Министерство образования и науки Российской Федерации  
Федеральное‌ ‌государственное‌ ‌бюджетное‌ ‌образовательное‌ ‌учреждение‌

высшего‌ ‌образования‌

**«Пермский национальный исследовательский  
политехнический университет»**

Кафедра «Информационные технологии и автоматизированные системы»

**ОТЧЁТ**

**по лабораторной работе №16**

Дисциплина: «Основы теории алгоритмов и структуры данных»

Тема: Методы внешней сортировки: метод прямого слияния, сбалансированного слияния, метод многофазной сортировки

Вариант 19

Выполнил:

Студент группы РИС-20-1б

Ремянников Александр Владимирович

Проверила:

Доцент кафедры ИТАС

Полякова О. А.

**Пермь, 2021**

**Цель работы**

Получить опыт работы с внешними сортировками

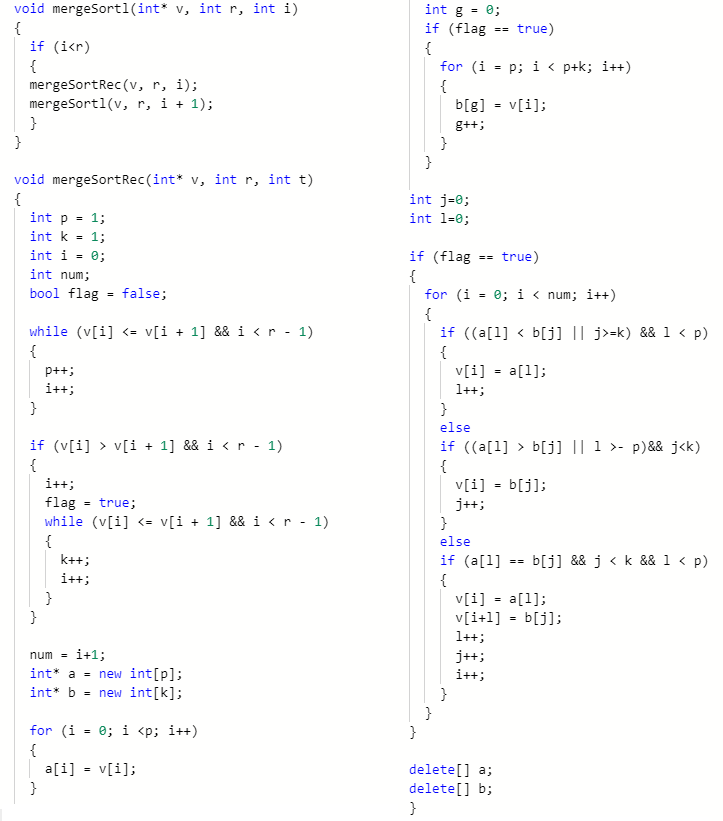
**Постановка задачи**

(Вариант 19) Выполнить сортировку внешними методами: Метод естественного слияния, сбалансированного слияния, многофазного слияния.

