Звіт

з лабораторної роботи №6

з курсу алгоритми і методи обчислень.

(АЛГОРИТМИ СОРТУВАННЯ ЛІНІЙНИХ СТРУКТУР)

Студента 1-го курсу,

Парфенюка Тимофія Павловича

2019

Завдання:

відсортувати масив за допомогою алгоритмів Шелла або швидкого сортування. Використати вказаний алгоритм для виконання індивідуального завдання.



**Код:**

using System;

namespace Algoritmes6

{

class Program

{

static public int k = 0;

static public int por = 0;

static public void ShellSort(int[] mass)

{

int i, j, step, t1 = 1, t2 = 0, tmp;

t1 = t1 + 1;

for (step = mass.Length / 2; step > 0; step /= 2, t1++)

{

for (i = step; i < mass.Length; i++, t1++)

{

tmp = mass[i];

t1 = t1 + 1;

for (j = i; j >= step; j -= step, t1++)

{

if (tmp < mass[j - step])

{

mass[j] = mass[j - step];

t2 = t2 + 1;

t1 = t1 + 1;

k++;

}

else

break;

}

mass[j] = tmp;

k++;

t1 = t1 + 1;

}

}

}

static public int InputInt(string s)

{

int input = 0;

string cont = "";

do

{

try

{

cont = "";

Console.WriteLine(s);

input = int.Parse(Console.ReadLine());

}

catch (Exception e)

{

Console.WriteLine(e.Message + "Again?");

cont = Console.ReadLine();

}

} while (cont == "yes");

return input;

}

static public int[] CreateMas(int dim)

{

int[] Mas = new int[dim];

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

{

Mas[i] = InputInt("Input " + i + "-th element");

}

return Mas;

}

static public void PrintMas(int[] Mas)

{

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

{

if (Mas[i]%2==0) Console.Write("|{0}| ", Mas[i]);

else Console.Write("{0} ", Mas[i]);

}

Console.WriteLine("");

}

static public int[] EvenArray(int[] Mas)

{

int k = 0;

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

{

if (Mas[i] % 2 == 0)

{

k++;

}

}

int[] EArr = new int[k];

int n = 0;

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

{

if (Mas[i] % 2 == 0)

{

EArr[n] = Mas[i];

n++;

}

}

return EArr;

}

static public int[] IndexArray(int[] Mas)

{

int k = 0;

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

{

if (Mas[i] % 2 == 0)

{

k++;

}

}

int[] IArr = new int[k];

int n = 0;

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

{

if (Mas[i] % 2 == 0)

{

IArr[n] = i;

n++;

}

}

return IArr;

}

static public void SortEvenNumbers(int[] Mas)

{

string SortMethod = "";

int[] EArr = EvenArray(Mas);

int[] IArr = IndexArray(Mas);

do

{

Console.WriteLine("Chose sorting method: 1. Shell Sort(ss) 2. Quick Sort (qs)");

SortMethod = Console.ReadLine();

} while (SortMethod != "ss" && SortMethod != "qs");

if (SortMethod == "ss")

{

ShellSort(EArr);

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

{

Mas[IArr[i]] = EArr[i];

}

}

else if (SortMethod == "qs")

{

QuickSort(EArr, 0, EArr.Length - 1);

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

{

Mas[IArr[i]] = EArr[i];

}

}

else Console.WriteLine("Error!");

}

static public int[] Partition(int[] A, int L, int R)

{

int i = L,

j = R;

int x = A[(L + R) / 2];

Console.WriteLine("Pivot: {0}", x);

do

{

while (x > A[i]) { i++; por++; }

while (A[j] > x) { j--; por++; }

if (i <= j)

{

int tmp = A[i];

A[i] = A[j];

A[j] = tmp;

Console.WriteLine($"Element {i} changed with element {j}");

i++;

j--;

k++;

por++;

}

} while (i < j);

PrintMas(A);

Console.WriteLine();

int[] ij = new int[2] { i, j };

return ij;

}

static public void QuickSort(int[] A, int L, int R)

{

int[] ij = Partition(A, L, R);

int i = ij[0];

int j = ij[1];

if (L < j) QuickSort(A, L, j);

if (i < R) QuickSort(A, i, R);

