Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

«**Пермский национальный исследовательский политехнический университет»**

Кафедра «Информационные технологии и автоматизированные системы»

**ОТЧЕТ**

Дисциплина: «Основы алгоритмизации и программирования»

Тема: Внешние сортировки

Семестр 1

Выполнил работу

Студент группы РИС-22-1Б

Аксёнов Д.О.

Проверил

Доцент кафедры ИТАС

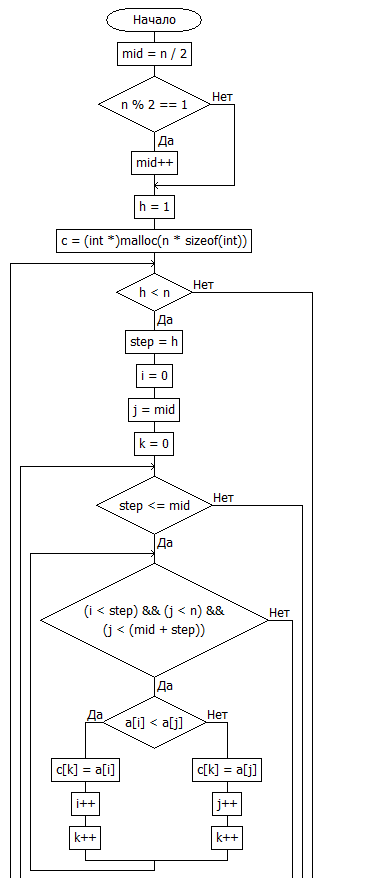
Полякова Ольга Андреевна

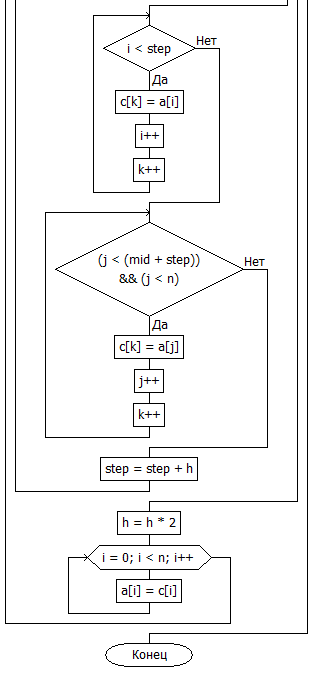
Г. Пермь-2023

**Постановка задачи**

Задача: реализовать внешние сортировки (двухфазная и естественная)

