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 <ctime>  #include <algorithm>  using namespace std;  const int MAX\_SIZE = 100000; // максимальный размер массива  // функция для заполнения массива случайными числами  void fill(int arr[], int size) {  for (int i = 0; i < size; i++) {  arr[i] = rand() % 100; // случайное число от 0 до 99  }  }  // функция для вывода массива на экран  void printArray(int arr[], int size) {  for (int i = 0; i < size; i++) {  cout << arr[i] << " ";  }  cout << endl;  }  // функция для сортировки массива методом пузырька  void bubbleSort(int arr[], int size) {  for (int i = 0; i < size - 1; i++) {  for(int j = 0; j < size - i - 1; j++) {  if (arr[j] < arr[j + 1]) {  swap(arr[j], arr[j + 1]);  }  }  }  }  // функция для быстрой сортировки массива методом Хоара  void quickSort(int arr[], int left, int right) {  int i = left, j = right;  int pivot = arr[(left + right) / 2];  while (i <= j) {  while (arr[i] > pivot) {  i++;  }  while (arr[j] < pivot) {  j--;  }  if (i <= j) {  swap(arr[i], arr[j]);  i++;  j--;  }  }  if (left < j) {  quickSort(arr, left, j);  }  if (i < right) {  quickSort(arr, i, right);  }  }  int main() {  int A[MAX\_SIZE], B[MAX\_SIZE], C[MAX\_SIZE];  int sizeA, sizeB, sizeC = 0;  int maxB = 0;  clock\_t start, end; // переменные для замера времени выполнения  // ввод размеров массивов  cout << "Enter size of array A: ";  cin >> sizeA;  cout << "Enter size of array B: ";  cin >> sizeB;  // заполнение массивов случайными числами  fill(A, sizeA);  fill(B, sizeB);  // вывод исходных массивов на экран  cout << "Array A: ";  printArray(A, sizeA);  cout << "Array B: ";  printArray(B, sizeB);  // поиск максимального элемента в массиве B  for (int i = 0; i < sizeB; i++) {  if (B[i] > maxB) {  maxB = B[i];  }  }  // перенос элементов из массива A в массив C  for (int i = 0; i < sizeA; i++) {  if (A[i] < maxB) {  C[sizeC] = A[i];  sizeC++;  }  }  // вывод массива C на экран  cout << "Array C: ";  printArray(C, sizeC);  // замер времени выполнения сортировки пузырьком  start = clock();  bubbleSort(C, sizeC);  end = clock();  cout << "Bubble sort time: " << (double)(end - start) / CLOCKS\_PER\_SEC << " seconds" << endl;  // замер времени выполнения быстрой сортировки  start = clock();  quickSort(C, 0, sizeC - 1);  end = clock();  cout << "Quick sort time: " << (double)(end - start) / CLOCKS\_PER\_SEC << "seconds" << endl;  return 0;  } |
| Вывод: |