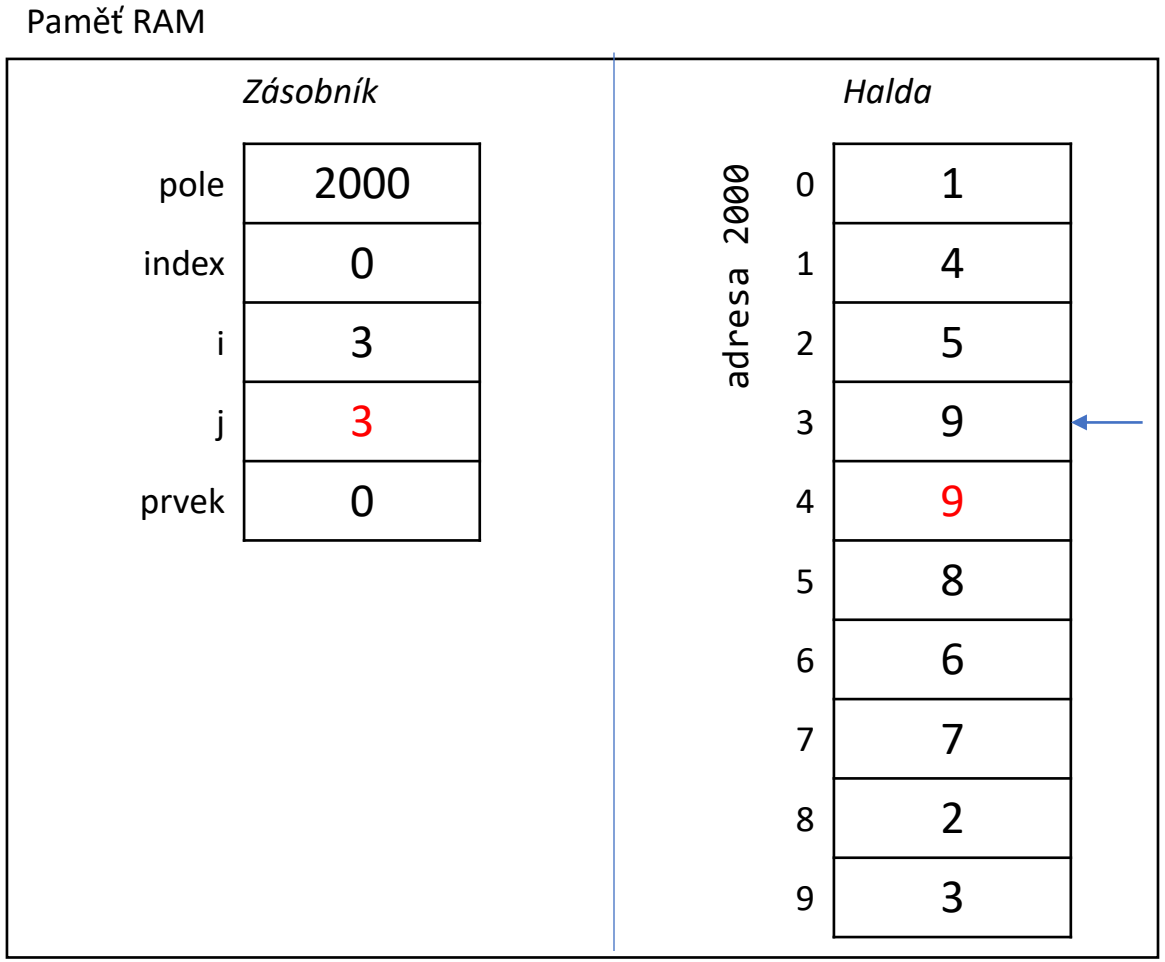
Insertion Sort

```
int[] pole = new int[]
{ 5, 9, 1, 4, 0, 8, 6, 7, 2, 3 };

int index = 0;

for (int i = index; i < pole.Length - 1; i++)
{
    int j = i + 1;
    int prvek = pole[j];
                        0       9
    while (j > 0 && prvek < pole[j - 1])
    {
        pole[j] = pole[j - 1];
        --j;
    }

    pole[j] = prvek;
}
```



