Національний технічний університет України

«Київський політехнічний інститут імені Ігоря Сікорського»

Факультет інформатики та обчислювальної техніки

Кафедра технічної кібернетики

ДИСКРЕТНІ СТРУКТУРИ

ЛАБОРАТОРНА РОБОТА №2

ТЕМА: Спеціальні алгоритми сортування

Виконали:

Студенти гр. ІТ-82

Бірюченко О.

Жук О.

Михайлова Д.

Кавун Ф.

Перевірив:

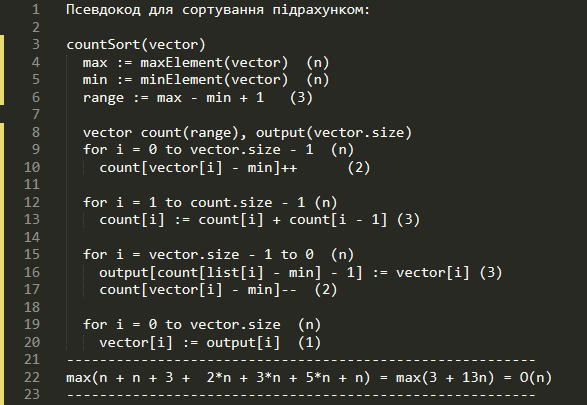
Сергеєв Д. С..

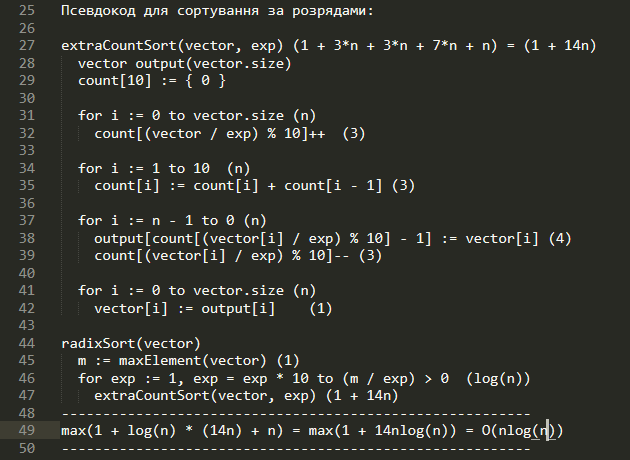
Київ – 2019

1. Обраний алгоритм сортування:

Сортування підрахунком (Count Sort) та сортування за розрядами (Radix Sort)

2. Складність алгоритму та псевдокод:





3. Програма:

#include "pch.h"

#include <iostream>

#include <vector>

#include <algorithm>

#include <fstream>

#include <string>

#include <ctime>

#include "lab5.h"

using namespace std;

void countSort(vector <int> &arr)

{

   int max = \*max\_element(arr.begin(), arr.end());

   int min = \*min\_element(arr.begin(), arr.end());

   int range = max - min + 1;

   vector<int> count(range), output(arr.size());

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

         count[arr[i] - min]++;

   for (int i = 1; i < count.size(); i++)

         count[i] += count[i - 1];

   for (int i = arr.size() - 1; i >= 0; i--)

   {

         output[count[arr[i] - min] - 1] = arr[i];

         count[arr[i] - min]--;

   }

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

         arr[i] = output[i];

}

void countSort1(vector <int> &arr, int n, int exp)

{

   vector <int> output(n);

   int i, count[10] = { 0 };

    for (i = 0; i < n; i++)

         count[(arr[i] / exp) % 10]++;

   for (i = 1; i < 10; i++)

         count[i] += count[i - 1];

    for (i = n - 1; i >= 0; i--)

   {

         output[count[(arr[i] / exp) % 10] - 1] = arr[i];

         count[(arr[i] / exp) % 10]--;

   }

   for (i = 0; i < n; i++)

         arr[i] = output[i];

}

void radixsort(vector <int> &arr, int n)

{

   int m = \*max\_element(arr.begin(), arr.end());

   for (int exp = 1; m / exp > 0; exp \*= 10)

         countSort1(arr, n, exp);

}

void printArray(vector <int> & arr)

