Міністерство освіти і науки України

Центральноукраїнський національний технічний університет

**МЕХАНІКО-ТЕХНОЛОГІЧНИЙ ФАКУЛЬТЕТ**

Кафедра програмування та захисту інформації

**Звіт**

з виконаної лабораторної роботи № 5

дисципліни “ Системне програмування”

на тему

«**МЕТОДИ ТОЧНОГО ВИМІРУ МАЛИХ ЧАСОВИХ ІНТЕРВАЛІВ**»

Виконав :

студент академічної групи КІ-15

Аннаєв А. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Перевірив :

Викладач

Константинова K.В.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Кропивницкий- 2017

**ТЕМА**: МЕТОДИ ТОЧНОГО ВИМІРУ МАЛИХ ЧАСОВИХ ІНТЕРВАЛІВ

**МЕТА**: Вивчення існуючих методів точного виміру малих часових інтервалів

**ЗНАТИ**: Основи RAD (наприклад Builder або Delphi).

ЗАВДАННЯ

Використовуючи наявну електронну документацію, написати програму точного виміру часу сортування масиву цілих невід'ємних чисел.

Для спрощення лабораторної роботи використовувати вже існуючі вихідні коди алгоритмів, які наведені у 4тому електронному документі:

- Сортування вставками;

- Бульбашкове сортування;

- Сортування Шейкером;

- Сортування злиттям;

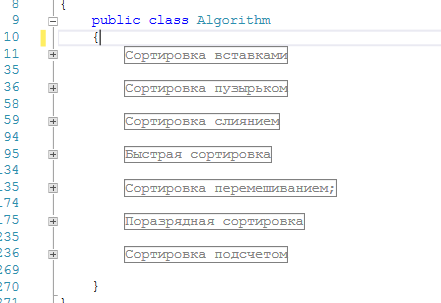
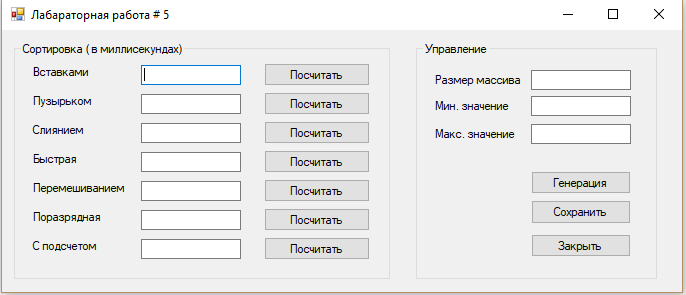
- Сортування двійковою купою;

- Швидке сортування;

- Сортування підрахунком;

- Цифрове сортування.

Результати вимірів алгоритмів сортування подати у порівняльній таблиці.

 public partial class Form1 : Form

{

public Form1()

{

InitializeComponent( );

algorithm = new Algorithm( );

}

private Algorithm algorithm;

private int []array;

private int max\_value;

private int min\_value;

private long start;

/// <summary>

///

/// </summary>

/// <param name="sender"></param>

/// <param name="e"></param>

private void btnInsert\_Click( object sender, EventArgs e )

{

if (array.Length != 0)

{

start = DateTime.Now.Ticks;

algorithm.InsertionSort(array);

txtInsert.Text = Convert.ToString

(TimeSpan.FromTicks(DateTime.Now.Ticks - start).TotalSeconds);

}

else

MessageBox.Show("С генерируйте массив" );

}

/// <summary>

/// Сортировка вставками

/// </summary>

/// <param name="sender"></param>

/// <param name="e"></param>

private void btnBubble\_Click( object sender, EventArgs e )

{

if (array.Length != 0)

{

start = DateTime.Now.Ticks;

algorithm.BubbleSort(array);

txtBubble.Text = Convert.ToString

(TimeSpan.FromTicks(DateTime.Now.Ticks - start).TotalSeconds);

}

else

MessageBox.Show("С генерируйте массив");

}

private void btnMerge\_Click( object sender, EventArgs e )

{

if (array.Length != 0)

{

start = DateTime.Now.Ticks;

algorithm.Merge\_Sort(array);

txtMerge.Text = Convert.ToString

(TimeSpan.FromTicks(DateTime.Now.Ticks - start).TotalSeconds);

}

else

MessageBox.Show("С генерируйте массив");

}

/// <summary>

///

/// </summary>

/// <param name="sender"></param>

/// <param name="e"></param>

private void btnQuick\_Click( object sender, EventArgs e )