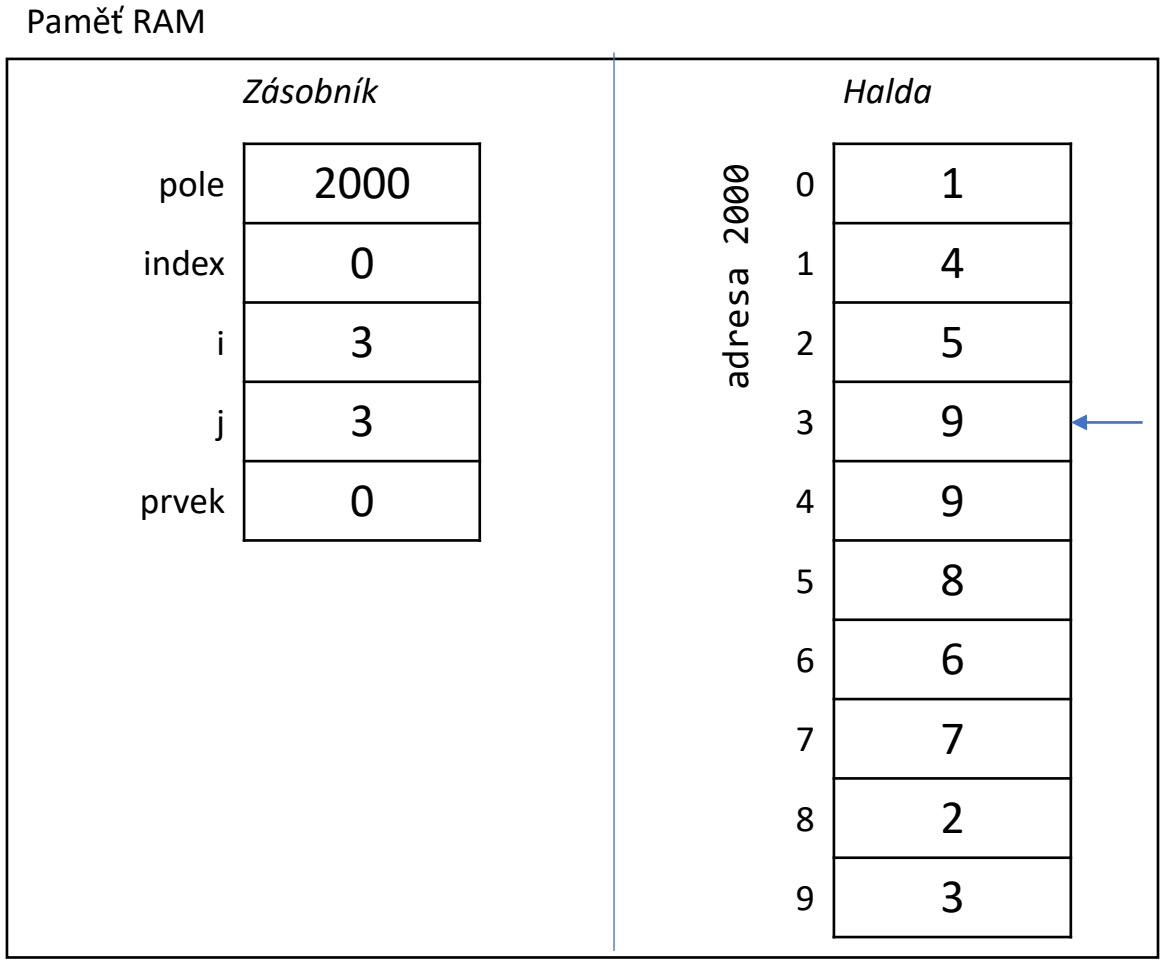
Insertion Sort

```
int[] pole = new int[]
{ 5, 9, 1, 4, 0, 8, 6, 7, 2, 3 };

int index = 0;

for (int i = index; i < pole.Length - 1; i++)
{
    int j = i + 1;
    int prvek = pole[j];

    while (j > 0 && prvek < pole[j - 1])
    {
        pole[j] = pole[j - 1];
        --j;
    }

    pole[j] = prvek;
}
```



