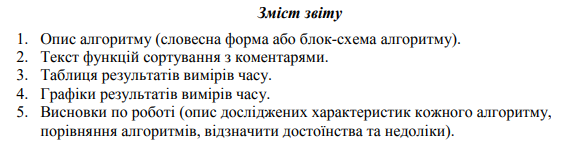
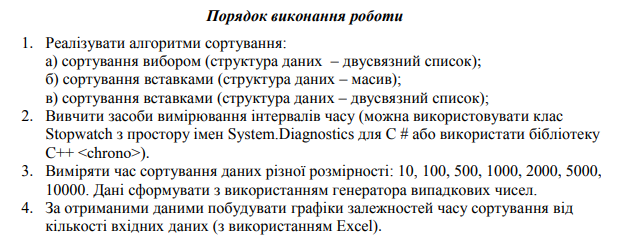
**Лабораторна робота № 5**

ПРОСТІ МЕТОДИ СОРТУВАННЯ

***Мета*** : реалізація простих алгоритмів сортування та дослідження їх характеристик (швидкодія, необхідний обсяг пам'яті, застосування тощо).

**5.1 Хід роботи**

****

****

5.1.1

**Завдання**:

Лістинг:

using System.Diagnostics;

namespace lab\_5

{

class Program

{

static void Main()

{

Console.OutputEncoding = System.Text.Encoding.Unicode;

Console.InputEncoding = System.Text.Encoding.Unicode;

Stopwatch stopwatch = new Stopwatch();

Random random = new Random();

LinkedList<int> list = new LinkedList<int>();

int[] mass\_1 = new int[10];

int[] mass\_2 = new int[100];

int[] mass\_3 = new int[500];

int[] mass\_4 = new int[1000];

int[] mass\_5 = new int[2000];

int[] mass\_6 = new int[5000];

int[] mass\_7 = new int[10000];

Console.WriteLine("a) сортування вибором (структура даних – двусвязний список);");

Console.WriteLine("б) сортування вставками (структура даних – масив);");

Console.WriteLine("в) сортування вставками (структура даних – двусвязний список); ");

int x = 10, num;

Console.WriteLine($"\nРозмірність {x}:");

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

{

num = random.Next(-x, x);

list.AddLast(num);

mass\_1[i] = num;

}

stopwatch.Reset();

stopwatch.Start();

listSelectionSort(list, x);

stopwatch.Stop();

list.Clear();

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

{

list.AddLast(mass\_1[i]);

}

Console.WriteLine($"а) {stopwatch.Elapsed.TotalMilliseconds} мс");

stopwatch.Reset();

stopwatch.Start();

massInsertionSort(mass\_1);

stopwatch.Stop();

Console.WriteLine($"б) {stopwatch.Elapsed.TotalMilliseconds} мс");

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

list.AddLast(random.Next(-x, x));

stopwatch.Reset();

stopwatch.Start();

listInsertionSort(list, x);

stopwatch.Stop();

list.Clear();

Console.WriteLine($"в) {stopwatch.Elapsed.TotalMilliseconds} мс");

////////////////////////////////////////////////////////////////////

x = 100;

Console.WriteLine($"\nРозмірність {x}:");

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

{

num = random.Next(-x, x);

list.AddLast(num);

mass\_2[i] = num;

}

stopwatch.Reset();

stopwatch.Start();

listSelectionSort(list, x);

stopwatch.Stop();

list.Clear();

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

{

list.AddLast(mass\_2[i]);

}

Console.WriteLine($"а) {stopwatch.Elapsed.TotalMilliseconds} мс");

stopwatch.Reset();

stopwatch.Start();

massInsertionSort(mass\_2);

stopwatch.Stop();

Console.WriteLine($"б) {stopwatch.Elapsed.TotalMilliseconds} мс");

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

list.AddLast(random.Next(-x, x));

stopwatch.Reset();

stopwatch.Start();

listInsertionSort(list, x);

stopwatch.Stop();

list.Clear();

Console.WriteLine($"в) {stopwatch.Elapsed.TotalMilliseconds} мс");

////////////////////////////////////////////////////////////////////

x = 500;

Console.WriteLine($"\nРозмірність {x}:");

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

{

num = random.Next(-x, x);

list.AddLast(num);

mass\_3[i] = num;

