Балтийский государственный технический университет  
«ВОЕНМЕХ» им. Д. Ф. Устинова

Кафедра О7 «Информационные системы и программная инженерия»

**Практическая работа №3**по дисциплине «Структуры и организация данных»  
на тему «Оценка эффективности алгоритмов»  
часть 1 «Алгоритмы сортировки»  
  
вариант 25

Выполнил:  
Студент Фокин В. К.  
Группа И924Б  
  
Преподаватель:  
Гладевич А. А.

Санкт-Петербург  
2023г.

Уровень сложности – базовый. Провести сравнение указанных алгоритмов сортировки массивов, содержащих *N1*, *N2*, *N3* и *N4* элементов, по указанному в вариативной части критерию и объему требуемой дополнительно памяти.

Порядок: по возрастанию элементов.

Методы: пузырька, Шелла (шаг сортировки ), Шелла (шаг сортировки задается числами Фибоначчи), поразрядная сортировка.

N1=10000, N2=30000, N3=70000, N4=100000. Критерий – количество присваиваний.

1. Алгоритм сортировки пузырьком:

Алгоритм проходит по списку несколько раз, сравнивая два соседних элемента и меняя их местами, если они находятся в неправильном порядке.

Трудоемкость сортировки пузырьком по количеству присваиваний:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Случай** | **Ситуация, соответствующая случаю** | **Обоснование** | **Ожидаемое число присваиваний элементов массива** | **Асимптотическая оценка сложности по количеству присваиваний** | **Ожидаемое число вспомогательных присваиваний** |
| наилучший | массив упорядочен | перемещение элементов не требуется, все элементы находятся на своих местах, операции присваивания не производятся | N1: 0  N2: 0  N3: 0  N4: 0 |  | N1: 50 014 999  N2: 450 044 999  N3: 2 450 104 999  N4: 50 001 499 999 |
| наихудший | массив упорядочен в обратном порядке | на каждом проходе алгоритма  выполняется максимальное количество перестановок элементов в каждой паре соседних элементов | N1: 149 985 000  N2: 1 349 955 000  N3: 7 349 895 000  N4: 14 999 850 000 |  | N1: 50 014 999  N2: 450 044 999  N3: 2 450 104 999  N4: 50 001 499 999 |
| средний | массив не упорядочен | какие-то элементы уже находятся на своем месте и часть операций присваивания не будут производится | N1: 74 992 500  N2: 674 977 500  N3: 3 674 947 500  N4: 7 499 925 000 |  | N1: 50 014 999  N2: 450 044 999  N3: 2 450 104 999  N4: 50 001 499 999 |

Пространственная сложность – (две вспомогательные переменные и переменная цикла)

1. Алгоритм сортировки Шелла:

Модифицированный алгоритма вставки, который разбивает список на подсписки и сортирует каждый из них отдельно. Он уменьшает количество перемещений элементов и выполняет сортировку быстрее.

Трудоемкость алгоритма сортировки Шелла по количеству присваиваний:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Случай** | **Ситуация, соответствующая случаю** | **Обоснование** | **Ожидаемое число присваиваний элементов массива** | **Асимптотическая оценка сложности по количеству присваиваний** | **Ожидаемое число вспомогательных присваиваний** |
| наилучший | массив упорядочен | перемещение элементов не требуется, все элементы находятся на своих местах, операции присваивания не производятся | 0  N1: 0  N2: 0  N3: 0  N4: 0 |  | N1: 260 000  N2: 880 000  N3: 2 200 000  N4: 3 200 000 |
| наихудший | массив упорядочен в обратном порядке | начальное расположение элементов массива хаотично, элементы реже попадают на свою места, производится больше операций присваивания | N1: 530 000  N2: 1 700 000  N3: 4 500 000  N4: 6 600 000 |  | N1: 400 000  N2: 1 300 000  N3: 3 400 000  N4: 5 000 000 |
| средний | массив не упорядочен | при сортировке элементы окажутся ближе к нужным местам, производится меньше операций присваивания | N1: 130 000  N2: 440 000  N3: 1 100 000  N4: 1 600 000 |  | N1: 260 000  N2: 880 000  N3: 2 200 000  N4: 3 200 000 |

Пространственная сложность – (две переменные цикла и две вспомогательные переменные)

1. Алгоритм сортировки Шелла (шаг задан числами Фибоначчи):

Модификация классического алгоритма Шелла для сортировки массивов

Трудоемкость алгоритма сортировки Шеллапо количеству присваиваний:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Случай** | **Ситуация, соответствующая случаю** | **Обоснование** | **Ожидаемое число присваиваний элементов массива** | **Асимптотическая оценка сложности по количеству присваиваний** | **Ожидаемое число вспомогательных присваиваний** |
| наилучший | массив упорядочен | перемещение элементов не требуется, все элементы находятся на своих местах, операции присваивания не производятся | 0  N1: 0  N2: 0  N3: 0  N4: 0 |  | N1: 260 000  N2: 880 000  N3: 2 200 000  N4: 3 200 000 |
| наихудший | массив упорядочен в обратном порядке | начальное расположение элементов массива хаотично, элементы реже попадают на свою места, производится больше операций присваивания | N1: 600 000  N2: 2 200 000  N3: 5 600 000  N4: 8 300 000 |  | N1: 530 000  N2: 1 700 000  N3: 4 500 000  N4: 6 600 000 |
| средний | массив не упорядочен | при сортировке элементы окажутся ближе к нужным местам, производится меньше операций присваивания | N1: 80 000  N2: 260 000  N3: 680 000  N4: 1 000 000 |  | N1: 400 000  N2: 1 300 000  N3: 3 400 000  N4: 5 000 000 |

Пространственная сложность – (две переменных цикла и четыре вспомогательных переменная).

1. Алгоритм поразрядной сортировки:

Алгоритм сортировки работает с числами путем разбиения их на разряды и последовательной сортировки этих разрядов. Алгоритм начинается с сортировки младших разрядов и заканчивается сортировкой старших разрядов.

Трудоемкость поразрядной сортировки по количеству присваиваний:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Случай** | **Ситуация, соответствующая случаю** | **Обоснование** | **Ожидаемое число присваиваний элементов массива** | **Асимптотическая оценка сложности по количеству присваиваний** | **Ожидаемое число вспомогательных присваиваний** |
| наилучший  наихудший  средний | массив упорядочен, массив не упорядочен, массив упорядочен в обратном порядке | начальное расположение элементов в массиве не влияет на трудоемкость сортировки | N1: 80 000  N2: 240 000  N3: 560 000  N4: 800 000 |  | N1: 480 000  N2: 1 440 000  N3: 3 360 000  N4: 4 800 000 |

Пространственная сложность – , где m – мощность алфавита (массив для сохранения элементов массив в новом порядке и массив счетчиков).

Текст программы:

Bubble.cpp:

#include <iostream>

#include <fstream>

#include <ctime>

#include <algorithm>

using namespace std;

const int n = 10, n1 = 10000, n2 = 30000, n3 = 70000, n4 = 100000;

unsigned long long int cntMain, cntAux;

int memory;

void bubbleSort(long arr[], int n) {

memory = 0;

cntAux = 0;

cntMain = 0;

int i, j, temp;

for (cntAux++, i = 0; i < n - 1; cntAux++, i++) {

for (cntAux++, j = 0; j < n - i - 1; cntAux++, j++) {

if (arr[j] > arr[j + 1]) {

temp = arr[j];

arr[j] = arr[j + 1];

arr[j + 1] = temp;

cntMain += 3;

}

}

}

memory = 3 \* sizeof(int);

}

int main() {

long arr1[n1], arr2[n2], arr3[n3], arr4[n4], s = 0;

char menu;

clock\_t startTime = 0;

clock\_t endTime = 0;

do {

ifstream file("C:\\Data Structures\\pr3\_1\\programm3\_1\\test\_numbers.txt");

cout << "----------------------------------------------------------------------" << endl;

cout << "1. Sort 10000 elements" << endl;

cout << "2. Sort 30000 elements" << endl;

cout << "3. Sort 70000 elements" << endl;

cout << "4. Sort 100000 elements" << endl;

cout << "0. Exit" << endl;

cout << "----------------------------------------------------------------------" << endl;

cin >> menu;

switch (menu) {

case '1':

for (int i = 0; i < n1; i++) {

file >> arr1[i];

}

startTime = 0;

endTime = 0;

cout << "----------------------------------------------------------------------" << endl;

cout << "1. Sorting an unordered array" << endl;

cout << "Part of the initial array:" << endl;

for (int i = 0; i < n; i++) {

cout << arr1[i] << " ";

}

startTime = clock();

bubbleSort(arr1, n1);

endTime = clock();

cout << "\n\nPart of the resulting array:" << endl;

for (int i = 0; i < n; i++) {

cout << arr1[i] << " ";

}

cout << endl;

cout << "\nAdditional memory used: " << memory << " bytes." << endl;

cout << "Time taken by program is : ";

cout << double(endTime - startTime) / CLOCKS\_PER\_SEC;

cout << " sec " << endl;

cout << "Number of assignments: " << cntMain << endl;

cout << "Number of auxiliary assignments: " << cntAux << endl;

cout << "----------------------------------------------------------------------" << endl;

startTime = 0;

endTime = 0;

cout << "2. Sorting an ordered array" << endl;

cout << "Part of the initial array:" << endl;

for (int i = 0; i < n; i++) {

cout << arr1[i] << " ";

}

startTime = clock();

bubbleSort(arr1, n1);

endTime = clock();

cout << "\n\nPart of the resulting array:" << endl;

for (int i = 0; i < n; i++) {

cout << arr1[i] << " ";