{

   for (int i = 0; i < 6; i++) cout << arr[i] << "\n";

         cout << "...\n";

         cout << "...\n";

         cout << "...\n";

   for (int i = arr.size() - 6; i < arr.size(); i++)

         cout << arr[i] << "\n";

}

int main()

{

   cout << "-----------------Add to vector----------------\n";

   cout << "Adding elements to vector time:\n";

   vector<int> arr, arr1;

   ifstream fin;

   int x;

   unsigned int start\_time, finish\_time;

         fin.open("input.txt");

         start\_time = clock();

         while (!fin.eof()) {

                fin >> x;

                arr.push\_back(x);

         }

         finish\_time = clock();

         fin.close();

   cout << "First Time: " << finish\_time - start\_time << "ms\n";

   fin.open("input.txt");

   start\_time = clock();

   while (!fin.eof()) {

         fin >> x;

         arr1.push\_back(x);

   }

   finish\_time = clock();

   fin.close();

   cout << "Second Time: " << finish\_time - start\_time << "ms\n";

   cout << "-----------------------------------------------\n";

   cout << "-------------Count Sort function----------\n";

         start\_time = clock();

         countSort(arr);

         finish\_time = clock();

   cout << "Time: " << finish\_time - start\_time << " ms\n";

   cout << "-----------------------------------------------\n";

   cout << "------------------Print array----------------\n";

   printArray(arr);

   cout << "------------Radix Sort function-----------\n";

   int d = arr1.size();

   start\_time = clock();

   radixsort(arr1, d);

   finish\_time = clock();

   cout << "Time: " << finish\_time - start\_time << " ms\n";

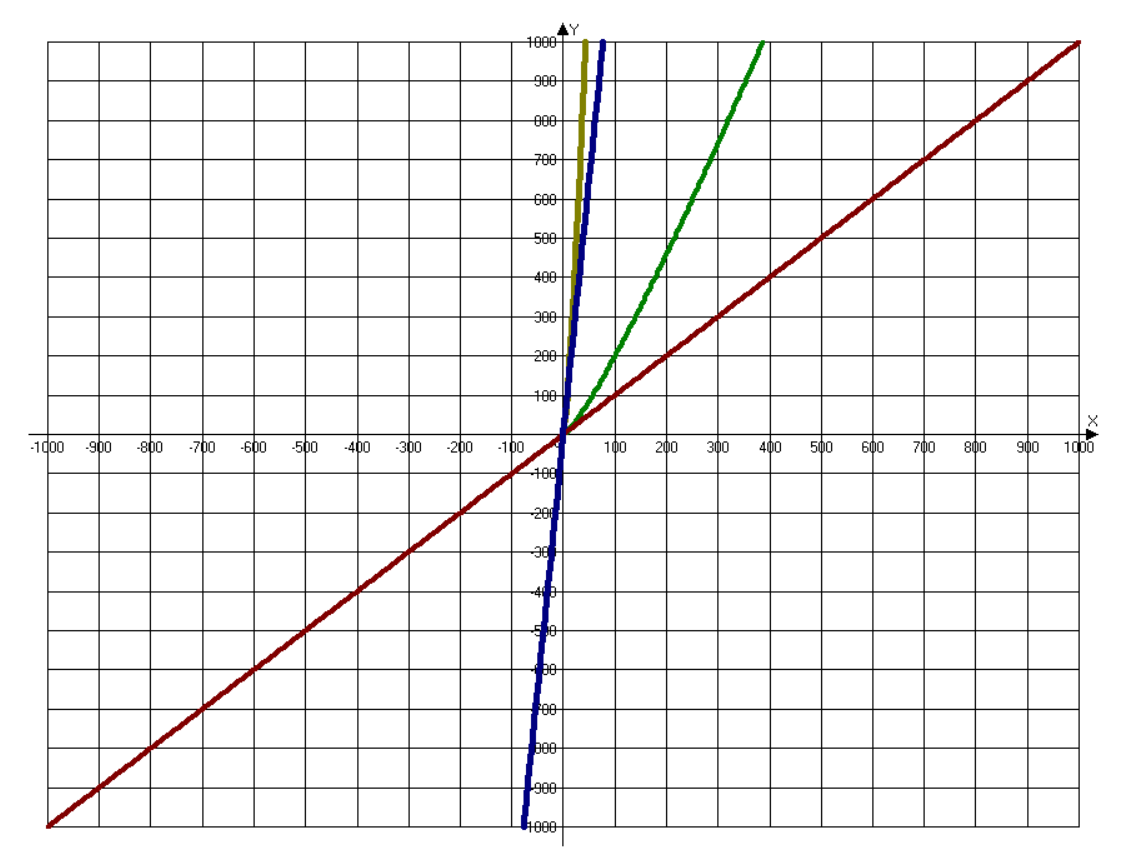
   cout << "------------------Print array----------------\n";

   printArray(arr1);

   return 0;