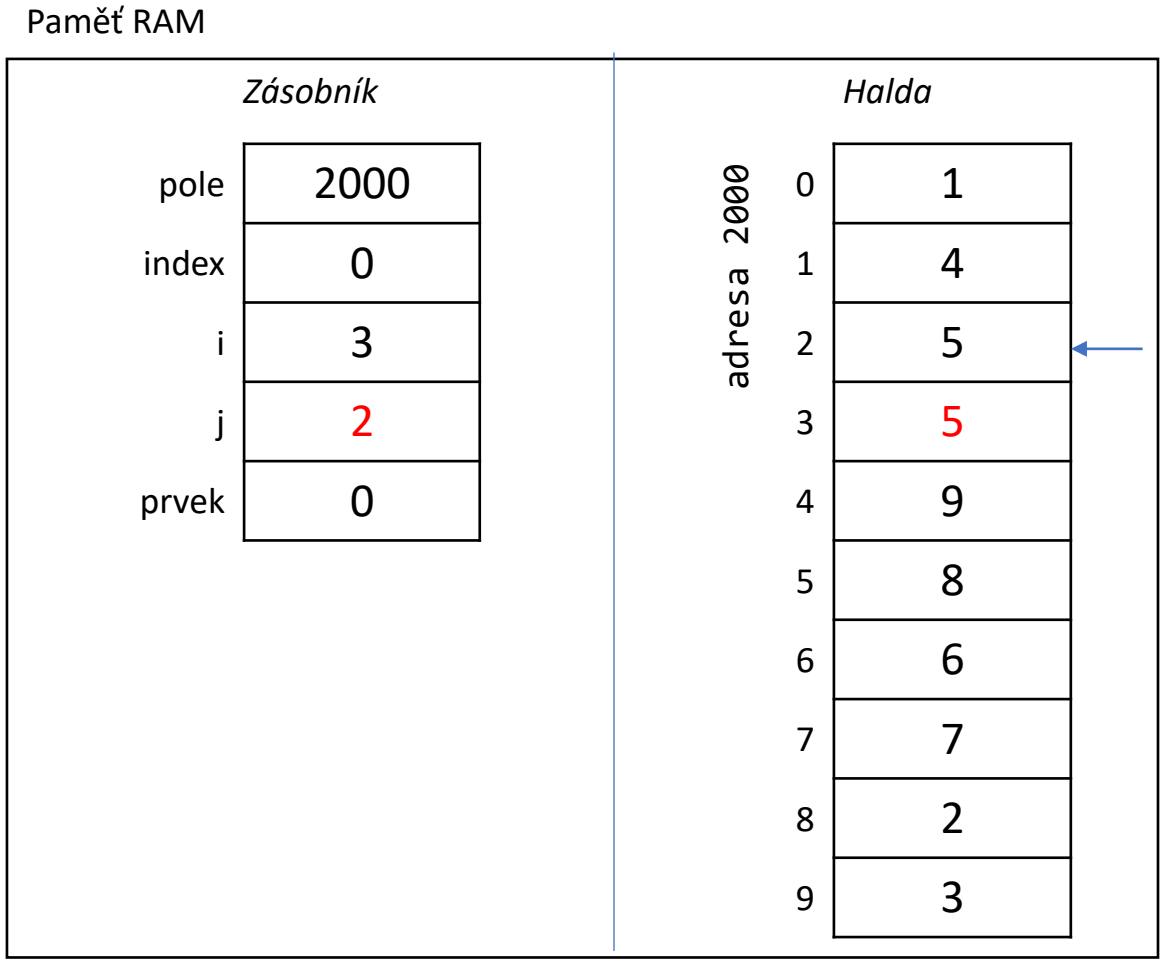
Insertion Sort

```
int[] pole = new int[]
{ 5, 9, 1, 4, 0, 8, 6, 7, 2, 3 };

int index = 0;

for (int i = index; i < pole.Length - 1; i++)
{
    int j = i + 1;
    int prvek = pole[j];
                        0       5
    while (j > 0 && prvek < pole[j - 1])
    {
        pole[j] = pole[j - 1];
        --j;
    }

    pole[j] = prvek;
}
```