}

cout << endl;

cout << "\nAdditional memory used: " << memory << " bytes." << endl;

cout << "Time taken by program is : ";

cout << double(endTime - startTime) / CLOCKS\_PER\_SEC;

cout << " sec " << endl;

cout << "Number of assignments: " << cntMain << endl;

cout << "Number of auxiliary assignments: " << cntAux << endl;

cout << "----------------------------------------------------------------------" << endl;

startTime = 0;

endTime = 0;

cout << "3. Sorting an array sorted in reverse order" << endl;

cout << "Part of the initial array:" << endl;

s = sizeof(arr1) / sizeof(arr1[0]);

reverse(arr1, arr1 + s);

for (int i = 0; i < n; i++) {

cout << arr1[i] << " ";

}

startTime = clock();

bubbleSort(arr1, n1);

endTime = clock();

cout << "\n\nPart of the resulting array:" << endl;

for (int i = 0; i < n; i++) {

cout << arr1[i] << " ";

}

cout << endl;

cout << "\nAdditional memory used: " << memory << " bytes." << endl;

cout << "Time taken by program is : ";

cout << double(endTime - startTime) / CLOCKS\_PER\_SEC;

cout << " sec " << endl;

cout << "Number of assignments: " << cntMain << endl;

cout << "Number of auxiliary assignments: " << cntAux << endl;

cout << "----------------------------------------------------------------------" << endl;

file.close();

break;

case '2':

for (int i = 0; i < n2; i++) {

file >> arr2[i];

}

startTime = 0;

endTime = 0;

cout << "----------------------------------------------------------------------" << endl;

cout << "1. Sorting an unordered array" << endl;

cout << "Part of the initial array:" << endl;

for (int i = 0; i < n; i++) {

cout << arr2[i] << " ";

}

startTime = clock();

bubbleSort(arr2, n2);

endTime = clock();

cout << "\n\nPart of the resulting array:" << endl;

for (int i = 0; i < n; i++) {

cout << arr2[i] << " ";

}

cout << endl;

cout << "\nAdditional memory used: " << memory << " bytes." << endl;

cout << "Time taken by program is : ";

cout << double(endTime - startTime) / CLOCKS\_PER\_SEC;

cout << " sec " << endl;

cout << "Number of assignments: " << cntMain << endl;

cout << "Number of auxiliary assignments: " << cntAux << endl;

cout << "----------------------------------------------------------------------" << endl;

startTime = 0;

endTime = 0;

cout << "2. Sorting an ordered array" << endl;

cout << "Part of the initial array:" << endl;

for (int i = 0; i < n; i++) {

cout << arr2[i] << " ";

}

startTime = clock();

bubbleSort(arr2, n2);

endTime = clock();

cout << "\n\nPart of the resulting array:" << endl;

for (int i = 0; i < n; i++) {

cout << arr2[i] << " ";

}

cout << endl;

cout << "\nAdditional memory used: " << memory << " bytes." << endl;

cout << "Time taken by program is : ";

cout << double(endTime - startTime) / CLOCKS\_PER\_SEC;

cout << " sec " << endl;

cout << "Number of assignments: " << cntMain << endl;

cout << "Number of auxiliary assignments: " << cntAux << endl;

cout << "----------------------------------------------------------------------" << endl;

startTime = 0;

endTime = 0;

cout << "3. Sorting an array sorted in reverse order" << endl;

cout << "Part of the initial array:" << endl;

s = sizeof(arr2) / sizeof(arr2[0]);

reverse(arr2, arr2 + s);

for (int i = 0; i < n; i++) {

cout << arr2[i] << " ";

}

startTime = clock();

bubbleSort(arr2, n2);

endTime = clock();

cout << "\n\nPart of the resulting array:" << endl;

for (int i = 0; i < n; i++) {

cout << arr2[i] << " ";

}

cout << endl;

cout << "\nAdditional memory used: " << memory << " bytes." << endl;

cout << "Time taken by program is : ";

cout << double(endTime - startTime) / CLOCKS\_PER\_SEC;

cout << " sec " << endl;

cout << "Number of assignments: " << cntMain << endl;

cout << "Number of auxiliary assignments: " << cntAux << endl;

cout << "----------------------------------------------------------------------" << endl;

file.close();

break;

case '3':

for (int i = 0; i < n3; i++) {

file >> arr3[i];

}

startTime = 0;

endTime = 0;

cout << "----------------------------------------------------------------------" << endl;

cout << "1. Sorting an unordered array" << endl;

cout << "Part of the initial array:" << endl;

for (int i = 0; i < n; i++) {

cout << arr3[i] << " ";

}

startTime = clock();

bubbleSort(arr3, n3);

endTime = clock();

cout << "\n\nPart of the resulting array:" << endl;

for (int i = 0; i < n; i++) {

cout << arr3[i] << " ";

}

cout << endl;

cout << "\nAdditional memory used: " << memory << " bytes." << endl;

cout << "Time taken by program is : ";

cout << double(endTime - startTime) / CLOCKS\_PER\_SEC;

cout << " sec " << endl;

cout << "Number of assignments: " << cntMain << endl;

cout << "Number of auxiliary assignments: " << cntAux << endl;

cout << "----------------------------------------------------------------------" << endl;

startTime = 0;

endTime = 0;

cout << "2. Sorting an ordered array" << endl;

cout << "Part of the initial array:" << endl;

for (int i = 0; i < n; i++) {

cout << arr3[i] << " ";

}

startTime = clock();

bubbleSort(arr3, n3);

endTime = clock();

cout << "\n\nPart of the resulting array:" << endl;

for (int i = 0; i < n; i++) {

cout << arr3[i] << " ";

}

cout << endl;

cout << "\nAdditional memory used: " << memory << " bytes." << endl;

cout << "Time taken by program is : ";

cout << double(endTime - startTime) / CLOCKS\_PER\_SEC;

cout << " sec " << endl;

cout << "Number of assignments: " << cntMain << endl;

cout << "Number of auxiliary assignments: " << cntAux << endl;

cout << "----------------------------------------------------------------------" << endl;

startTime = 0;

endTime = 0;

cout << "3. Sorting an array sorted in reverse order" << endl;

cout << "Part of the initial array:" << endl;

s = sizeof(arr3) / sizeof(arr3[0]);

reverse(arr3, arr3 + s);

for (int i = 0; i < n; i++) {

cout << arr3[i] << " ";

}

startTime = clock();

bubbleSort(arr3, n3);

endTime = clock();

cout << "\n\nPart of the resulting array:" << endl;

for (int i = 0; i < n; i++) {

cout << arr3[i] << " ";

}

cout << endl;

cout << "\nAdditional memory used: " << memory << " bytes." << endl;

cout << "Time taken by program is : ";

cout << double(endTime - startTime) / CLOCKS\_PER\_SEC;

cout << " sec " << endl;

cout << "Number of assignments: " << cntMain << endl;

cout << "Number of auxiliary assignments: " << cntAux << endl;

cout << "----------------------------------------------------------------------" << endl;

file.close();

break;

case '4':

for (long i = 0; i < n4; i++) {

file >> arr4[i];

}

startTime = 0;

endTime = 0;

cout << "----------------------------------------------------------------------" << endl;

cout << "1. Sorting an unordered array" << endl;

cout << "Part of the initial array:" << endl;

for (int i = 0; i < n; i++) {

cout << arr4[i] << " ";

}

startTime = clock();

bubbleSort(arr4, n4);

endTime = clock();

cout << "\n\nPart of the resulting array:" << endl;

for (int i = 0; i < n; i++) {

cout << arr4[i] << " ";

}

cout << endl;

cout << "\nAdditional memory used: " << memory << " bytes." << endl;

cout << "Time taken by program is : ";

cout << double(endTime - startTime) / CLOCKS\_PER\_SEC;

cout << " sec " << endl;

cout << "Number of assignments: " << cntMain << endl;

cout << "Number of auxiliary assignments: " << cntAux << endl;

cout << "----------------------------------------------------------------------" << endl;

startTime = 0;

endTime = 0;

cout << "2. Sorting an ordered array" << endl;

cout << "Part of the initial array:" << endl;

for (int i = 0; i < n; i++) {

cout << arr4[i] << " ";

}

startTime = clock();

bubbleSort(arr4, n4);

endTime = clock();

cout << "\n\nPart of the resulting array:" << endl;

for (int i = 0; i < n; i++) {

cout << arr4[i] << " ";

}

cout << endl;

cout << "\nAdditional memory used: " << memory << " bytes." << endl;

cout << "Time taken by program is : ";

cout << double(endTime - startTime) / CLOCKS\_PER\_SEC;

cout << " sec " << endl;

cout << "Number of assignments: " << cntMain << endl;

cout << "Number of auxiliary assignments: " << cntAux << endl;

cout << "----------------------------------------------------------------------" << endl;

startTime = 0;

endTime = 0;

cout << "3. Sorting an array sorted in reverse order" << endl;

cout << "Part of the initial array:" << endl;

s = sizeof(arr4) / sizeof(arr4[0]);

reverse(arr4, arr4 + s);

for (int i = 0; i < n; i++) {

cout << arr4[i] << " ";

}

startTime = clock();

bubbleSort(arr4, n4);

endTime = clock();

cout << "\n\nPart of the resulting array:" << endl;

for (int i = 0; i < n; i++) {

cout << arr4[i] << " ";

}

cout << endl;

cout << "\nAdditional memory used: " << memory << " bytes." << endl;

cout << "Time taken by program is : ";

cout << double(endTime - startTime) / CLOCKS\_PER\_SEC;

cout << " sec " << endl;

cout << "Number of assignments: " << cntMain << endl;

cout << "Number of auxiliary assignments: " << cntAux << endl;

cout << "----------------------------------------------------------------------" << endl;

file.close();

break;

case '0':

break;

default:

cout << "Invalide choice" << endl;

