## Tablica z zadania numer 1

```
tatic void Main(string[] a̞r̞gs)
  double quickSortValue = 0;
  Stopwatch stopwatch = new Stopwatch();
  int[] arr = { 15, 12, 11, 6, 5, 32, 0, 22 };
  stopwatch.Start();
  //bubblesort
  BubbleSort(arr);
  stopwatch.Stop();
  bubbleSortValue = stopwatch.ElapsedMilliseconds;
  stopwatch.Reset();
  stopwatch.Start();
  //Quicksort
  QuickSort(arr, 0, arr.Length - 1);
  stopwatch.Stop();
  quickSortValue = stopwatch.ElapsedMilliseconds;
  Console.WriteLine($"Bubblesort: {bubbleSortValue}");
  Console.WriteLine($"QuickSort: {quickSortValue}");
```

```
Fukcja Main.
```

Aby pokazać czas w jakim odbywają się sortowania zastosowałem metodę StopWatch. Dzięki niej program wyświetlił jak długo wykonuje się dana funkcja.

```
public static void BubbleSort(int[] arr)
{
    int temp = 0;
    for (int write = 0; write < arr.Length; write++)
    {
        for (int sort = 0; sort < arr.Length - 1; sort++)
        {
            if (arr[sort] > arr[sort + 1])
            {
                temp = arr[sort + 1];
                 arr[sort + 1] = arr[sort];
                 arr[sort] = temp;
            }
        }
    }
}

references
public static void QuickSort(int[] array, int left, int right)
{
    var i = left;
    var j = right;
    var pivot = array[(left + right) / 2];
    while (i < j)
    {
        while (array[i] < pivot) i++;
        while (array[j] > pivot) j--;
        if (i < j)
        {
            var tmp = array[i];
                 array[i++] = array[j];
                 array[i--] = tmp;
        }
        if (left < j) QuickSort(array, left, j);
        if (i < right) QuickSort(array, i, right);
}</pre>
```

Funkcje sortujące.	
BubleSort	
QuickSort	

Microsoft Visual Studio Debug Console

Bubblesort: 0

QuickSort: 0

C:\Users\Bartek\source\repos\SortBubleQuick\Sotth code 0.

To automatically close the console when debuggle when debugging stops.

Press any key to close this window . . .

Wynik programu. Niestety tablica jest zbyt mała aby program nawet w milisekundach pokazał różnicę

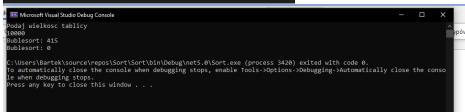
## **Tablica Random**

```
using System;
using System.Diagnostics;
     O references
class Program
          Oreferences
static void Main(string[] args)
               double bubbleSortValue = 0;
               double quickSortValue = 0;
Stopwatch stopwatch = new Stopwatch();
               Random random = new Random();
Console.WriteLine("Podaj wielkosc tablicy");
               int n =Convert.ToInt32(Console.ReadLine());
               int[] arr = new int[n];
               for (int i = 0; i < n; i++)
                    arr[i] = random.Next(0, 1000000);
               stopwatch.Start();
               BubleSort(arr);
               bubbleSortValue = stopwatch.ElapsedMilliseconds;
               stopwatch.Reset();
               stopwatch.Start();
               QuickSort(arr, 0, arr.Length - 1);
               stopwatch.Stop();
quickSortValue = stopwatch.ElapsedMilliseconds;
               Console.WriteLine($"Bublesort: {bubbleSortValue}");
Console.WriteLine($"Bublesort: {quickSortValue}");
```

Fukcja Main.

Aby pokazać dokładnie czy jest różnica między sortowaniami zastosowałem metodę Random.Next. Program sam generuje losowe liczby z przedziału który wyznaczy użytkownik. Maxymalny rozmiar tablicy to 1.000.000

Funkcje sortujące.	
BubleSort	
	$\neg$
QuickSort	



Wynik programu. Tutaj już wyraźnie widzimy różnicę między sortowaniami. Przykład jest dla tablicy która posiada 10.000 elementów.