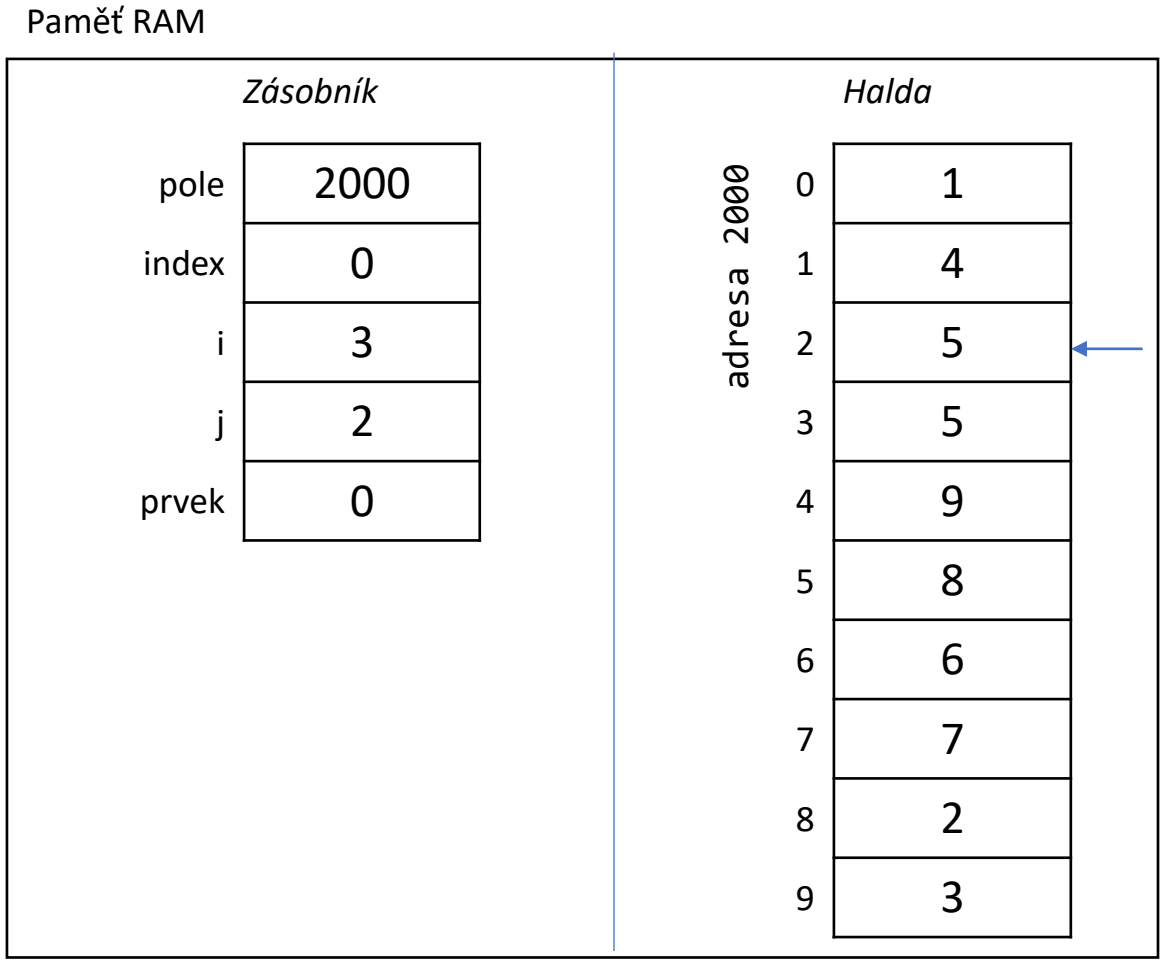
Insertion Sort

```
int[] pole = new int[]
{ 5, 9, 1, 4, 0, 8, 6, 7, 2, 3 };

int index = 0;

for (int i = index; i < pole.Length - 1; i++)
{
    int j = i + 1;
    int prvek = pole[j];

    while (j > 0 && prvek < pole[j - 1])
    {
        pole[j] = pole[j - 1];
        --j;
    }

    pole[j] = prvek;
}
```



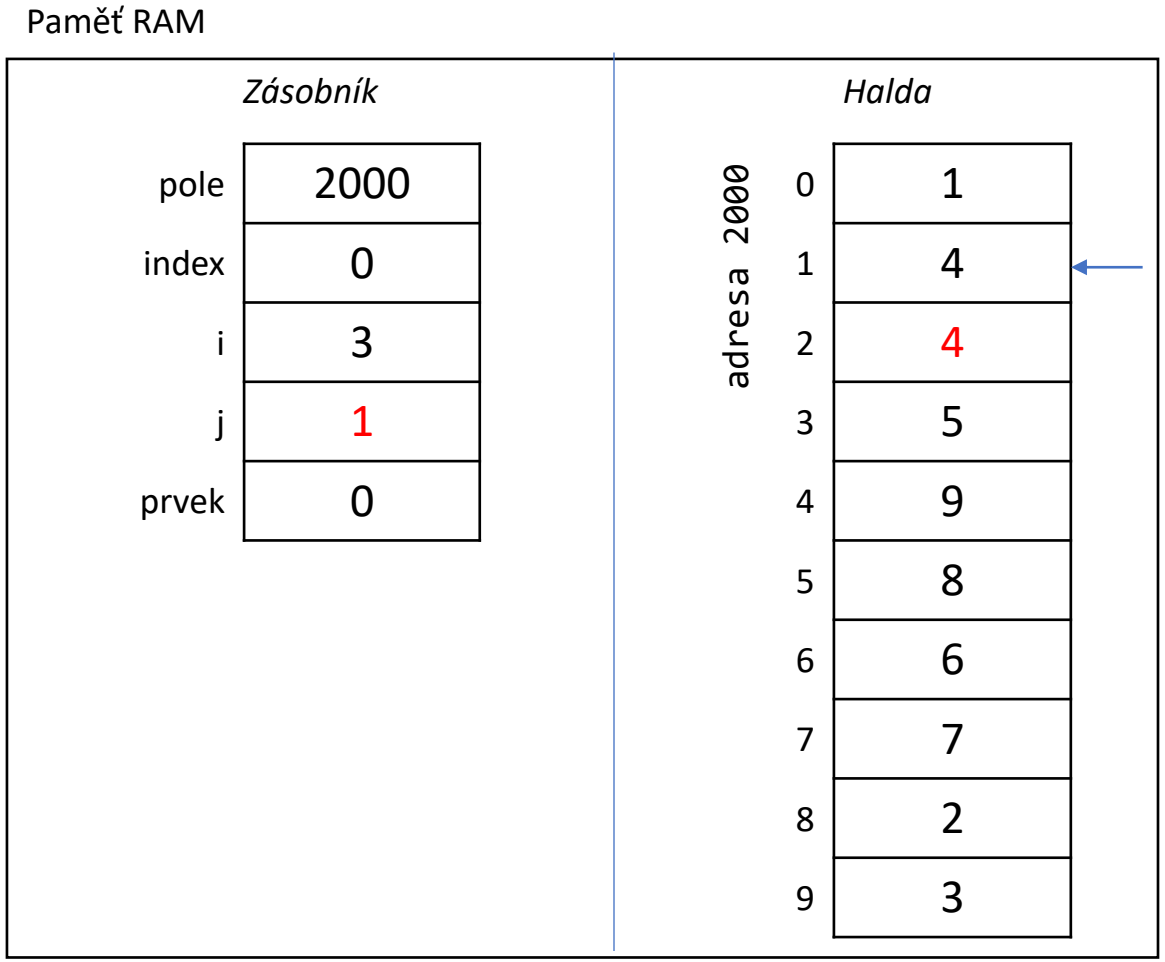
Insertion Sort

```
int[] pole = new int[]
{ 5, 9, 1, 4, 0, 8, 6, 7, 2, 3 };

int index = 0;

for (int i = index; i < pole.Length - 1; i++)
{
    int j = i + 1;
    int prvek = pole[j];
                        0       4
    while (j > 0 && prvek < pole[j - 1])
    {
        pole[j] = pole[j - 1];
        --j;
    }

    pole[j] = prvek;
}
```