}

} while (menu != '0');

}

Shell.cpp:

#include <iostream>

#include <fstream>

#include <ctime>

#include <algorithm>

using namespace std;

const int n = 10, n1 = 10000, n2 = 30000, n3 = 70000, n4 = 100000;

unsigned long long int cntMain, cntAux;

int memory;

void shellSort(long arr[], int n) {

int h = 1, temp, i, j;

cntAux = 1;

cntMain = 0;

memory = 0;

while (h < n / 2) {

h = 2 \* h + 1;

cntAux++;

}

while (h >= 1) {

for (cntAux++, i = h; i < n; cntAux++, i++) {

for (cntAux++, j = i; j >= h && arr[j] < arr[j - h]; cntAux++, j -= h) {

temp = arr[j];

arr[j] = arr[j - h];

arr[j - h] = temp;

cntMain += 3;

}

}

h = (h - 1) / 2;

cntAux++;

}

memory = 4 \* sizeof(int);

}

int main() {

long arr1[n1], arr2[n2], arr3[n3], arr4[n4], s = 0;

char menu;

clock\_t start;

clock\_t end;

do {

ifstream file("C:\\Data Structures\\pr3\_1\\programm3\_1\\test\_numbers.txt");

cout << "----------------------------------------------------------------------" << endl;

cout << "1. Sort 10000 elements" << endl;

cout << "2. Sort 30000 elements" << endl;

cout << "3. Sort 70000 elements" << endl;

cout << "4. Sort 100000 elements" << endl;

cout << "0. Exit" << endl;

cout << "----------------------------------------------------------------------" << endl;

cin >> menu;

switch (menu) {

case '1':

for (int i = 0; i < n1; i++) {

file >> arr1[i];

}

start = 0;

end = 0;

cout << "----------------------------------------------------------------------" << endl;

cout << "1. Sorting an unordered array" << endl;

cout << "Part of the initial array:" << endl;

for (int i = 0; i < n; i++) {

cout << arr1[i] << " ";

}

start = clock();

shellSort(arr1, n1);

end = clock();

cout << "\n\nPart of the resulting array:" << endl;

for (int i = 0; i < n; i++) {

cout << arr1[i] << " ";

}

cout << endl;

cout << "\nAdditional memory used: " << memory << " bytes." << endl;

cout << "Time taken by program is : ";

cout << double(end - start) / CLOCKS\_PER\_SEC;

cout << " sec " << endl;

cout << "Number of main assignments: " << cntMain << endl;

cout << "Number of auxiliary assignments: " << cntAux << endl;

cout << "----------------------------------------------------------------------" << endl;

start = 0;

end = 0;

cout << "2. Sorting an ordered array" << endl;

cout << "Part of the initial array:" << endl;

for (int i = 0; i < n; i++) {

cout << arr1[i] << " ";

}

start = clock();

shellSort(arr1, n1);

end = clock();

cout << "\n\nPart of the resulting array:" << endl;

for (int i = 0; i < n; i++) {

cout << arr1[i] << " ";

}

cout << endl;

cout << "\nAdditional memory used: " << memory << " bytes." << endl;

cout << "Time taken by program is : ";

cout << double(end - start) / CLOCKS\_PER\_SEC;

cout << " sec " << endl;

cout << "Number of main assignments: " << cntMain << endl;

cout << "Number of auxiliary assignments: " << cntAux << endl;

cout << "----------------------------------------------------------------------" << endl;

start = 0;

end = 0;

cout << "3. Sorting an array sorted in reverse order" << endl;

cout << "Part of the initial array:" << endl;

s = sizeof(arr1) / sizeof(arr1[0]);

reverse(arr1, arr1 + s);

for (int i = 0; i < n; i++) {

cout << arr1[i] << " ";

}

start = clock();

shellSort(arr1, n1);

end = clock();

cout << "\n\nPart of the resulting array:" << endl;

for (int i = 0; i < n; i++) {

cout << arr1[i] << " ";

}

cout << endl;

cout << "\nAdditional memory used: " << memory << " bytes." << endl;

cout << "Time taken by program is : ";

cout << double(end - start) / CLOCKS\_PER\_SEC;

cout << " sec " << endl;

cout << "Number of main assignments: " << cntMain << endl;

cout << "Number of auxiliary assignments: " << cntAux << endl;

cout << "----------------------------------------------------------------------" << endl;

file.close();

break;

case '2':

for (int i = 0; i < n2; i++) {

file >> arr2[i];

}

start = 0;

end = 0;

cout << "----------------------------------------------------------------------" << endl;

cout << "1. Sorting an unordered array" << endl;

cout << "Part of the initial array:" << endl;

for (int i = 0; i < n; i++) {

cout << arr2[i] << " ";

}

start = clock();

shellSort(arr2, n2);

end = clock();

cout << "\n\nPart of the resulting array:" << endl;

for (int i = 0; i < n; i++) {

cout << arr2[i] << " ";

}

cout << endl;

cout << "\nAdditional memory used: " << memory << " bytes." << endl;

cout << "Time taken by program is : ";

cout << double(end - start) / CLOCKS\_PER\_SEC;

cout << " sec " << endl;

cout << "Number of main assignments: " << cntMain << endl;

cout << "Number of auxiliary assignments: " << cntAux << endl;

cout << "----------------------------------------------------------------------" << endl;

start = 0;

end = 0;

cout << "2. Sorting an ordered array" << endl;

cout << "Part of the initial array:" << endl;

for (int i = 0; i < n; i++) {

cout << arr2[i] << " ";

}

start = clock();

shellSort(arr2, n2);

end = clock();

cout << "\n\nPart of the resulting array:" << endl;

for (int i = 0; i < n; i++) {

cout << arr2[i] << " ";

}

cout << endl;

cout << "\nAdditional memory used: " << memory << " bytes." << endl;

cout << "Time taken by program is : ";

cout << double(end - start) / CLOCKS\_PER\_SEC;

cout << " sec " << endl;

cout << "Number of main assignments: " << cntMain << endl;

cout << "Number of auxiliary assignments: " << cntAux << endl;

cout << "----------------------------------------------------------------------" << endl;

start = 0;

end = 0;

cout << "3. Sorting an array sorted in reverse order" << endl;

cout << "Part of the initial array:" << endl;

s = sizeof(arr2) / sizeof(arr2[0]);

reverse(arr2, arr2 + s);

for (int i = 0; i < n; i++) {

cout << arr2[i] << " ";

}

start = clock();

shellSort(arr2, n2);

end = clock();

cout << "\n\nPart of the resulting array:" << endl;

for (int i = 0; i < n; i++) {

cout << arr2[i] << " ";

}

cout << endl;

cout << "\nAdditional memory used: " << memory << " bytes." << endl;

cout << "Time taken by program is : ";

cout << double(end - start) / CLOCKS\_PER\_SEC;

cout << " sec " << endl;

cout << "Number of main assignments: " << cntMain << endl;

cout << "Number of auxiliary assignments: " << cntAux << endl;

cout << "----------------------------------------------------------------------" << endl;

file.close();

break;

case '3':

for (long i = 0; i < n3; i++) {

file >> arr3[i];

}

start = 0;

end = 0;

cout << "----------------------------------------------------------------------" << endl;

cout << "1. Sorting an unordered array" << endl;

cout << "Part of the initial array:" << endl;

for (int i = 0; i < n; i++) {

cout << arr3[i] << " ";

}

start = clock();

shellSort(arr3, n3);

end = clock();

cout << "\n\nPart of the resulting array:" << endl;

for (int i = 0; i < n; i++) {

cout << arr3[i] << " ";

}

cout << endl;

cout << "\nAdditional memory used: " << memory << " bytes." << endl;

cout << "Time taken by program is : ";

cout << double(end - start) / CLOCKS\_PER\_SEC;

cout << " sec " << endl;

cout << "Number of main assignments: " << cntMain << endl;

cout << "Number of auxiliary assignments: " << cntAux << endl;

cout << "----------------------------------------------------------------------" << endl;

start = 0;

end = 0;

cout << "2. Sorting an ordered array" << endl;

cout << "Part of the initial array:" << endl;

for (int i = 0; i < n; i++) {

cout << arr3[i] << " ";

}

start = clock();

shellSort(arr3, n3);

end = clock();

cout << "\n\nPart of the resulting array:" << endl;

for (int i = 0; i < n; i++) {

cout << arr3[i] << " ";

}

cout << endl;

cout << "\nAdditional memory used: " << memory << " bytes." << endl;

cout << "Time taken by program is : ";

cout << double(end - start) / CLOCKS\_PER\_SEC;

cout << " sec " << endl;

cout << "Number of main assignments: " << cntMain << endl;

cout << "Number of auxiliary assignments: " << cntAux << endl;

cout << "----------------------------------------------------------------------" << endl;

start = 0;

end = 0;

cout << "3. Sorting an array sorted in reverse order" << endl;

cout << "Part of the initial array:" << endl;

s = sizeof(arr3) / sizeof(arr3[0]);

reverse(arr3, arr3 + s);

for (int i = 0; i < n; i++) {

cout << arr3[i] << " ";

}

start = clock();

shellSort(arr3, n3);

end = clock();

cout << "\n\nPart of the resulting array:" << endl;

for (int i = 0; i < n; i++) {

cout << arr3[i] << " ";

}

cout << endl;

cout << "\nAdditional memory used: " << memory << " bytes." << endl;

cout << "Time taken by program is : ";

cout << double(end - start) / CLOCKS\_PER\_SEC;

cout << " sec " << endl;

cout << "Number of main assignments: " << cntMain << endl;

cout << "Number of auxiliary assignments: " << cntAux << endl;

cout << "----------------------------------------------------------------------" << endl;

file.close();

break;

case '4':

for (long i = 0; i < n4; i++) {

file >> arr4[i];