}

stopwatch.Reset();

stopwatch.Start();

listSelectionSort(list, x);

stopwatch.Stop();

list.Clear();

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

{

list.AddLast(mass\_3[i]);

}

Console.WriteLine($"а) {stopwatch.Elapsed.TotalMilliseconds} мс");

stopwatch.Reset();

stopwatch.Start();

massInsertionSort(mass\_3);

stopwatch.Stop();

Console.WriteLine($"б) {stopwatch.Elapsed.TotalMilliseconds} мс");

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

list.AddLast(random.Next(-x, x));

stopwatch.Reset();

stopwatch.Start();

listInsertionSort(list, x);

stopwatch.Stop();

list.Clear();

Console.WriteLine($"в) {stopwatch.Elapsed.TotalMilliseconds} мс");

////////////////////////////////////////////////////////////////////

x = 1000;

Console.WriteLine($"\nРозмірність {x}:");

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

{

num = random.Next(-x, x);

list.AddLast(num);

mass\_4[i] = num;

}

stopwatch.Reset();

stopwatch.Start();

listSelectionSort(list, x);

stopwatch.Stop();

list.Clear();

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

{

list.AddLast(mass\_4[i]);

}

Console.WriteLine($"а) {stopwatch.Elapsed.TotalMilliseconds} мс");

stopwatch.Reset();

stopwatch.Start();

massInsertionSort(mass\_4);

stopwatch.Stop();

Console.WriteLine($"б) {stopwatch.Elapsed.TotalMilliseconds} мс");

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

list.AddLast(random.Next(-x, x));

stopwatch.Reset();

stopwatch.Start();

listInsertionSort(list, x);

stopwatch.Stop();

list.Clear();

Console.WriteLine($"в) {stopwatch.Elapsed.TotalMilliseconds} мс");

////////////////////////////////////////////////////////////////////

x = 2000;

Console.WriteLine($"\nРозмірність {x}:");

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

{

num = random.Next(-x, x);

list.AddLast(num);

mass\_5[i] = num;

}

stopwatch.Reset();

stopwatch.Start();

listSelectionSort(list, x);

stopwatch.Stop();

list.Clear();

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

{

list.AddLast(mass\_5[i]);

}

Console.WriteLine($"а) {stopwatch.Elapsed.TotalMilliseconds} мс");

stopwatch.Reset();

stopwatch.Start();

massInsertionSort(mass\_5);

stopwatch.Stop();

Console.WriteLine($"б) {stopwatch.Elapsed.TotalMilliseconds} мс");

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

list.AddLast(random.Next(-x, x));

stopwatch.Reset();

stopwatch.Start();

listInsertionSort(list, x);

stopwatch.Stop();

list.Clear();

Console.WriteLine($"в) {stopwatch.Elapsed.TotalMilliseconds} мс");

////////////////////////////////////////////////////////////////////

x = 5000;

Console.WriteLine($"\nРозмірність {x}:");

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

{

num = random.Next(-x, x);

list.AddLast(num);

mass\_6[i] = num;

}

stopwatch.Reset();

stopwatch.Start();

listSelectionSort(list, x);

stopwatch.Stop();

list.Clear();

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

{

list.AddLast(mass\_6[i]);

}

Console.WriteLine($"а) {stopwatch.Elapsed.TotalMilliseconds} мс");

stopwatch.Reset();

stopwatch.Start();

massInsertionSort(mass\_6);

stopwatch.Stop();

Console.WriteLine($"б) {stopwatch.Elapsed.TotalMilliseconds} мс");

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

list.AddLast(random.Next(-x, x));

stopwatch.Reset();

stopwatch.Start();

listInsertionSort(list, x);

stopwatch.Stop();

list.Clear();

Console.WriteLine($"в) {stopwatch.Elapsed.TotalMilliseconds} мс");

////////////////////////////////////////////////////////////////////

x = 10000;

Console.WriteLine($"\nРозмірність {x}:");

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

{

num = random.Next(-x, x);

list.AddLast(num);

mass\_7[i] = num;

}

stopwatch.Reset();

stopwatch.Start();

listSelectionSort(list, x);

stopwatch.Stop();

list.Clear();

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

{

list.AddLast(mass\_7[i]);

}

Console.WriteLine($"а) {stopwatch.Elapsed.TotalMilliseconds} мс");

stopwatch.Reset();

stopwatch.Start();

massInsertionSort(mass\_7);

stopwatch.Stop();