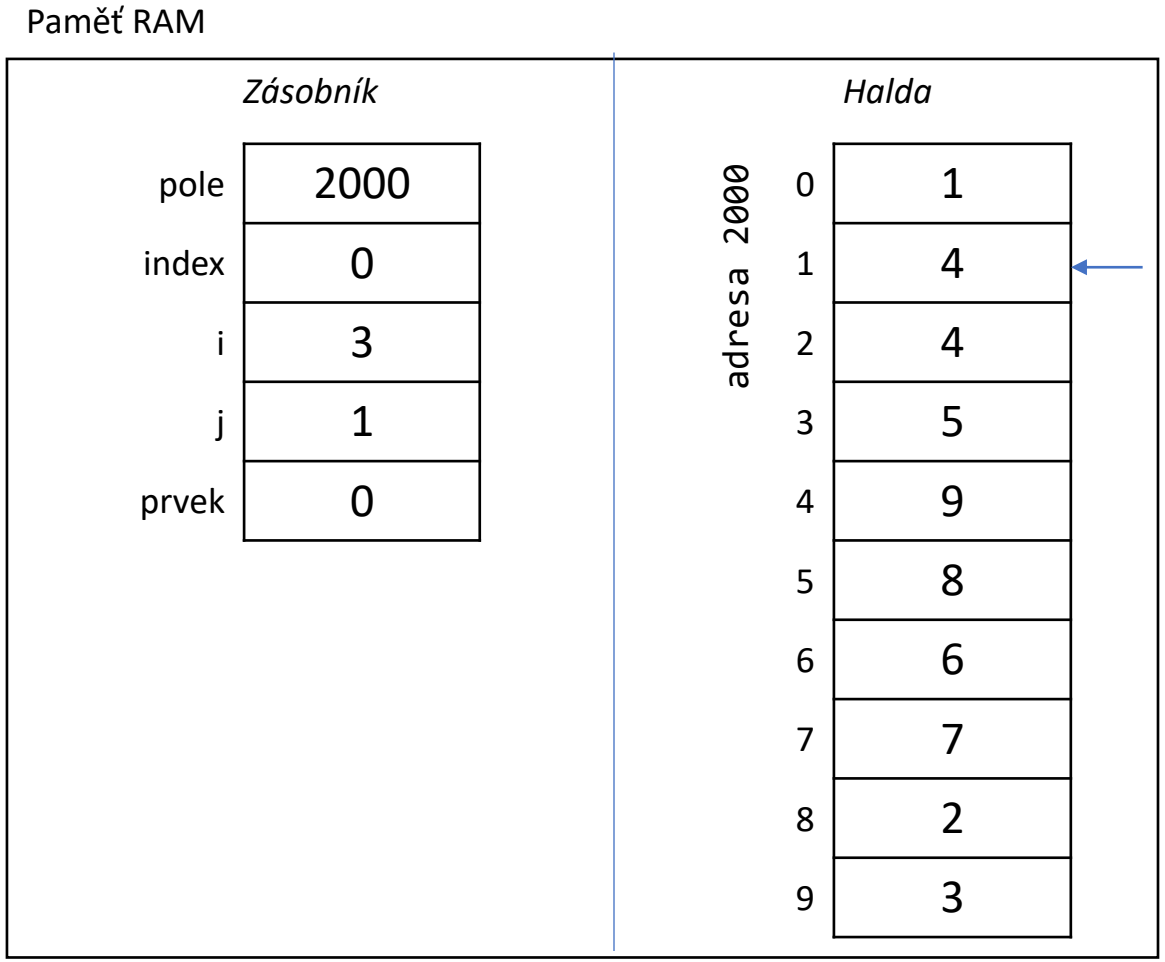
Insertion Sort

```
int[] pole = new int[]
{ 5, 9, 1, 4, 0, 8, 6, 7, 2, 3 };

int index = 0;

for (int i = index; i < pole.Length - 1; i++)
{
    int j = i + 1;
    int prvek = pole[j];

    while (j > 0 && prvek < pole[j - 1])
    {
        pole[j] = pole[j - 1];
        --j;
    }

    pole[j] = prvek;
}
```