}

start = 0;

end = 0;

cout << "----------------------------------------------------------------------" << endl;

cout << "1. Sorting an unordered array" << endl;

cout << "Part of the initial array:" << endl;

for (int i = 0; i < n; i++) {

cout << arr4[i] << " ";

}

start = clock();

shellSort(arr4, n4);

end = clock();

cout << "\n\nPart of the resulting array:" << endl;

for (int i = 0; i < n; i++) {

cout << arr4[i] << " ";

}

cout << endl;

cout << "\nAdditional memory used: " << memory << " bytes." << endl;

cout << "Time taken by program is : ";

cout << double(end - start) / CLOCKS\_PER\_SEC;

cout << " sec " << endl;

cout << "Number of main assignments: " << cntMain << endl;

cout << "Number of auxiliary assignments: " << cntAux << endl;

cout << "----------------------------------------------------------------------" << endl;

start = 0;

end = 0;

cout << "2. Sorting an ordered array" << endl;

cout << "Part of the initial array:" << endl;

for (int i = 0; i < n; i++) {

cout << arr4[i] << " ";

}

start = clock();

shellSort(arr4, n4);

end = clock();

cout << "\n\nPart of the resulting array:" << endl;

for (int i = 0; i < n; i++) {

cout << arr4[i] << " ";

}

cout << endl;

cout << "\nAdditional memory used: " << memory << " bytes." << endl;

cout << "Time taken by program is : ";

cout << double(end - start) / CLOCKS\_PER\_SEC;

cout << " sec " << endl;

cout << "Number of main assignments: " << cntMain << endl;

cout << "Number of auxiliary assignments: " << cntAux << endl;

cout << "----------------------------------------------------------------------" << endl;

start = 0;

end = 0;

cout << "3. Sorting an array sorted in reverse order" << endl;

cout << "Part of the initial array:" << endl;

s = sizeof(arr4) / sizeof(arr4[0]);

reverse(arr4, arr4 + s);

for (int i = 0; i < n; i++) {

cout << arr4[i] << " ";

}

start = clock();

shellSort(arr4, n4);

end = clock();

cout << "\n\nPart of the resulting array:" << endl;

for (int i = 0; i < n; i++) {

cout << arr4[i] << " ";

}

cout << endl;

cout << "\nAdditional memory used: " << memory << " bytes." << endl;

cout << "Time taken by program is : ";

cout << double(end - start) / CLOCKS\_PER\_SEC;

cout << " sec " << endl;

cout << "Number of main assignments: " << cntMain << endl;

cout << "Number of auxiliary assignments: " << cntAux << endl;

cout << "----------------------------------------------------------------------" << endl;

file.close();

break;

case '0':

break;

default:

cout << "Invalide choice" << endl;

}

} while (menu != '0');

}

Fibonacci.cpp:

#include <iostream>

#include <fstream>

#include <ctime>

#include <algorithm>

using namespace std;

const int n = 10, n1 = 10000, n2 = 30000, n3 = 70000, n4 = 100000;

unsigned long long int cntMain, cntAux;

int memory;

void fibonacciSort(long arr[], int n) {

memory = 0;

cntMain = 0;

int fib1 = 0, fib2 = 1, fib3 = 1, i, j, temp;

cntAux = 3;

while (fib3 < n) {

fib1 = fib2;

fib2 = fib3;

fib3 = fib1 + fib2;

cntAux += 3;

}

while (fib1 > 0) {

for (cntAux++, i = fib3; i < n; cntAux++, i++) {

for (cntAux++, j = i; j >= fib3 && arr[j - fib3] > arr[j]; cntAux++, j -= fib3) {

temp = arr[j];

arr[j] = arr[j - fib3];

arr[j - fib3] = temp;

cntMain += 3

;

}

}

fib3 = fib2;

fib2 = fib1;

fib1 = fib3 - fib2;

cntAux += 3;

}

memory = 6 \* sizeof(int);

}

int main() {

long arr1[n1], arr2[n2], arr3[n3], arr4[n4], s = 0;

char menu;

clock\_t start;

clock\_t end;

do {

ifstream file("C:\\Data Structures\\pr3\_1\\programm3\_1\\test\_numbers.txt");

cout << "----------------------------------------------------------------------" << endl;

cout << "1. Sort 10000 elements" << endl;

cout << "2. Sort 30000 elements" << endl;

cout << "3. Sort 70000 elements" << endl;

cout << "4. Sort 100000 elements" << endl;

cout << "0. Exit" << endl;

cout << "----------------------------------------------------------------------" << endl;

cin >> menu;

switch (menu) {

case '1':

for (int i = 0; i < n1; i++) {

file >> arr1[i];

}

start = 0;

end = 0;

cout << "----------------------------------------------------------------------" << endl;

cout << "1. Sorting an unordered array" << endl;

cout << "Part of the initial array:" << endl;

for (int i = 0; i < n; i++) {

cout << arr1[i] << " ";

}

start = clock();

fibonacciSort(arr1, n1);

end = clock();

cout << "\n\nPart of the resulting array:" << endl;

for (int i = 0; i < n; i++) {

cout << arr1[i] << " ";

}

cout << endl;

cout << "\nAdditional memory used: " << memory << " bytes." << endl;

cout << "Time taken by program is : ";

cout << double(end - start) / CLOCKS\_PER\_SEC;

cout << " sec " << endl;

cout << "Number of main assignments: " << cntMain << endl;

cout << "Number of auxiliary assignments: " << cntAux << endl;

cout << "----------------------------------------------------------------------" << endl;

start = 0;

end = 0;

cout << "2. Sorting an ordered array" << endl;

cout << "Part of the initial array:" << endl;

for (int i = 0; i < n; i++) {

cout << arr1[i] << " ";

}

start = clock();

fibonacciSort(arr1, n1);

end = clock();

cout << "\n\nPart of the resulting array:" << endl;

for (int i = 0; i < n; i++) {

cout << arr1[i] << " ";

}

cout << endl;

cout << "\nAdditional memory used: " << memory << " bytes." << endl;

cout << "Time taken by program is : ";

cout << double(end - start) / CLOCKS\_PER\_SEC;

cout << " sec " << endl;

cout << "Number of main assignments: " << cntMain << endl;

cout << "Number of auxiliary assignments: " << cntAux << endl;

cout << "----------------------------------------------------------------------" << endl;

start = 0;

end = 0;

cout << "3. Sorting an array sorted in reverse order" << endl;

cout << "Part of the initial array:" << endl;

s = sizeof(arr1) / sizeof(arr1[0]);

reverse(arr1, arr1 + s);

for (int i = 0; i < n; i++) {

cout << arr1[i] << " ";

}

start = clock();

fibonacciSort(arr1, n1);

end = clock();

cout << "\n\nPart of the resulting array:" << endl;

for (int i = 0; i < n; i++) {

cout << arr1[i] << " ";

}

cout << endl;

cout << "\nAdditional memory used: " << memory << " bytes." << endl;

cout << "Time taken by program is : ";

cout << double(end - start) / CLOCKS\_PER\_SEC;

cout << " sec " << endl;

cout << "Number of main assignments: " << cntMain << endl;

cout << "Number of auxiliary assignments: " << cntAux << endl;

cout << "----------------------------------------------------------------------" << endl;

file.close();

break;

case '2':

for (int i = 0; i < n2; i++) {

file >> arr2[i];

}

start = 0;

end = 0;

cout << "----------------------------------------------------------------------" << endl;

cout << "1. Sorting an unordered array" << endl;

cout << "Part of the initial array:" << endl;

for (int i = 0; i < n; i++) {

cout << arr2[i] << " ";

}

start = clock();

fibonacciSort(arr2, n2);

end = clock();

cout << "\n\nPart of the resulting array:" << endl;

for (int i = 0; i < n; i++) {

cout << arr2[i] << " ";

}

cout << endl;

cout << "\nAdditional memory used: " << memory << " bytes." << endl;

cout << "Time taken by program is : ";

cout << double(end - start) / CLOCKS\_PER\_SEC;

cout << " sec " << endl;

cout << "Number of main assignments: " << cntMain << endl;

cout << "Number of auxiliary assignments: " << cntAux << endl;

cout << "----------------------------------------------------------------------" << endl;

start = 0;

end = 0;

cout << "2. Sorting an ordered array" << endl;

cout << "Part of the initial array:" << endl;

for (int i = 0; i < n; i++) {

cout << arr2[i] << " ";

}

start = clock();

fibonacciSort(arr2, n2);

end = clock();

cout << "\n\nPart of the resulting array:" << endl;

for (int i = 0; i < n; i++) {

cout << arr2[i] << " ";

}

cout << endl;

cout << "\nAdditional memory used: " << memory << " bytes." << endl;

cout << "Time taken by program is : ";

cout << double(end - start) / CLOCKS\_PER\_SEC;

cout << " sec " << endl;

cout << "Number of main assignments: " << cntMain << endl;

cout << "Number of auxiliary assignments: " << cntAux << endl;

cout << "----------------------------------------------------------------------" << endl;

start = 0;

end = 0;

cout << "3. Sorting an array sorted in reverse order" << endl;

cout << "Part of the initial array:" << endl;

s = sizeof(arr2) / sizeof(arr2[0]);

reverse(arr2, arr2 + s);

for (int i = 0; i < n; i++) {

cout << arr2[i] << " ";

}

start = clock();

fibonacciSort(arr2, n2);

end = clock();

cout << "\n\nPart of the resulting array:" << endl;

for (int i = 0; i < n; i++) {

cout << arr2[i] << " ";