**Алгоритм программы (двух путевым слиянием)**

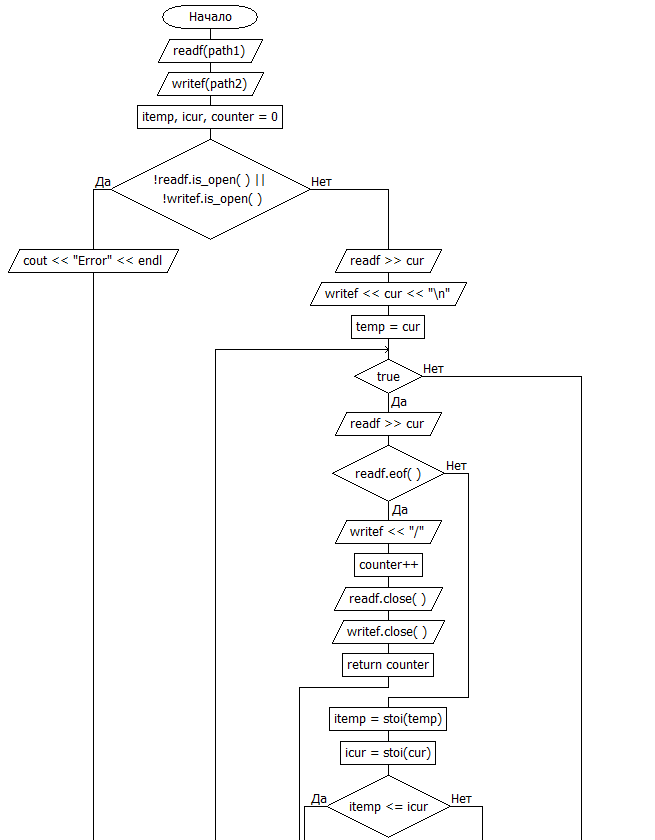


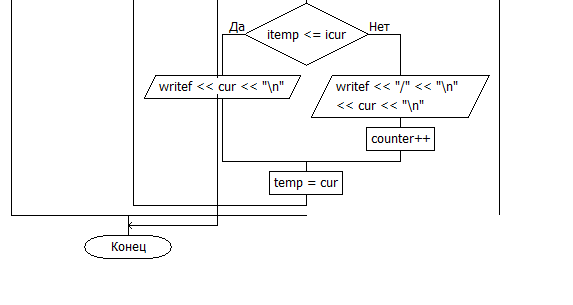


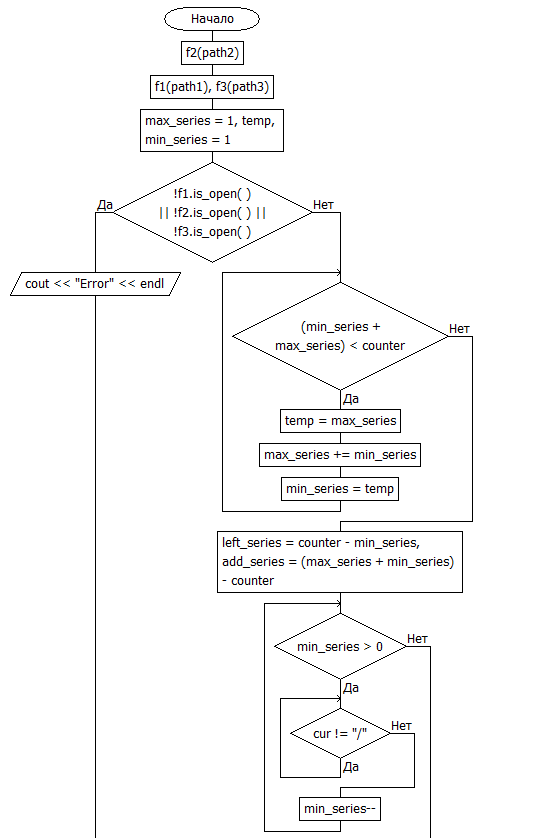
**Код программы (двух путевым слиянием)**