{

if (array.Length != 0)

{

start = DateTime.Now.Ticks;

algorithm.quicksort(array, 0, array.Length-1);

txtQuick.Text = Convert.ToString

(TimeSpan.FromTicks(DateTime.Now.Ticks - start).TotalSeconds);

}

else

MessageBox.Show("С генерируйте массив");

}

/// <summary>

///

/// </summary>

/// <param name="sender"></param>

/// <param name="e"></param>

private void btnSheider\_Click( object sender, EventArgs e )

{

if (array.Length != 0)

{

start = DateTime.Now.Ticks;

algorithm.SheiderSort(array);

txtSheider.Text = Convert.ToString

(TimeSpan.FromTicks(DateTime.Now.Ticks - start).TotalSeconds);

}

else

MessageBox.Show("С генерируйте массив");

}

/// <summary>

///

/// </summary>

/// <param name="sender"></param>

/// <param name="e"></param>

private void btnRadix\_Click( object sender, EventArgs e )

{

if (array.Length != 0)

{

start = DateTime.Now.Ticks;

algorithm.RadixSort(array);

txtRadix.Text = Convert.ToString

(TimeSpan.FromTicks(DateTime.Now.Ticks - start).TotalSeconds);

}

else

MessageBox.Show("С генерируйте массив");

}

/// <summary>

///

/// </summary>

/// <param name="sender"></param>

/// <param name="e"></param>

private void btnCounter\_Click( object sender, EventArgs e )

{

if (array.Length != 0)

{

start = DateTime.Now.Ticks;

algorithm.countingSort(array, min\_value, max\_value);

txtCounter.Text = Convert.ToString

(TimeSpan.FromTicks(DateTime.Now.Ticks - start).TotalSeconds);

}

else

MessageBox.Show("С генерируйте массив");

}

/// <summary>

///

/// </summary>

/// <param name="sender"></param>

/// <param name="e"></param>

private void btnClose\_Click( object sender, EventArgs e )

{

Close( );

}

/// <summary>

///

/// </summary>

/// <param name="sender"></param>

/// <param name="e"></param>

private void button1\_Click( object sender, EventArgs e )

{

if (string.IsNullOrEmpty(txtArraySize.Text))

MessageBox.Show( "Укажите размер массива ");

else

if(string.IsNullOrEmpty(txtMinValue.Text) ||

string.IsNullOrEmpty(txtMaxValue.Text))

MessageBox.Show("Укажите диапазон чисел");

else

{

min\_value = int.Parse(txtMinValue.Text);

max\_value = int.Parse(txtMaxValue.Text);

array = new int[int.Parse(txtArraySize.Text)];

Random rand = new Random( );

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

{

array[i] = rand.Next(min\_value,max\_value);

}

}

}

#region Сортировка вставками

/// <summary>

/// Сортировка вставками

/// </summary>

/// <param name="array"></param>

/// <returns></returns>

public int[] InsertionSort( int[] array )

{

int[] result = new int[array.Length];

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

{

int j = i;

while (j > 0 && result[j - 1] > array[i])

{

result[j] = result[j - 1];

j--;

}

result[j] = array[i];

}

return result;

}

#endregion

#region Сортировка пузырьком

/// <summary>

/// Сортировка пузырьком

/// </summary>

/// <param name="array"></param>

public void BubbleSort( int[] array )

{

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

{

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

{

if (array[j] > array[j + 1])

{

int temp = array[j];

array[j] = array[j + 1];

array[j + 1] = temp;

}

}

}

}

#endregion

#region Сортировка слиянием

/// <summary>

/// Сортировка слиянием

/// </summary>

/// <param name="massive"></param>

/// <returns></returns>

public Int32[] Merge\_Sort( Int32[] massive )

{

if (massive.Length == 1)

return massive;

Int32 mid\_point = massive.Length / 2;

return Merge

(Merge\_Sort(massive.Take(mid\_point).ToArray( )),

Merge\_Sort(massive.Skip(mid\_point).ToArray( )));

}

private Int32[] Merge( Int32[] mass1, Int32[] mass2 )

{

Int32 a = 0, b = 0;

Int32[] merged = new int[mass1.Length + mass2.Length];

for (Int32 i = 0; i < mass1.Length + mass2.Length; i++)