}

cout << endl;

cout << "\nAdditional memory used: " << memory << " bytes." << endl;

cout << "Time taken by program is : ";

cout << double(end - start) / CLOCKS\_PER\_SEC;

cout << " sec " << endl;

cout << "Number of main assignments: " << cntMain << endl;

cout << "Number of auxiliary assignments: " << cntAux << endl;

cout << "----------------------------------------------------------------------" << endl;

file.close();

break;

case '3':

for (long i = 0; i < n3; i++) {

file >> arr3[i];

}

start = 0;

end = 0;

cout << "----------------------------------------------------------------------" << endl;

cout << "1. Sorting an unordered array" << endl;

cout << "Part of the initial array:" << endl;

for (int i = 0; i < n; i++) {

cout << arr3[i] << " ";

}

start = clock();

fibonacciSort(arr3, n3);

end = clock();

cout << "\n\nPart of the resulting array:" << endl;

for (int i = 0; i < n; i++) {

cout << arr3[i] << " ";

}

cout << endl;

cout << "\nAdditional memory used: " << memory << " bytes." << endl;

cout << "Time taken by program is : ";

cout << double(end - start) / CLOCKS\_PER\_SEC;

cout << " sec " << endl;

cout << "Number of main assignments: " << cntMain << endl;

cout << "Number of auxiliary assignments: " << cntAux << endl;

cout << "----------------------------------------------------------------------" << endl;

start = 0;

end = 0;

cout << "2. Sorting an ordered array" << endl;

cout << "Part of the initial array:" << endl;

for (int i = 0; i < n; i++) {

cout << arr3[i] << " ";

}

start = clock();

fibonacciSort(arr3, n3);

end = clock();

cout << "\n\nPart of the resulting array:" << endl;

for (int i = 0; i < n; i++) {

cout << arr3[i] << " ";

}

cout << endl;

cout << "\nAdditional memory used: " << memory << " bytes." << endl;

cout << "Time taken by program is : ";

cout << double(end - start) / CLOCKS\_PER\_SEC;

cout << " sec " << endl;

cout << "Number of main assignments: " << cntMain << endl;

cout << "Number of auxiliary assignments: " << cntAux << endl;

cout << "----------------------------------------------------------------------" << endl;

start = 0;

end = 0;

cout << "3. Sorting an array sorted in reverse order" << endl;

cout << "Part of the initial array:" << endl;

s = sizeof(arr3) / sizeof(arr3[0]);

reverse(arr3, arr3 + s);

for (int i = 0; i < n; i++) {

cout << arr3[i] << " ";

}

start = clock();

fibonacciSort(arr3, n3);

end = clock();

cout << "\n\nPart of the resulting array:" << endl;

for (int i = 0; i < n; i++) {

cout << arr3[i] << " ";

}

cout << endl;

cout << "\nAdditional memory used: " << memory << " bytes." << endl;

cout << "Time taken by program is : ";

cout << double(end - start) / CLOCKS\_PER\_SEC;

cout << " sec " << endl;

cout << "Number of main assignments: " << cntMain << endl;

cout << "Number of auxiliary assignments: " << cntAux << endl;

cout << "----------------------------------------------------------------------" << endl;

file.close();

break;

case '4':

for (long i = 0; i < n4; i++) {

file >> arr4[i];

}

start = 0;

end = 0;

cout << "----------------------------------------------------------------------" << endl;

cout << "1. Sorting an unordered array" << endl;

cout << "Part of the initial array:" << endl;

for (int i = 0; i < n; i++) {

cout << arr4[i] << " ";

}

start = clock();

fibonacciSort(arr4, n4);

end = clock();

cout << "\n\nPart of the resulting array:" << endl;

for (int i = 0; i < n; i++) {

cout << arr4[i] << " ";

}

cout << endl;

cout << "\nAdditional memory used: " << memory << " bytes." << endl;

cout << "Time taken by program is : ";

cout << double(end - start) / CLOCKS\_PER\_SEC;

cout << " sec " << endl;

cout << "Number of main assignments: " << cntMain << endl;

cout << "Number of auxiliary assignments: " << cntAux << endl;

cout << "----------------------------------------------------------------------" << endl;

start = 0;

end = 0;

cout << "2. Sorting an ordered array" << endl;

cout << "Part of the initial array:" << endl;

for (int i = 0; i < n; i++) {

cout << arr4[i] << " ";

}

start = clock();

fibonacciSort(arr4, n4);

end = clock();

cout << "\n\nPart of the resulting array:" << endl;

for (int i = 0; i < n; i++) {

cout << arr4[i] << " ";

}

cout << endl;

cout << "\nAdditional memory used: " << memory << " bytes." << endl;

cout << "Time taken by program is : ";

cout << double(end - start) / CLOCKS\_PER\_SEC;

cout << " sec " << endl;

cout << "Number of main assignments: " << cntMain << endl;

cout << "Number of auxiliary assignments: " << cntAux << endl;

cout << "----------------------------------------------------------------------" << endl;

start = 0;

end = 0;

cout << "3. Sorting an array sorted in reverse order" << endl;

cout << "Part of the initial array:" << endl;

s = sizeof(arr4) / sizeof(arr4[0]);

reverse(arr4, arr4 + s);

for (int i = 0; i < n; i++) {

cout << arr4[i] << " ";

}

start = clock();

fibonacciSort(arr4, n4);

end = clock();

cout << "\n\nPart of the resulting array:" << endl;

for (int i = 0; i < n; i++) {

cout << arr4[i] << " ";

}

cout << endl;

cout << "\nAdditional memory used: " << memory << " bytes." << endl;

cout << "Time taken by program is : ";

cout << double(end - start) / CLOCKS\_PER\_SEC;

cout << " sec " << endl;

cout << "Number of main assignments: " << cntMain << endl;

cout << "Number of auxiliary assignments: " << cntAux << endl;

cout << "----------------------------------------------------------------------" << endl;

file.close();

break;

case '0':

break;

default:

cout << "Invalide choice" << endl;

}

} while (menu != '0');

}

Radix.cpp:

#include <iostream>

#include <fstream>

#include <ctime>

#include <algorithm>

using namespace std;

const int n = 10, n1 = 10000, n2 = 30000, n3 = 70000, n4 = 100000;

unsigned long int cntMain, cntAux;

long memory;

void countingSort(long arr[], int n, int exp) {

int \*output = new int[n], count[10] = { 0 };

int i;

for (cntAux++, i = 0; i < n; cntAux++, i++) {

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

cntAux++;

}

for (cntAux++, i = 1; i < 10; cntAux++, i++) {

count[i] += count[i - 1];

cntAux++;

}

for (cntAux++, i = n - 1; i >= 0; cntAux++, i--) {

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

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

cntAux += 2;

}

for (cntAux++, i = 0; i < n; cntAux++, i++) {

arr[i] = output[i];

cntMain++;

}

delete[] output;

}

void radixSort(long arr[], int n) {

cntAux = 0;

cntMain = 0;

int maxValue = arr[0], i, exp;

memory = 3 \* sizeof(int);

for (cntAux++, i = 1; i < n; cntAux++, i++) {

if (arr[i] > maxValue) {

maxValue = arr[i];

cntAux++;

}

}

for (cntAux++, exp = 1; maxValue / exp > 0;cntAux++, exp \*= 10) {

countingSort(arr, n, exp);

}

memory += (n + 11) \* sizeof(int);

}

int main() {

long arr1[n1], arr2[n2], arr3[n3], arr4[n4], s = 0;

char menu;

clock\_t start;

clock\_t end;

do {

ifstream file("C:\\Data Structures\\pr3\_1\\programm3\_1\\test\_numbers.txt");

cout << "----------------------------------------------------------------------" << endl;

cout << "1. Sort 10000 elements" << endl;

cout << "2. Sort 30000 elements" << endl;

cout << "3. Sort 70000 elements" << endl;

cout << "4. Sort 100000 elements" << endl;

cout << "0. Exit" << endl;

cout << "----------------------------------------------------------------------" << endl;

cin >> menu;

switch (menu) {

case '1':

for (int i = 0; i < n1; i++) {

file >> arr1[i];

}

start = 0;

end = 0;

cout << "----------------------------------------------------------------------" << endl;

cout << "1. Sorting an unordered array" << endl;

cout << "Part of the initial array:" << endl;

for (int i = 0; i < n; i++) {

cout << arr1[i] << " ";

}

start = clock();

radixSort(arr1, n1);

end = clock();

cout << "\n\nPart of the resulting array:" << endl;

for (int i = 0; i < n; i++) {

cout << arr1[i] << " ";

}

cout << endl;

cout << "\nAdditional memory used: " << memory << " bytes." << endl;

cout << "Time taken by program is : ";

cout << double(end - start) / CLOCKS\_PER\_SEC;

cout << " sec " << endl;

cout << "Number of main assignments: " << cntMain << endl;

cout << "Number of auxiliary assignments: " << cntAux << endl;

cout << "----------------------------------------------------------------------" << endl;

start = 0;

end = 0;

cout << "2. Sorting an ordered array" << endl;

cout << "Part of the initial array:" << endl;

for (int i = 0; i < n; i++) {

cout << arr1[i] << " ";

}

start = clock();

radixSort(arr1, n1);

end = clock();

cout << "\n\nPart of the resulting array:" << endl;

for (int i = 0; i < n; i++) {

cout << arr1[i] << " ";