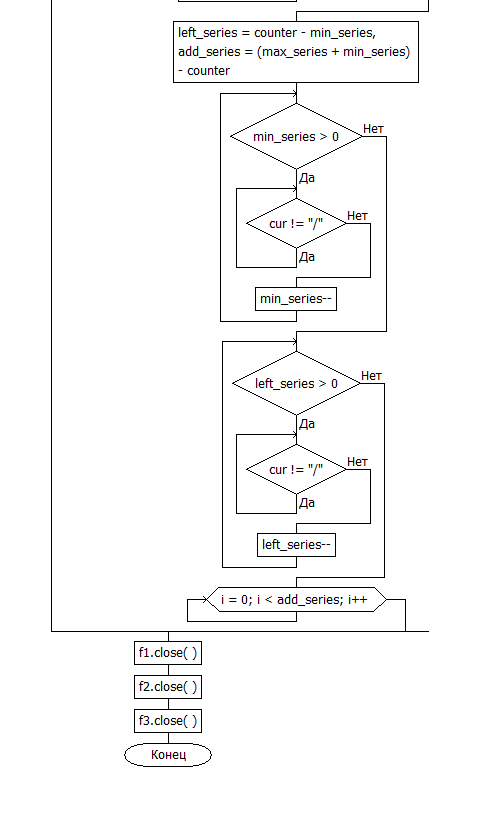
#include <stdio.h>  
#include <stdlib.h>  
// Функция сортировки двухпутевым слиянием  
void merge(int \*a, int n)  
{  
  int mid = n / 2; // находим середину сортируемой последовательности  
  if (n % 2 == 1)  
    mid++;  
  int h = 1; // шаг  
  // выделяем память под формируемую последовательность  
  int \*c = (int\*)malloc(n \* sizeof(int));  
  int step;  
  while (h < n)   
  {  
    step = h;  
    int i = 0;   // индекс первого пути  
    int j = mid; // индекс второго пути  
    int k = 0;   // индекс элемента в результирующей последовательности  
    while (step <= mid)   
    {  
      while ((i < step) && (j < n) && (j < (mid + step)))   
      { // пока не дошли до конца пути  
        // заполняем следующий элемент формируемой последовательности  
        // меньшим из двух просматриваемых  
        if (a[i] < a[j])    
        {  
          c[k] = a[i];  
          i++; k++;  
        }  
        else {  
          c[k] = a[j];  
          j++; k++;  
        }  
      }  
      while (i < step)   
      { // переписываем оставшиеся элементы первого пути (если второй кончился раньше)  
        c[k] = a[i];  
        i++; k++;  
      }  
      while ((j < (mid + step)) && (j<n))   
      {  // переписываем оставшиеся элементы второго пути (если первый кончился раньше)  
        c[k] = a[j];  
        j++; k++;  
      }  
      step = step + h; // переходим к следующему этапу  
    }  
    h = h \* 2;  
    // Переносим упорядоченную последовательность (промежуточный вариант) в исходный массив  
    for (i = 0; i<n; i++)  
      a[i] = c[i];  
  }  
}  
int main()   
{  
  int a[8];  
  // Заполнение массива случайными числами  
  for (int i = 0; i<8; i++)  
    a[i] = rand() % 20 - 10;  
  // Вывод элементов массива до сортировки  
  for (int i = 0; i<8; i++)  
    printf("%d ", a[i]);  
  printf("\n");  
  merge(a, 8); // вызов функции сортировки  
  // Вывод элементов массива после сортировки  
  for (int i = 0; i<8; i++)  
    printf("%d ", a[i]);  
  printf("\n");  
  getchar();  
  return 0;  
}

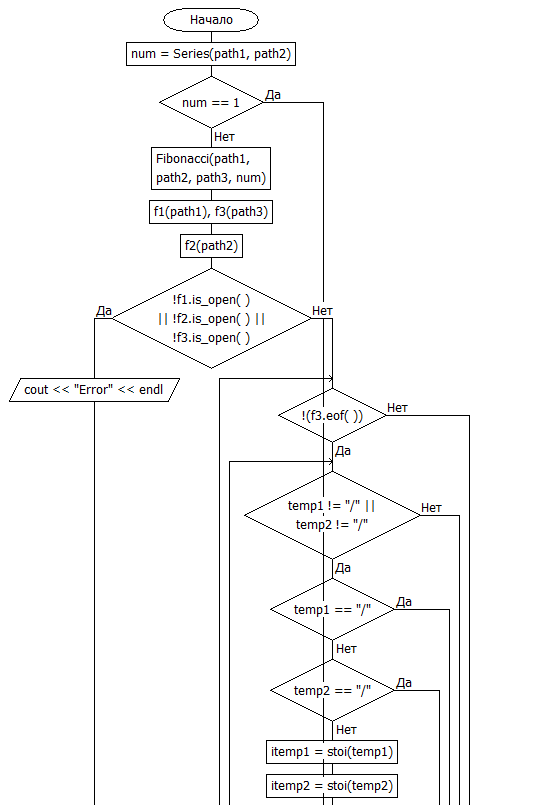
**Алгоритм программы (двухфазная)**

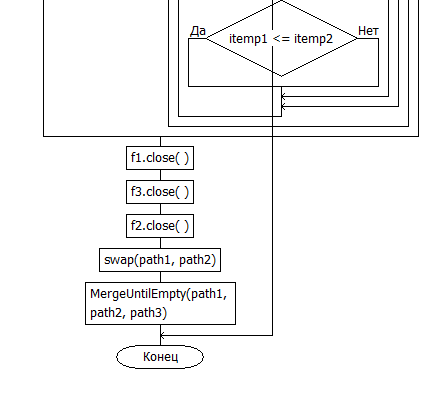












**Код программы(двухфазная)**

#include <iostream>

#include <fstream>

#include <string>

#include <stdio.h>

#include <Windows.h>

using namespace std;