Console.WriteLine($"б) {stopwatch.Elapsed.TotalMilliseconds} мс");

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

list.AddLast(random.Next(-x, x));

stopwatch.Reset();

stopwatch.Start();

listInsertionSort(list, x);

stopwatch.Stop();

list.Clear();

Console.WriteLine($"в) {stopwatch.Elapsed.TotalMilliseconds} мс");

//foreach (var person in list) Console.WriteLine(person);

}

static void listInsertionSort(LinkedList<int> list, int x)

{

int count = 1;

int k;

int j;

int temp;

var currentNode = list.First;

for (int i = 1; i < x; i++)//початок сортування вставок для масиву

{

currentNode = list.First;

for(count = 0; count < i; count++)//повернення до теперішнього елемента

currentNode = currentNode.Next;

k = currentNode.Value;//теперішній елемент

j = i;//індекс теперішнього елемента

while (j > 0 && currentNode.Previous.Value > k)//заміна

{

temp = currentNode.Value;

currentNode.Value = currentNode.Previous.Value;

currentNode.Previous.Value = temp;

j--;

currentNode = currentNode.Previous;

}

currentNode.Value = k;

}

}

static void massInsertionSort(int[] mass)

{

int k;

int j;

int temp;

for (int i = 1; i < mass.Length; i++)//початок сортування вставок для масиву

{

k = mass[i];//теперішній елемент

j = i;//індекс теперішнього елемента

while (j > 0 && mass[j - 1] > k)//заміна

{

temp = mass[j];

mass[j] = mass[j-1];

mass[j-1] = temp;

j--;

}

mass[j] = k;

}

}

static void listSelectionSort(LinkedList<int> list, int x, int currentIndex = 0, int count\_1 = 50000, int index = 0, int i = -1, int count\_2 = 50000)

{

for (currentIndex = 0; currentIndex < x; currentIndex++)//початок сортування вибором для списків

{

count\_2 = 50000;

i = -1;

index = 0;

foreach (var person in list)//пошук теперішнього та найменшого елемента

{

i++;//індекс теперішнього елемента

if (person < count\_2 && i > currentIndex)//найменший елемент

{

count\_2 = person;//найменший знайдений елемент

index = i;//індекс найменшого елемента

}

if (i == currentIndex)//теперішній елемент

count\_1 = person;

}

if (index != currentIndex && count\_2 <= count\_1)//заміна

{

i = -1;

var currentNode = list.First;

while (currentNode != null)//пошук цих елементів

{

i++;

if (i == currentIndex)//заміна

currentNode.Value = count\_2;

if (i == index)//заміна

currentNode.Value = count\_1;

currentNode = currentNode.Next;

}

}

}

}

}

}

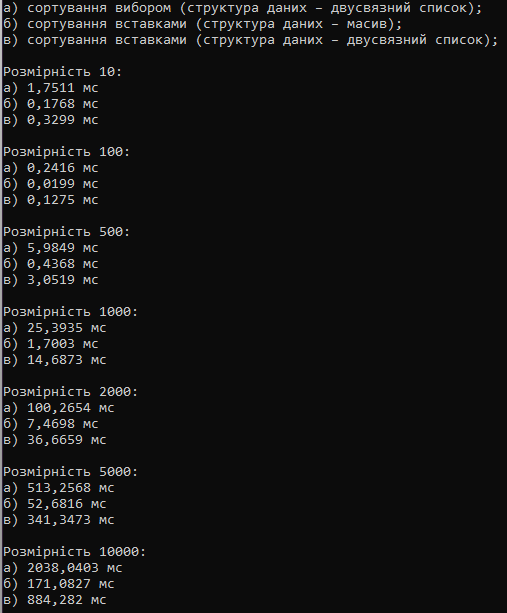


Рисунок 5.1 – Результат виконання завдання

***Словесний опис алгоритмів:***

**Сортування вибором**  – алгоритм сортування масиву, який по швидкості виконання можна зрівняти з сортуванням бульбашкою.

Алгоритм сортування вибором складається з наступних кроків:

* Для початку визначаємо позицію мінімального елементу масиву;
* Здійснюємо обмін мінімального елементу з елементом на початку масиву. Виходить, що перший елемент масиву вже відсортовано;
* Зменшуємо робочу область масиву, відкидаючи перший елемент, а для підмасиву, що залишився, повторюємо сортування.