}

3. Графіки та час виконання

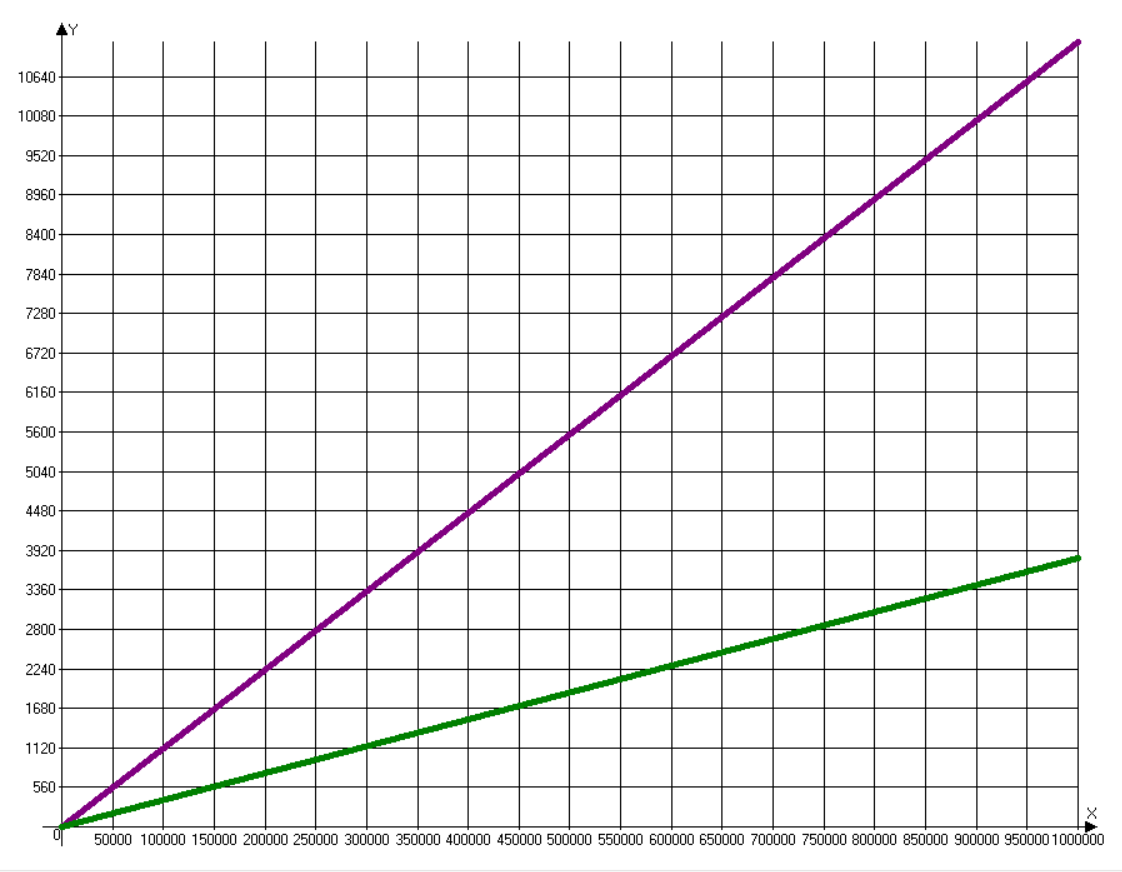


Графік 1,2

Червоний - складність  сортування підрахунком O(n); зелений- складність  сортування за розрядами O(nlogn);

жовтий - формула складності  сортування за розрядами; синій- формула складності  сортування підрахунком

Час виконання:



Графік 3

Фіолетовий - час виконання для сортування за розрядами; зелений - час виконання для сортування підрахунком