int Series(string path1, string path2) {

ifstream readf(path1); //чтение из 1 файла

ofstream writef(path2); //запись со слешами во 2 файл

string temp, cur;

int itemp, icur, counter = 0;

if (!readf.is\_open() || !writef.is\_open()) { cout << "Error" << endl; }

else {

readf >> cur;

writef << cur << "\n";

temp = cur;

while (true) {

readf >> cur;

if (readf.eof()) {

writef << "/";

counter++;

readf.close();

writef.close();

return counter;

}

itemp = stoi(temp);

icur = stoi(cur);

if (itemp <= icur) writef << cur << "\n";

else {

writef << "/" << "\n" << cur << "\n";

counter++;

}

temp = cur;

}

}

}

void Fibonacci(string path1, string path2, string path3, int counter) {

ifstream f2(path2); //чтение из второго файла

ofstream f1(path1), f3(path3); //запись в 1 и 3 файл

string cur; int max\_series = 1, temp, min\_series = 1;

if (!f1.is\_open() || !f2.is\_open() || !f3.is\_open()) { cout << "Error" << endl; }

else {

while ((min\_series + max\_series) < counter) {

temp = max\_series;

max\_series += min\_series;

min\_series = temp;

}

f2 >> cur;

int left\_series = counter - min\_series, add\_series = (max\_series + min\_series) - counter;

while (min\_series > 0) {

while (cur != "/") {

f1 << cur << "\n";

f2 >> cur;

}

f1 << "/" << "\n";

f2 >> cur;

min\_series--;

}

while (left\_series > 0) {

while (cur != "/") {

f3 << cur << "\n";

f2 >> cur;

}

f3 << "/" << "\n";

f2 >> cur;

left\_series--;

}

for (int i = 0; i < add\_series; i++) f3 << "/" << "\n";

}

f1.close();

f2.close();

f3.close();

}

void MergeUntilEmpty(string path1, string path2, string path3) {

int num = Series(path1, path2);

if (num == 1) return;

Fibonacci(path1, path2, path3, num);

ifstream f1(path1), f3(path3);//чтение из 1 и 3 файла

ofstream f2(path2); //запись во второй файл

string temp1, temp2; int itemp1, itemp2;

if (!f1.is\_open() || !f2.is\_open() || !f3.is\_open()) { cout << "Error" << endl; }

else {

while (!(f3.eof())) {

f1 >> temp1;

f3 >> temp2;

while (temp1 != "/" || temp2 != "/") {

if (temp1 == "/") {

f2 << temp2 << "\n";

f3 >> temp2;

}

else if (temp2 == "/") {

f2 << temp1 << "\n";

f1 >> temp1;

}

else {

itemp1 = stoi(temp1);

itemp2 = stoi(temp2);

if (itemp1 <= itemp2) {

f2 << temp1 << "\n";

f1 >> temp1;

}

else {

f2 << temp2 << "\n";

f3 >> temp2;

}

}

}

}

}

f1.close(); f3.close(); f2.close();

swap(path1, path2);

MergeUntilEmpty(path1, path2, path3);

}

int main() {

SetConsoleCP(1251); SetConsoleOutputCP(1251);

//system("color F0");

srand(time(NULL));

int size, current;

string firstfile = "1.txt", secondfile = "2.txt", thirdfile = "3.txt", temp;

ofstream fir; ifstream res;

cout << "Кол-во чисел: ";

cin >> size;

cout << "Массив до сортировки:\n"; //заполнение файла с данными случайными значениями

fir.open(firstfile);

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

current = rand() % 100;

cout << current << " ";

fir << current << "\n";

}

fir.close();

cout << endl << endl;

MergeUntilEmpty(firstfile, secondfile, thirdfile);

cout << "Отсортированный массив:\n"; //вывод в консоль данных в результирующем файле

res.open(firstfile);

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

res >> temp;

cout << temp << " ";

}

res.close();

cout << endl;

return 0;

}

**Работа программ.**