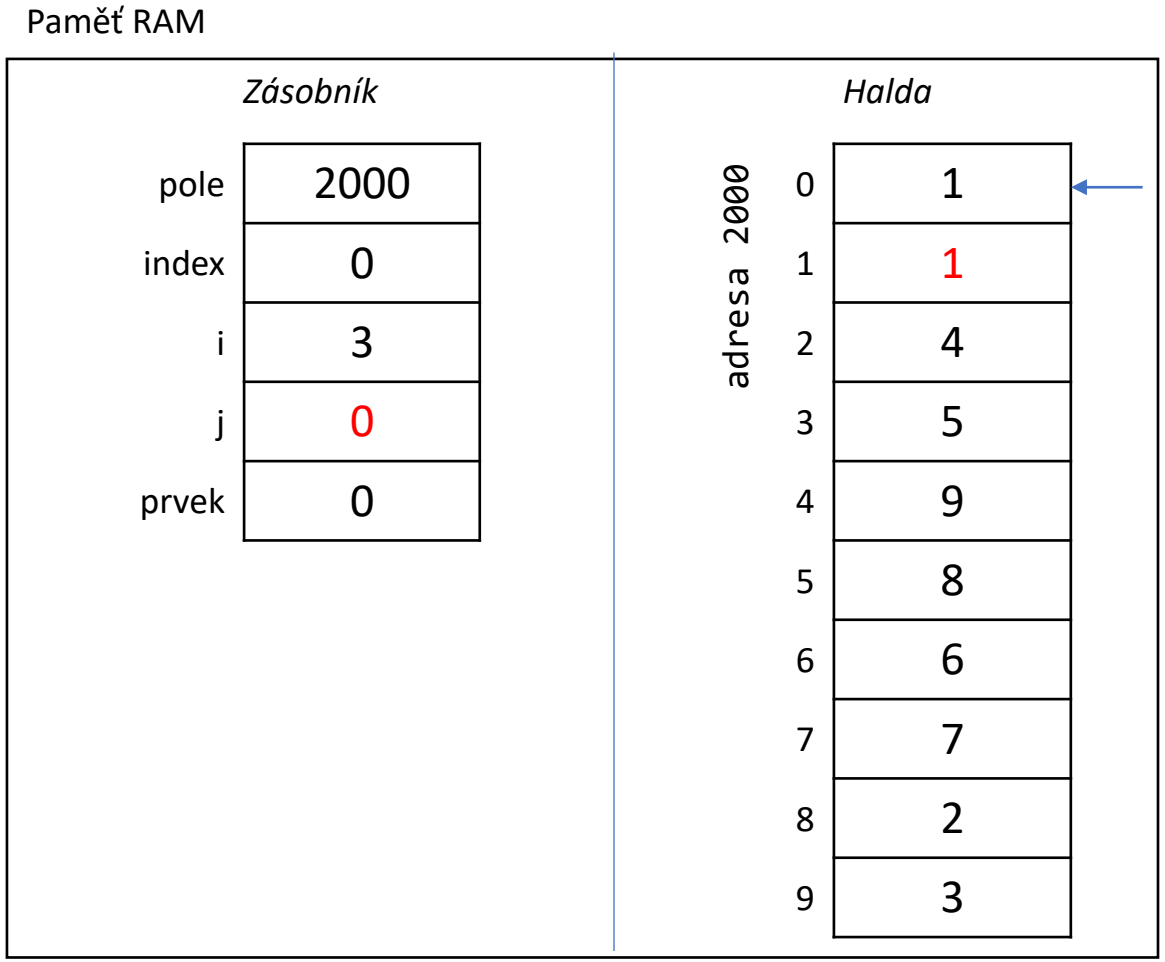
Insertion Sort

```
int[] pole = new int[]
{ 5, 9, 1, 4, 0, 8, 6, 7, 2, 3 };

int index = 0;

for (int i = index; i < pole.Length - 1; i++)
{
    int j = i + 1;
    int prvek = pole[j];
                        0       1
    while (j > 0 && prvek < pole[j - 1])
    {
        pole[j] = pole[j - 1];
        --j;
    }

    pole[j] = prvek;
}
```