}

cout << endl;

cout << "\nAdditional memory used: " << memory << " bytes." << endl;

cout << "Time taken by program is : ";

cout << double(end - start) / CLOCKS\_PER\_SEC;

cout << " sec " << endl;

cout << "Number of main assignments: " << cntMain << endl;

cout << "Number of auxiliary assignments: " << cntAux << endl;

cout << "----------------------------------------------------------------------" << endl;

start = 0;

end = 0;

cout << "3. Sorting an array sorted in reverse order" << endl;

cout << "Part of the initial array:" << endl;

s = sizeof(arr1) / sizeof(arr1[0]);

reverse(arr1, arr1 + s);

for (int i = 0; i < n; i++) {

cout << arr1[i] << " ";

}

start = clock();

radixSort(arr1, n1);

end = clock();

cout << "\n\nPart of the resulting array:" << endl;

for (int i = 0; i < n; i++) {

cout << arr1[i] << " ";

}

cout << endl;

cout << "\nAdditional memory used: " << memory << " bytes." << endl;

cout << "Time taken by program is : ";

cout << double(end - start) / CLOCKS\_PER\_SEC;

cout << " sec " << endl;

cout << "Number of main assignments: " << cntMain << endl;

cout << "Number of auxiliary assignments: " << cntAux << endl;

cout << "----------------------------------------------------------------------" << endl;

file.close();

break;

case '2':

for (int i = 0; i < n2; i++) {

file >> arr2[i];

}

start = 0;

end = 0;

cout << "----------------------------------------------------------------------" << endl;

cout << "1. Sorting an unordered array" << endl;

cout << "Part of the initial array:" << endl;

for (int i = 0; i < n; i++) {

cout << arr2[i] << " ";

}

start = clock();

radixSort(arr2, n2);

end = clock();

cout << "\n\nPart of the resulting array:" << endl;

for (int i = 0; i < n; i++) {

cout << arr2[i] << " ";

}

cout << endl;

cout << "\nAdditional memory used: " << memory << " bytes." << endl;

cout << "Time taken by program is : ";

cout << double(end - start) / CLOCKS\_PER\_SEC;

cout << " sec " << endl;

cout << "Number of main assignments: " << cntMain << endl;

cout << "Number of auxiliary assignments: " << cntAux << endl;

cout << "----------------------------------------------------------------------" << endl;

start = 0;

end = 0;

cout << "2. Sorting an ordered array" << endl;

cout << "Part of the initial array:" << endl;

for (int i = 0; i < n; i++) {

cout << arr2[i] << " ";

}

start = clock();

radixSort(arr2, n2);

end = clock();

cout << "\n\nPart of the resulting array:" << endl;

for (int i = 0; i < n; i++) {

cout << arr2[i] << " ";

}

cout << endl;

cout << "\nAdditional memory used: " << memory << " bytes." << endl;

cout << "Time taken by program is : ";

cout << double(end - start) / CLOCKS\_PER\_SEC;

cout << " sec " << endl;

cout << "Number of main assignments: " << cntMain << endl;

cout << "Number of auxiliary assignments: " << cntAux << endl;

cout << "----------------------------------------------------------------------" << endl;

start = 0;

end = 0;

cout << "3. Sorting an array sorted in reverse order" << endl;

cout << "Part of the initial array:" << endl;

s = sizeof(arr2) / sizeof(arr2[0]);

reverse(arr2, arr2 + s);

for (int i = 0; i < n; i++) {

cout << arr2[i] << " ";

}

start = clock();

radixSort(arr2, n2);

end = clock();

cout << "\n\nPart of the resulting array:" << endl;

for (int i = 0; i < n; i++) {

cout << arr2[i] << " ";

}

cout << endl;

cout << "\nAdditional memory used: " << memory << " bytes." << endl;

cout << "Time taken by program is : ";

cout << double(end - start) / CLOCKS\_PER\_SEC;

cout << " sec " << endl;

cout << "Number of main assignments: " << cntMain << endl;

cout << "Number of auxiliary assignments: " << cntAux << endl;

cout << "----------------------------------------------------------------------" << endl;

file.close();

break;

case '3':

for (long i = 0; i < n3; i++) {

file >> arr3[i];

}

start = 0;

end = 0;

cout << "----------------------------------------------------------------------" << endl;

cout << "1. Sorting an unordered array" << endl;

cout << "Part of the initial array:" << endl;

for (int i = 0; i < n; i++) {

cout << arr3[i] << " ";

}

start = clock();

radixSort(arr3, n3);

end = clock();

cout << "\n\nPart of the resulting array:" << endl;

for (int i = 0; i < n; i++) {

cout << arr3[i] << " ";

}

cout << endl;

cout << "\nAdditional memory used: " << memory << " bytes." << endl;

cout << "Time taken by program is : ";

cout << double(end - start) / CLOCKS\_PER\_SEC;

cout << " sec " << endl;

cout << "Number of main assignments: " << cntMain << endl;

cout << "Number of auxiliary assignments: " << cntAux << endl;

cout << "----------------------------------------------------------------------" << endl;

start = 0;

end = 0;

cout << "2. Sorting an ordered array" << endl;

cout << "Part of the initial array:" << endl;

for (int i = 0; i < n; i++) {

cout << arr3[i] << " ";

}

start = clock();

radixSort(arr3, n3);

end = clock();

cout << "\n\nPart of the resulting array:" << endl;

for (int i = 0; i < n; i++) {

cout << arr3[i] << " ";

}

cout << endl;

cout << "\nAdditional memory used: " << memory << " bytes." << endl;

cout << "Time taken by program is : ";

cout << double(end - start) / CLOCKS\_PER\_SEC;

cout << " sec " << endl;

cout << "Number of main assignments: " << cntMain << endl;

cout << "Number of auxiliary assignments: " << cntAux << endl;

cout << "----------------------------------------------------------------------" << endl;

start = 0;

end = 0;

cout << "3. Sorting an array sorted in reverse order" << endl;

cout << "Part of the initial array:" << endl;

s = sizeof(arr3) / sizeof(arr3[0]);

reverse(arr3, arr3 + s);

for (int i = 0; i < n; i++) {

cout << arr3[i] << " ";

}

start = clock();

radixSort(arr3, n3);

end = clock();

cout << "\n\nPart of the resulting array:" << endl;

for (int i = 0; i < n; i++) {

cout << arr3[i] << " ";

}

cout << endl;

cout << "\nAdditional memory used: " << memory << " bytes." << endl;

cout << "Time taken by program is : ";

cout << double(end - start) / CLOCKS\_PER\_SEC;

cout << " sec " << endl;

cout << "Number of main assignments: " << cntMain << endl;

cout << "Number of auxiliary assignments: " << cntAux << endl;

cout << "----------------------------------------------------------------------" << endl;

file.close();

break;

case '4':

for (long i = 0; i < n4; i++) {

file >> arr4[i];

}

start = 0;

end = 0;

cout << "----------------------------------------------------------------------" << endl;

cout << "1. Sorting an unordered array" << endl;

cout << "Part of the initial array:" << endl;

for (int i = 0; i < n; i++) {

cout << arr4[i] << " ";

}

start = clock();

radixSort(arr4, n4);

end = clock();

cout << "\n\nPart of the resulting array:" << endl;

for (int i = 0; i < n; i++) {

cout << arr4[i] << " ";

}

cout << endl;

cout << "\nAdditional memory used: " << memory << " bytes." << endl;

cout << "Time taken by program is : ";

cout << double(end - start) / CLOCKS\_PER\_SEC;

cout << " sec " << endl;

cout << "Number of main assignments: " << cntMain << endl;

cout << "Number of auxiliary assignments: " << cntAux << endl;

cout << "----------------------------------------------------------------------" << endl;

start = 0;

end = 0;

cout << "2. Sorting an ordered array" << endl;

cout << "Part of the initial array:" << endl;

for (int i = 0; i < n; i++) {

cout << arr4[i] << " ";

}

start = clock();

radixSort(arr4, n4);

end = clock();

cout << "\n\nPart of the resulting array:" << endl;

for (int i = 0; i < n; i++) {

cout << arr4[i] << " ";

}

cout << endl;

cout << "\nAdditional memory used: " << memory << " bytes." << endl;

cout << "Time taken by program is : ";

cout << double(end - start) / CLOCKS\_PER\_SEC;

cout << " sec " << endl;

cout << "Number of main assignments: " << cntMain << endl;

cout << "Number of auxiliary assignments: " << cntAux << endl;

cout << "----------------------------------------------------------------------" << endl;

start = 0;

end = 0;

cout << "3. Sorting an array sorted in reverse order" << endl;

cout << "Part of the initial array:" << endl;

s = sizeof(arr4) / sizeof(arr4[0]);

reverse(arr4, arr4 + s);

for (int i = 0; i < n; i++) {

cout << arr4[i] << " ";

}

start = clock();

radixSort(arr4, n4);

end = clock();

cout << "\n\nPart of the resulting array:" << endl;

for (int i = 0; i < n; i++) {

cout << arr4[i] << " ";

}

cout << endl;

cout << "\nAdditional memory used: " << memory << " bytes." << endl;

cout << "Time taken by program is : ";

cout << double(end - start) / CLOCKS\_PER\_SEC;

cout << " sec " << endl;

cout << "Number of main assignments: " << cntMain << endl;

cout << "Number of auxiliary assignments: " << cntAux << endl;

cout << "----------------------------------------------------------------------" << endl;

file.close();

break;

case '0':

break;

default:

cout << "Invalide choice" << endl;

}

} while (menu != '0');

}

Результаты работы программы:

Bubble.cpp:

При запуске программ нужно выбрать, массив из скольких элементов нужно отсортировать, что показано на рисунке 1.



Рисунок 1 – Главное меню

При нажатии на «1» произведется сортировка 10.000 элементов. Сначала сортируется неотсортированный массив, затем отсортированный массив, и в конце массив, отсортированный в обратном порядке. Выведется часть массива, поступившего на вход, затем часть отсортированного массива, количество использованной дополнительной памяти, время сортировки, количество основных операций и количество вспомогательных операций, что показано на рисунках 2, 3 и 4.