{

if (b < mass2.Length && a < mass1.Length)

if (mass1[a] > mass2[b])

merged[i] = mass2[b++];

else //if int go for

merged[i] = mass1[a++];

else

if (b < mass2.Length)

merged[i] = mass2[b++];

else

merged[i] = mass1[a++];

}

return merged;

}

#endregion

#region Быстрая сортировка

/// <summary>

/// Быстрая сортировка

/// </summary>

/// <param name="array"></param>

/// <param name="start"></param>

/// <param name="end"></param>

public void quicksort( int[] array, int start, int end )

{

if (start >= end)

{

return;

}

int pivot = partition(array, start, end);

quicksort(array, start, pivot - 1);

quicksort(array, pivot + 1, end);

}

private int partition( int[] array, int start, int end )

{

int temp;//swap helper

int marker = start;//divides left and right subarrays

for (int i = start; i <= end; i++)

{

if (array[i] < array[end]) //array[end] is pivot

{

temp = array[marker]; // swap

array[marker] = array[i];

array[i] = temp;

marker += 1;

}

}

//put pivot(array[end]) between left and right subarrays

temp = array[marker];

array[marker] = array[end];

array[end] = temp;

return marker;

}

#endregion

#region Сортировка перемешиванием;

/// <summary>

/// Сортировка перемешиванием;

/// </summary>

/// <param name="name"></param>

public void SheiderSort( int[] name )

{

int b = 0;

int left = 0;//Левая граница

int right = name.Length - 1;//Правая граница

while (left < right)

{

for (int i = left; i < right; i++)//Слева направо...

{

if (name[i] > name[i + 1])

{

b = name[i];

name[i] = name[i + 1];

name[i + 1] = b;

b = i;

}

}

right = b;//Сохраним последнюю перестановку как границу

if (left >= right)

break;//Если границы сошлись выходим

for (int i = right; i > left; i--)//Справа налево...

{

if (name[i - 1] > name[i])

{

b = name[i];

name[i] = name[i - 1];

name[i - 1] = b;

b = i;

}

}

left = b;//Сохраним последнюю перестановку как границу

}

}

#endregion

#region Поразрядная сортировка

/// <summary>

/// Поразрядная сортировка

/// </summary>

/// <param name="array"></param>

/// <returns></returns>

public int[] RadixSort( int[] array )

{

bool isFinished = false;

int digitPosition = 0;

List<Queue<int>> buckets = new List<Queue<int>>( );

InitializeBuckets(buckets);

while (!isFinished)

{

isFinished = true;

foreach (int value in array)

{

int bucketNumber = GetBucketNumber(value, digitPosition);

if (bucketNumber > 0)

{

isFinished = false;

}

buckets[bucketNumber].Enqueue(value);

}

int i = 0;

foreach (Queue<int> bucket in buckets)

{

while (bucket.Count > 0)

{

array[i] = bucket.Dequeue( );

i++;

}

}

digitPosition++;

}

return array;

}

private int GetBucketNumber( int value, int digitPosition )

{

int bucketNumber = (value / (int)Math.Pow(10, digitPosition)) % 10;

return bucketNumber;

}

private static void InitializeBuckets( List<Queue<int>> buckets )

{

for (int i = 0; i < 10; i++)

{

Queue<int> q = new Queue<int>( );

buckets.Add(q);

}

}

#endregion

#region Сортировка подсчетом

/// <summary>

/// Сортировка подсчетом

/// </summary>

public int[] countingSort( int[] arr, int min, int max )

{

int[] count = new int[max - min + 1];

int z = 0;

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

count[i] = 0;

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

count[arr[i] - min]++;

for (int i = min; i <= max; i++)

while (count[i - min]-- > 0)

{

arr[z] = i;

z++;

}

return arr;

}

#endregion

КОНТРОЛЬНІ ЗАПИТАННЯ

1. Що таке мітка реального часу (Time Stamp Counter)?

*64-bit регістр, який поветрає число тіко с момента останнього скиду процесора.*

1. Які існують функції виміру часу?

*Такти, мілісекунди, секунди, години, доба.*

1. Яка функція дає найточніший результат?

*В мові C# найточніший результат дає метод StopWatch. Час виміряний цією функцією вимірюється в тактах, і є можливість перевести в мілісекунди.*

1. Які існують труднощі при точному вимірюванні часу у сучасних ОС?

*Такти потрібно переводити в секунди. Зазвичай це робиться не вручну, а за допомогою стандартних функцій.*