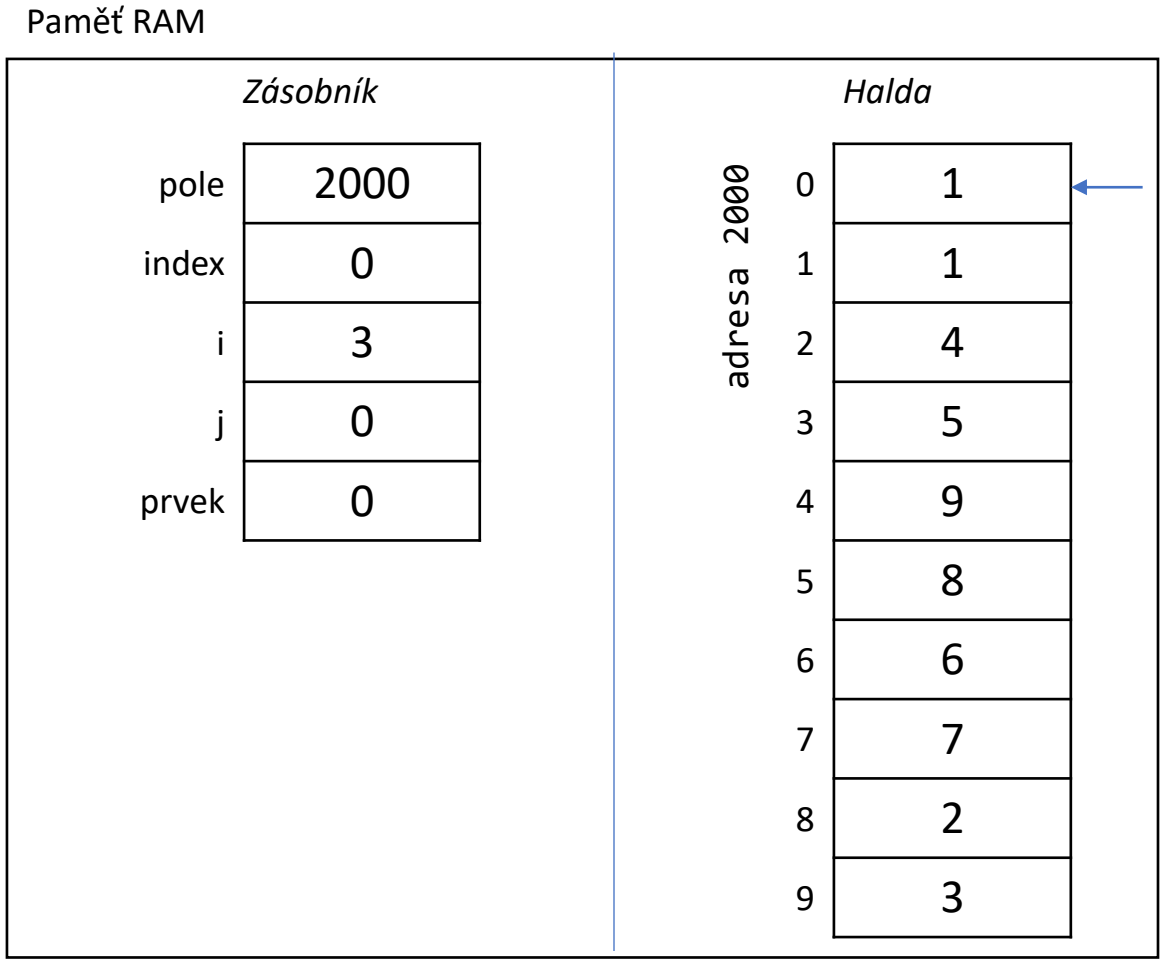
Insertion Sort

```
int[] pole = new int[]
{ 5, 9, 1, 4, 0, 8, 6, 7, 2, 3 };

int index = 0;

for (int i = index; i < pole.Length - 1; i++)
{
    int j = i + 1;
    int prvek = pole[j];

    while (j > 0 && prvek < pole[j - 1])
    {
        pole[j] = pole[j - 1];
        --j;
    }

    pole[j] = prvek;
}
```