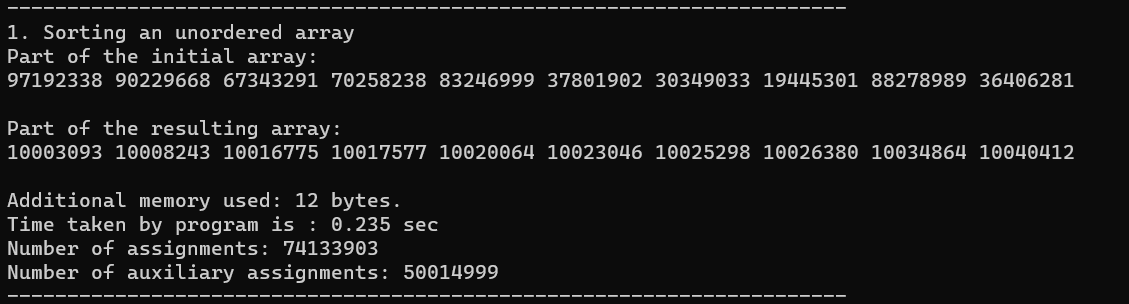


Рисунок 2 – Сортировка неупорядоченного массива

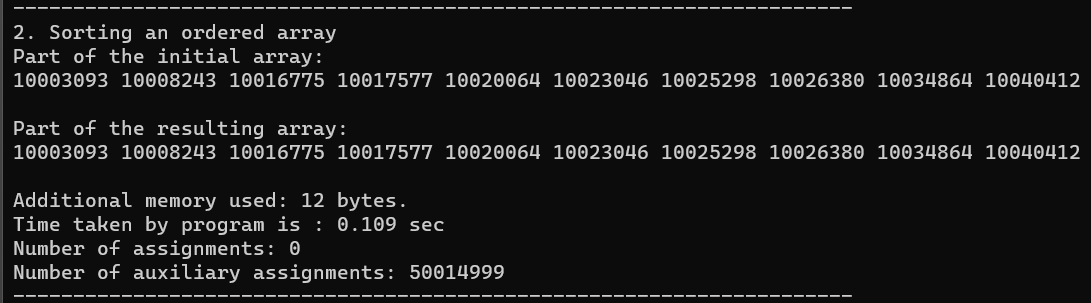


Рисунок 3 – Сортировка упорядоченного массива

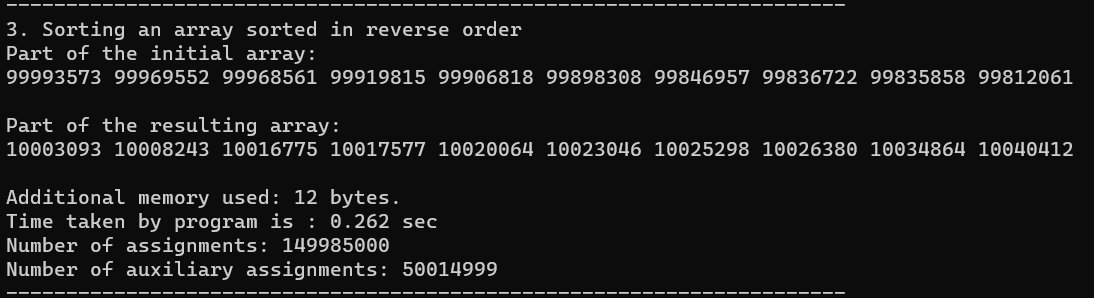


Рисунок 4 – Сортировка массива, упорядоченного в обратном порядке

При нажатии на «2», «3» и «4» произведется та же работа, только для 30.000 элементов, 70.000 элементов и 100.000 элементов соответственно.

Shell.cpp:

При запуске программ нужно выбрать, массив из скольких элементов нужно отсортировать, что показано на рисунке 5.



Рисунок 5 – Главное меню

При нажатии на «1» произведется сортировка 10.000 элементов. Сначала сортируется неотсортированный массив, затем отсортированный массив, и в конце массив, отсортированный в обратном порядке. Выведется часть массива, поступившего на вход, затем часть отсортированного массива, количество использованной дополнительной памяти, время сортировки, количество основных операций и количество вспомогательных операций, что показано на рисунках 6, 7 и 8.

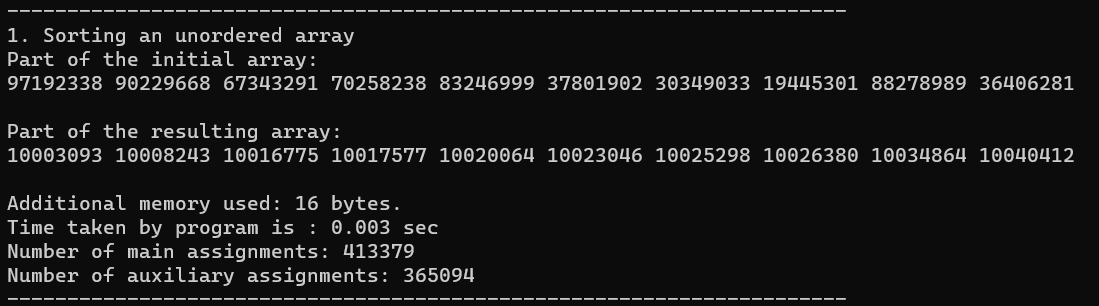


Рисунок 6 – Сортировка неупорядоченного массива

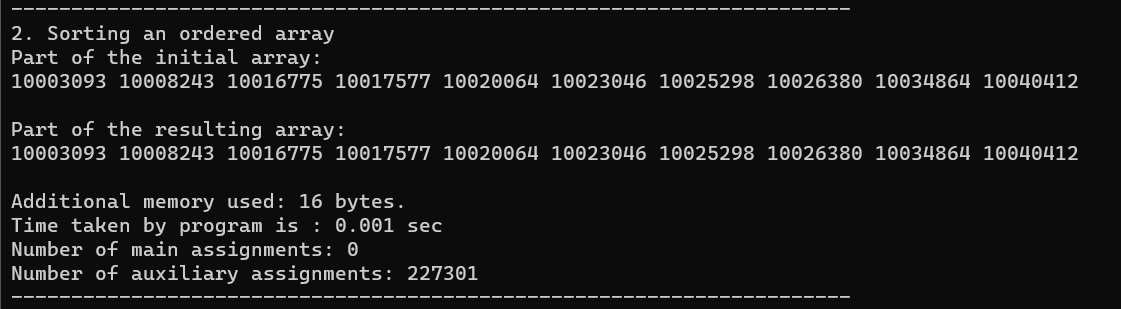


Рисунок 7 – Сортировка упорядоченного массива

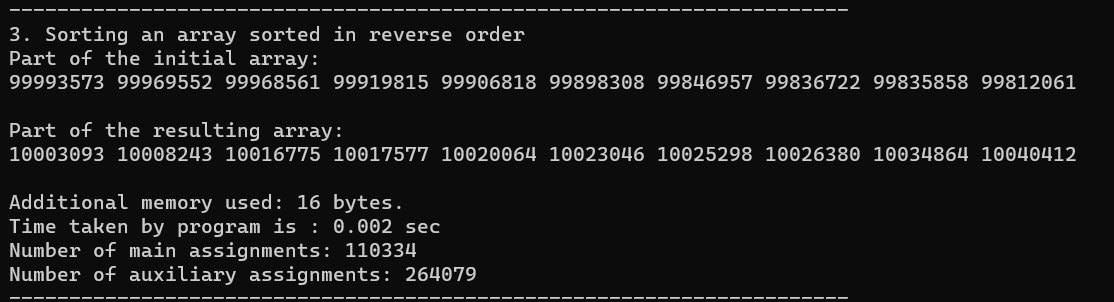


Рисунок 8 – Сортировка массива, упорядоченного в обратном порядке

При нажатии на «2», «3» и «4» произведется та же работа, только для 30.000 элементов, 70.000 элементов и 100.000 элементов соответственно.

Fibonacci.cpp:

При запуске программ нужно выбрать, массив из скольких элементов нужно отсортировать, что показано на рисунке 9.



Рисунок 9 – Главное меню

При нажатии на «1» произведется сортировка 10.000 элементов. Сначала сортируется неотсортированный массив, затем отсортированный массив, и в конце массив, отсортированный в обратном порядке. Выведется часть массива, поступившего на вход, затем часть отсортированного массива, количество использованной дополнительной памяти, время сортировки, количество основных операций и количество вспомогательных операций, что показано на рисунках 10, 11 и 12.