Алгоритмічна складність: O(n) = n^2.

**Сортування вставками** - це алгоритм сортування, в якому всі елементи масиву почергово переглядаються, при цьому кожен елемент переміщається у відповідне місце серед раніше впорядкованих значень.

Алгоритм роботи сортування включенням наступний:

* На початку роботи впорядкована частина порожня;
* Додаємо в неї перший елемент масиву з не впорядкованих даних;
* Переходимо до наступного елементу в невідсортованих даних, і знаходимо йому правильну позицію у відсортованій частині масиву, цим ми розширюємо область впорядкованих даних;
* Повторюємо попередній крок для всіх елементів, що залишилися.

Алгоритмічна складність: O(n) = n^2.

***Графіки та таблиця:***

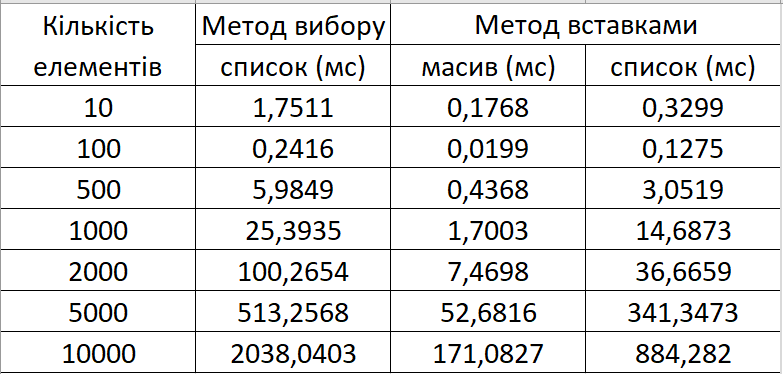


Рисунок 5.2 – Таблиця

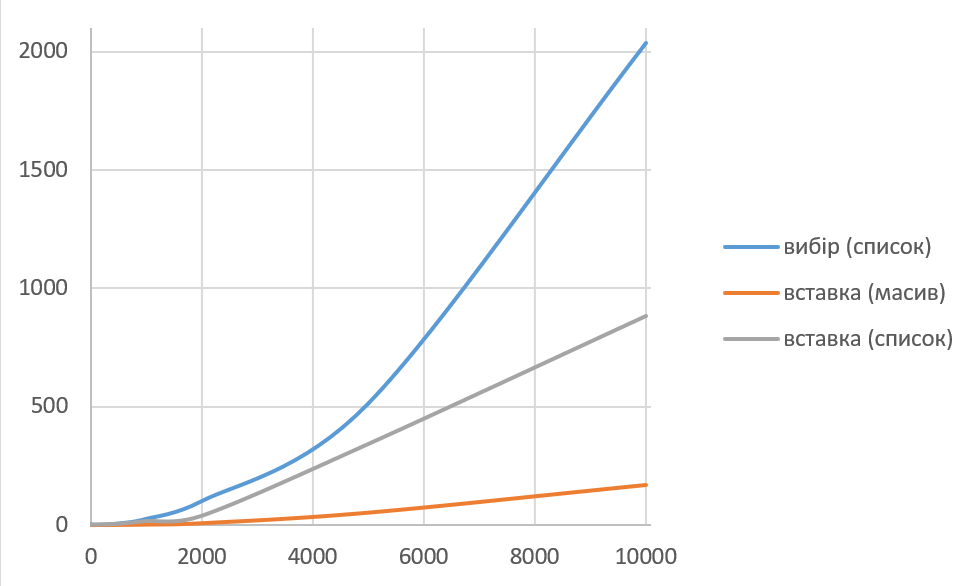


Рисунок 5.3 – Графіки

***Висновки:*** я дослідив та порівняв характеристики кожного з алгоритмів, та дійшов висновку, що сортування вибором найповільніше сортування, а вставка, навпаки виграє як для масиву, так і для списку, на мою думку, це відбувається так, тому що вставка, може бути частково відсортована, це значно допомагає в сортуванні, а під час вибірки, сортування не залежить від розміщення елемента, на малій кількості елементів, різниця невелика, але зі збільшенням їх кількості, збільшується і різниця, яка продемонстрована на графіку.