}

static void Main(string[] args)

{

int[] Arr = CreateMas(InputInt("Input number of numbers in array: "));

PrintMas(Arr);

QuickSort(Arr, 0, Arr.Length - 1);

//SortEvenNumbers(Arr);

PrintMas(Arr);

Console.WriteLine($"Number of changes: {k}");

Console.WriteLine($"Number of compare: {por}");

}

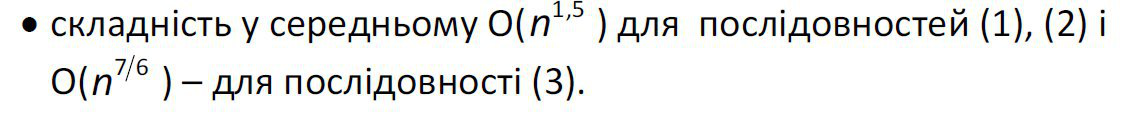
}

}

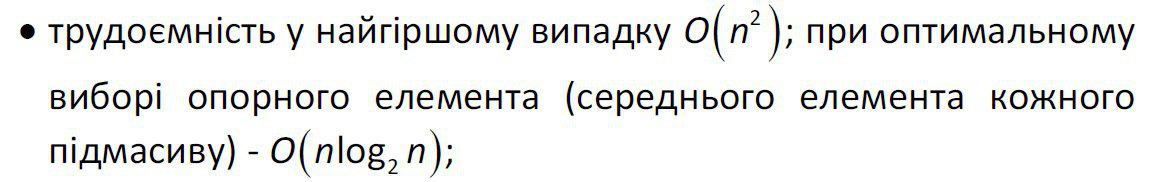
***Висновок:***

Алгоритми сортування мають різну складність:

**Shellsort:**

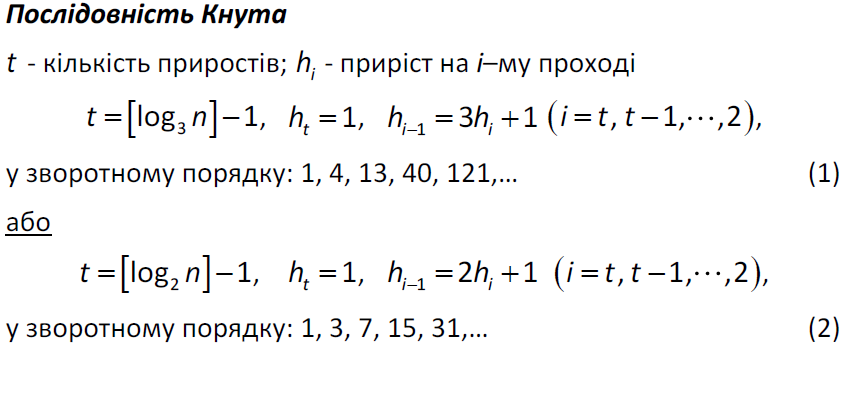


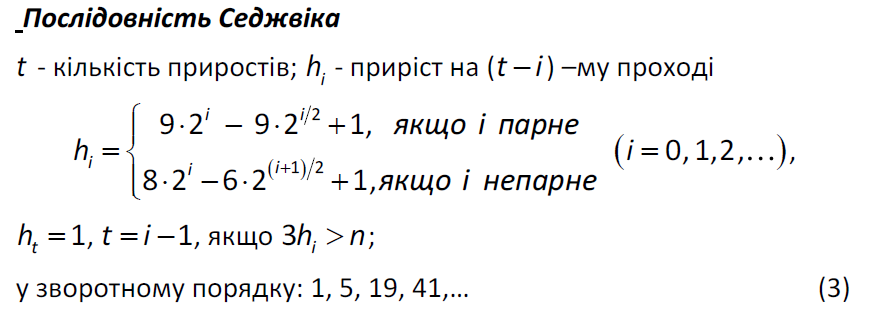
**QuickSort:**



Для масивыв з малою кількістю елементів краще всього сортувати Shellsort, а коли елементів більше, то QuickSort.

Для Shellsort рекомендується використовувати послідовність Кнута або послідовність Седжвіка. Для послідовності Кнута в середньому менша складність:





***Вхідні та вихідні данні:***