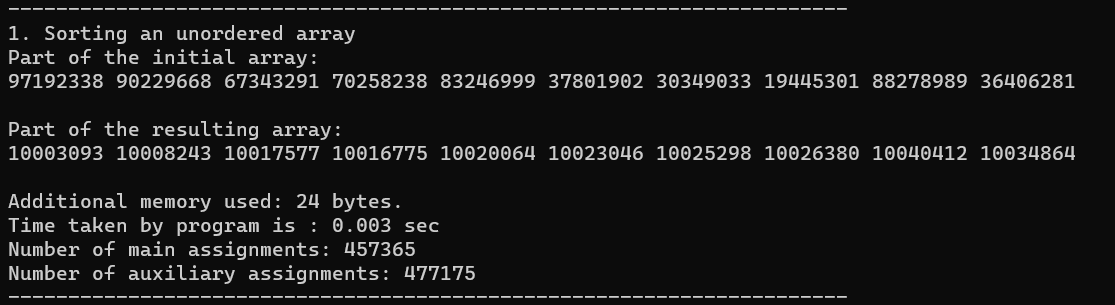


Рисунок 10 – Сортировка неупорядоченного массива

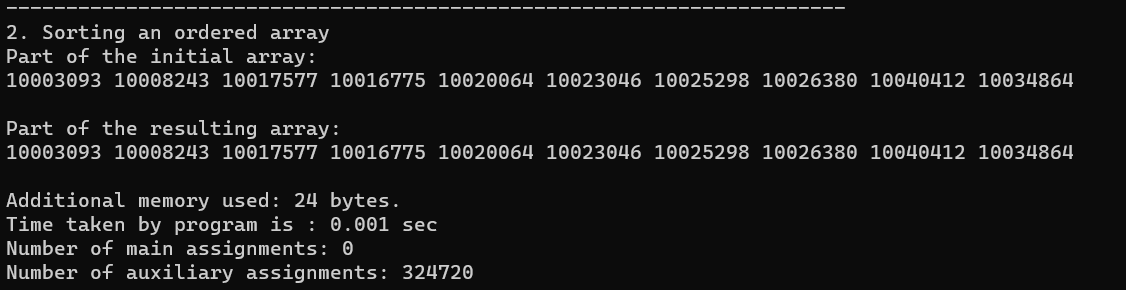


Рисунок 11 – Сортировка упорядоченного массива

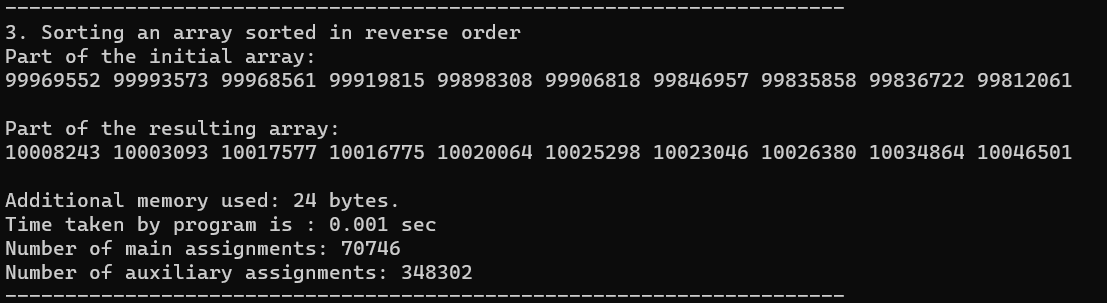


Рисунок 12 – Сортировка массива, упорядоченного в обратном порядке

При нажатии на «2», «3» и «4» произведется та же работа, только для 30.000 элементов, 70.000 элементов и 100.000 элементов соответственно.

Radix.cpp:

При запуске программ нужно выбрать, массив из скольких элементов нужно отсортировать, что показано на рисунке 13.



Рисунок 13 – Главное меню

При нажатии на «1» произведется сортировка 10.000 элементов. Сначала сортируется неотсортированный массив, затем отсортированный массив, и в конце массив, отсортированный в обратном порядке. Выведется часть массива, поступившего на вход, затем часть отсортированного массива, количество использованной дополнительной памяти, время сортировки, количество основных операций и количество вспомогательных операций, что показано на рисунках 14, 15 и 16.

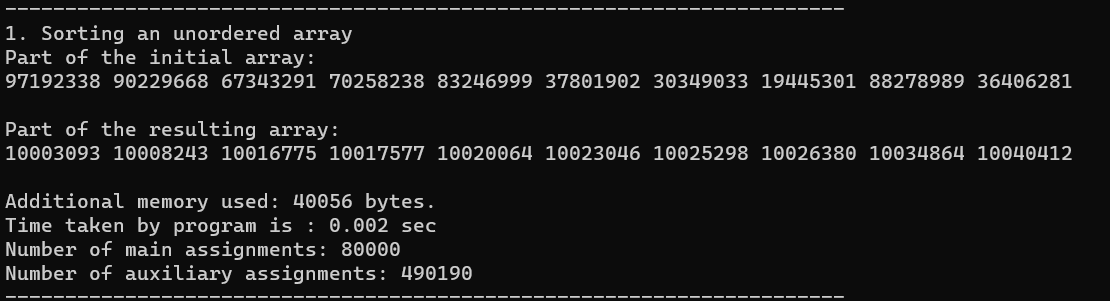


Рисунок 14 – Сортировка неупорядоченного массива

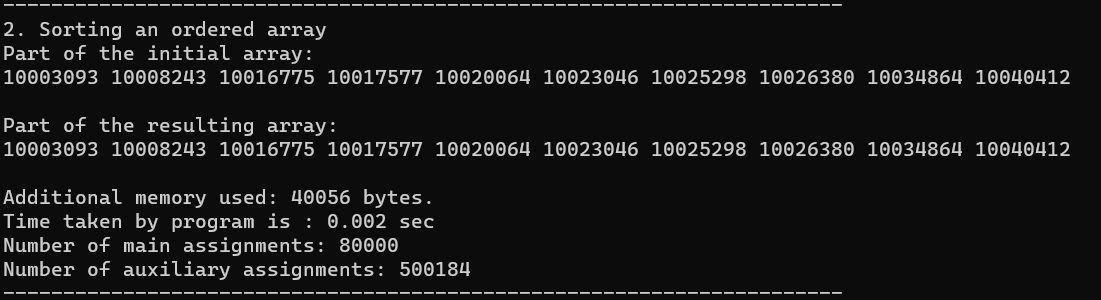


Рисунок 15 – Сортировка упорядоченного массива

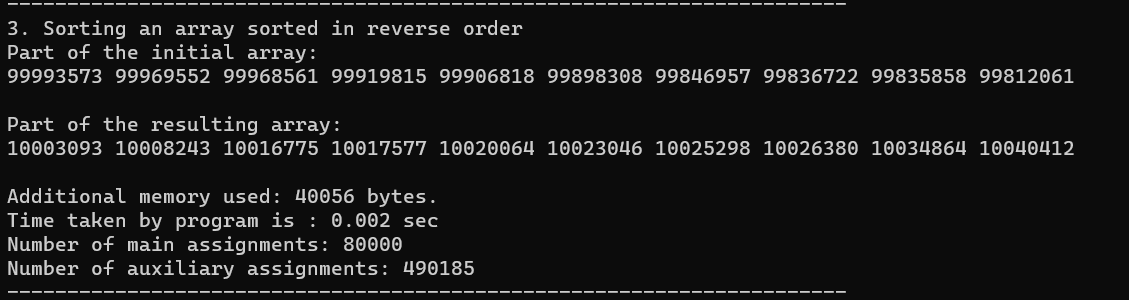


Рисунок 16 – Сортировка массива, упорядоченного в обратном порядке

При выборе несуществующего пункта меню выведется ошибка, что показано на рисунке 17.

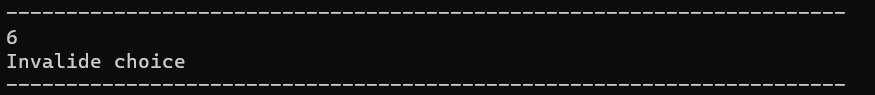


Рисунок 17 – Выбор несуществующего пункта меню

При нажатии на «0» работа программы прекратится.

Анализ полученных данных:

Полученные данные в сортировке пузырьком полностью соответствуют расчетным. Количество основных операций больше количества второстепенных, вне зависимости от количества элементов в массиве, кроме лучшего случая.

Полученные данные в сортировке Шелла отличаются от расчетных с небольшой погрешностью из-за неучтенного коэффициента. В лучшем случае основные операции не совершаются вовсе, в среднем случае количество второстепенных операций превышает количество основных, а в худшем случае наоборот, количество основных операций превышает количество второстепенных, независимо от размера массива.

Полученные данные в сортировке Шелла с шагом, установленным числами Фибоначчи, отличаются от расчетных с небольшой погрешностью из-за неучтенного коэффициента. В лучшем случае основные операции не совершаются вовсе, в среднем случае количество второстепенных операций превышает количество основных, а в худшем случае наоборот, количество основных операций превышает количество второстепенных, независимо от размера массива.

Полученные данные в поразрядной сортировке отличаются от расчетных с небольшой погрешностью из-за неучтенного коэффициента. Количество основных операций меньше количества второстепенных, вне зависимости от размера массива.

По трудоемкости:

В худшем случае самая оптимальная сортировка – поразрядная, так как используется наименьшее число присваиваний, но худший случай в этой сортировке отличается от худшего случая в других сортировках.

В среднем случае самая оптимальная сортировка – поразрядная, так как использует наименьшее число присваиваний, но средний случай в этой сортировке отличается от среднего случая в других сортировках.

В лучшем случае самые оптимальные сортировки – сортировка Шелла с заданным шагом и сортировка Шелла с шагом, заданным числами Фибоначчи, так как они используют одинаковое минимальное количество вспомогательных присваиваний, а основные присваивания не производятся вообще.

По времени:

В худшем случае поразрядная сортировка и сортировки Шелла работают одинаково.

В среднем случае поразрядная сортировка и сортировки Шелла работают одинаково.

В лучшем случае все сортировки работают примерно одинаково.

По пространственной сложности:

Лучшая пространственная сложность у сортировки пузырьком и сортировок Шелла, они используют минимальное количество дополнительной памяти .

Худшая пространственная сложность у поразрядной сортировки, так как использует количество дополнительно памяти , где *m* – мощность алфавита (массив для сохранения элементов массив в новом порядке и массив счетчиков).

Вывод: по результатам проведенного анализа самыми эффективными из рассмотренных алгоритмов по соотношению время-память являются алгоритм сортировкиШелла с заданным шагом и алгоритм сортировки Шелла с шагом, заданным числами Фибоначчи.