Белорусский государственный технологический университет

Факультет информационных технологий

Кафедра Программной Инженерии

Лабораторная работа 15

По дисциплине «Основы алгоритмизации и программирования»

На тему «Сортировка массивов»

Выполнил:

Студент 1 курса 3 группы

Шатерник Г.И.

Преподаватель: асс. Андронова М.В.

2023, Минск

5. В соответствии со своим вариантом написать программу сортировок массивов указанными в таблице методами. Исходные массивы заполняются случайными числами. Определить зависимость времени выполнения алгоритмов от количества элементов для каждого из алгоритмов. Выполнить моделирование для массивов различных размеров.

Ввести массивы **А** и **В**. В массив **С** перенести те элементы массива **А**, которые меньше максимального элемента массива **В**. Массив **С** отсортировать по убыванию, используя алгоритмы сортировок: «пузырек», сортировка Хоара.

|  |
| --- |
| Программный код  #include <iostream>  #include <vector>  #include <ctime>  #include <cstdlib>  #include <chrono>  #include <algorithm>  using namespace std;  // Функция для генерации случайных чисел  int random\_number() {  return rand() % 10000;  }  // Сортировка подсчетом  void counting\_sort(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];  }  // Сортировка выбором  void selection\_sort(vector<int>& arr) {  for (int i = 0; i < arr.size() - 1; i++) {  int min\_index = i;  for (int j = i + 1; j < arr.size(); j++) {  if (arr[j] < arr[min\_index])  min\_index = j;  }  swap(arr[i], arr[min\_index]);  }  }  // Слияние двух подмассивов  void merge(vector<int>& arr, int left, int mid, int right) {  int n1 = mid - left + 1;  int n2 = right - mid;  vector<int> L(n1), R(n2);  for (int i = 0; i < n1; i++)  L[i] = arr[left + i];  for (int j = 0; j < n2; j++)  R[j] = arr[mid + 1 + j];  int i = 0, j = 0, k = left;  while (i < n1 && j < n2) {  if (L[i] <= R[j]) {  arr[k] = L[i];  i++;  }  else {  arr[k] = R[j];  j++;  }  k++;  }  while (i < n1) {  arr[k] = L[i];  i++;  k++;  }  while (j < n2) {  arr[k] = R[j];  j++;  k++;  }  }  // Сортировка слиянием  void merge\_sort(vector<int>& arr, int left, int right) {  if (left < right) {  int mid = left + (right - left) / 2;  merge\_sort(arr, left, mid);  merge\_sort(arr, mid + 1, right);  merge(arr, left, mid, right);  }  }  int main() {  srand(time(0));  vector<int> sizes = { 1000, 2000, 3000, 4000, 5000 };  for (int size : sizes) {  vector<int> arr(size);  generate(arr.begin(), arr.end(), random\_number);  vector<int> arr\_copy1 = arr;  vector<int> arr\_copy2 = arr;  auto start = chrono::high\_resolution\_clock::now();  counting\_sort(arr);  auto end = chrono::high\_resolution\_clock::now();  chrono::duration<double> elapsed = end - start;  cout << "Counting sort (" << size << " elements): " << elapsed.count() << " seconds" << endl;  start = chrono::high\_resolution\_clock::now();  selection\_sort(arr\_copy1);  end = chrono::high\_resolution\_clock::now();  elapsed = end - start;  cout << "Selection sort (" << size << " elements): " << elapsed.count() << " seconds" << endl;  start = chrono::high\_resolution\_clock::now();  merge\_sort(arr\_copy2, 0, size - 1);  end = chrono::high\_resolution\_clock::now();  elapsed = end - start;  cout << "Merge sort (" << size << " elements): " << elapsed.count() << " seconds" << endl;  cout << endl;  }  return 0;  } |
| Вывод: |