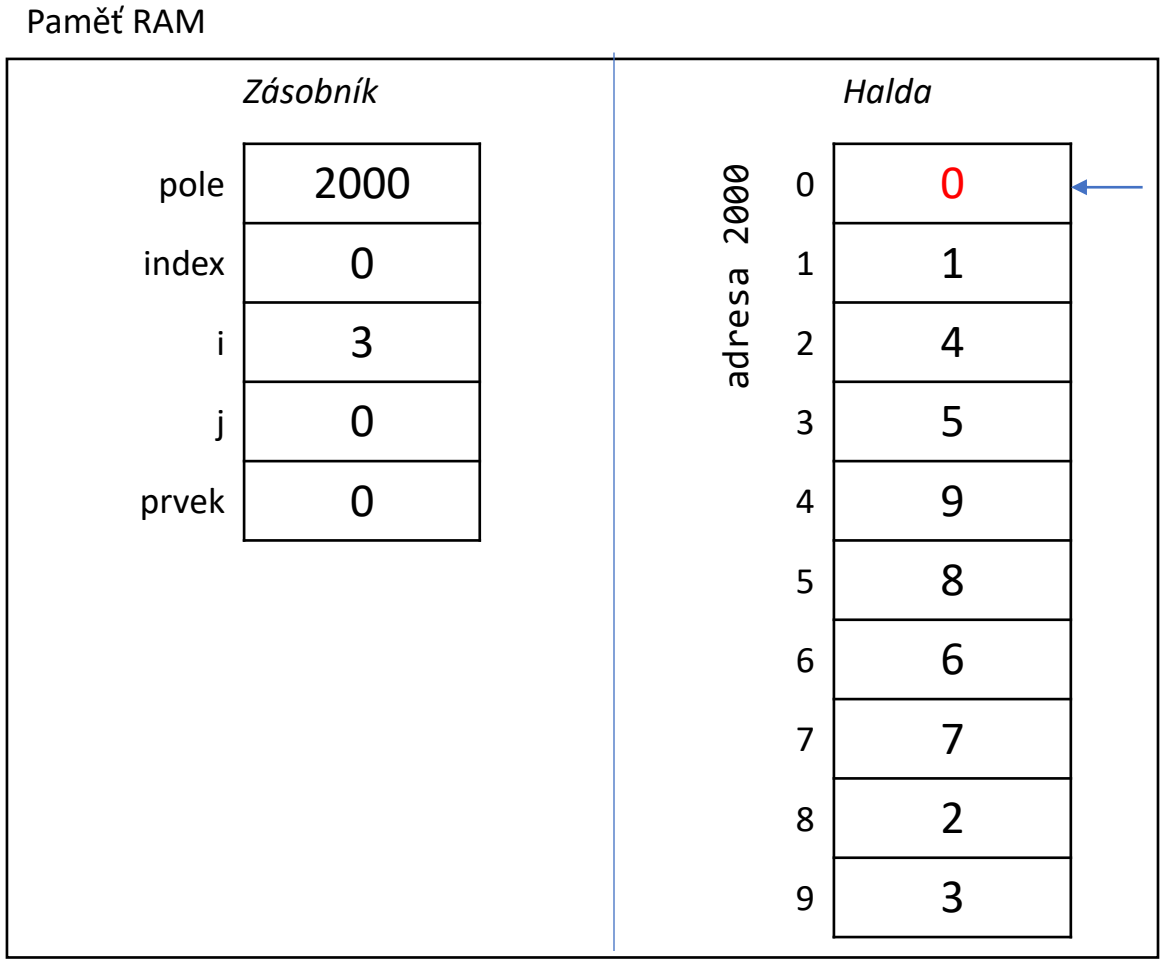
Insertion Sort

```
int[] pole = new int[]
{ 5, 9, 1, 4, 0, 8, 6, 7, 2, 3 };

int index = 0;

for (int i = index; i < pole.Length - 1; i++)
{
    int j = i + 1;
    int prvek = pole[j];
            0
    while (j > 0 && prvek < pole[j - 1])
    {
        pole[j] = pole[j - 1];
        --j;
    }

    pole[j] = prvek;
}
```



Insertion Sort

```
int[] pole = new int[]
{ 5, 9, 1, 4, 0, 8, 6, 7, 2, 3 };

int index = 0;

for (int i = index; i < pole.Length - 1; i++)
{
    int j = i + 1;
    int prvek = pole[j];

    while (j > 0 && prvek < pole[j - 1])
    {
        pole[j] = pole[j - 1];
        --j;
    }

    pole[j] = prvek;
}
```



Insertion Sort – ukončení algoritmu

- Vzhledem k množství iterací nebude procházet další kroky.
- Algoritmus se zastaví až budou provedeny všechny iterace.

Použité zdroje

[1] Insertion sort. *Algoritmus* [online]. Copyright © 2015 [cit. 26.02.2021]. Dostupné z: https://www.algoritmy.net/article/8/Insertion-sort

[2] 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

[3] for statement - C# reference | Microsoft Docs. [online]. Copyright © Microsoft 2021 [cit. 26.02.2021]. Dostupné z: https://docs.microsoft.com/en-us/dotnet/csharp/language-reference/keywords/for

[4] Insertion Sort vs Bubble Sort + Some analysis - YouTube. *YouTube* [online]. Copyright © 2021 Google LLC [cit. 26.02.2021]. Dostupné z: https://www.youtube.com/watch?v=TZRWRjq2CAg

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