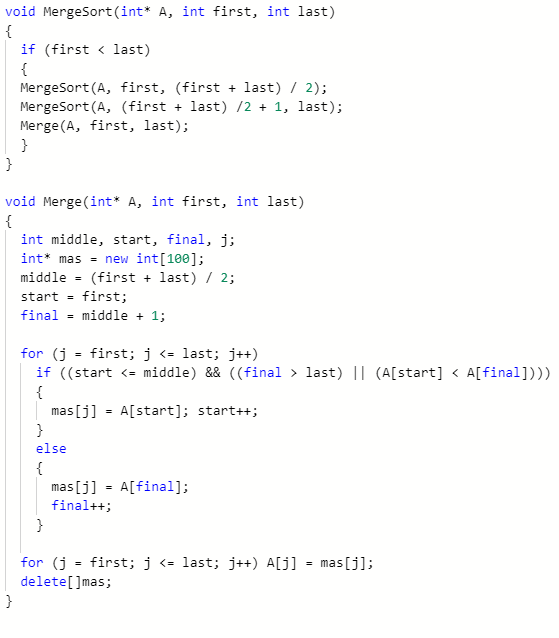
**Анализ задачи**

**1.** Для решения задачи необходимо:

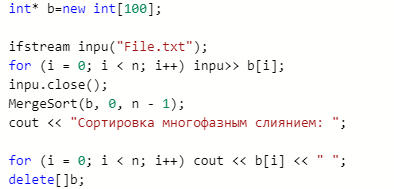
**1.1.** Реализовать код для выполнения сортировки методом естественного слияния.



**1.2.** Реализовать код для выполнения сортировки методом сбалансированного слияния.



**1.3.** Реализовать код для выполнения сортировки методом многофазного слияния.



**2.** В ходе работы были использованы следующие типы данных:

**2.1.** Для получения информации из файла используется класс данных ifstream.



**2.2.** Для хранения списка элементов используется динамический массив типа int.



**3.** Для решения задачи данные были представлены в следующем виде:

**3.1.** Данные из файла были записаны в динамический массив типа int.



**4.** Для операций ввода и вывода использовались следующие операторы и функции:

**4.1.** Операция ввода данных из файла производится с использованием оператора cin.

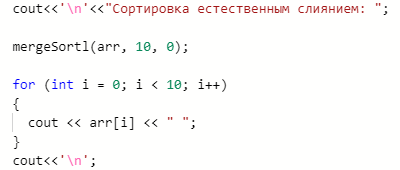


**4.2.** Для вывода отсортированного массива в консоль используется оператор cout.

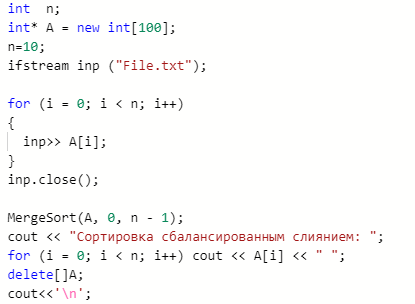


**5.** Поставленные задачи будут решены следующими действиями:

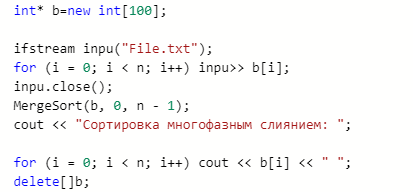
**5.1.** Будет использован код для сортировки методом естественного слияния и результат его работы будет выведен на экран.



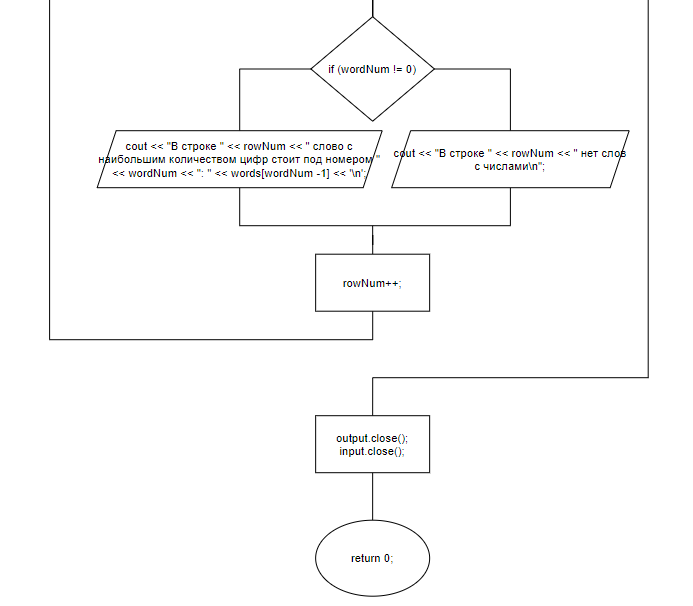
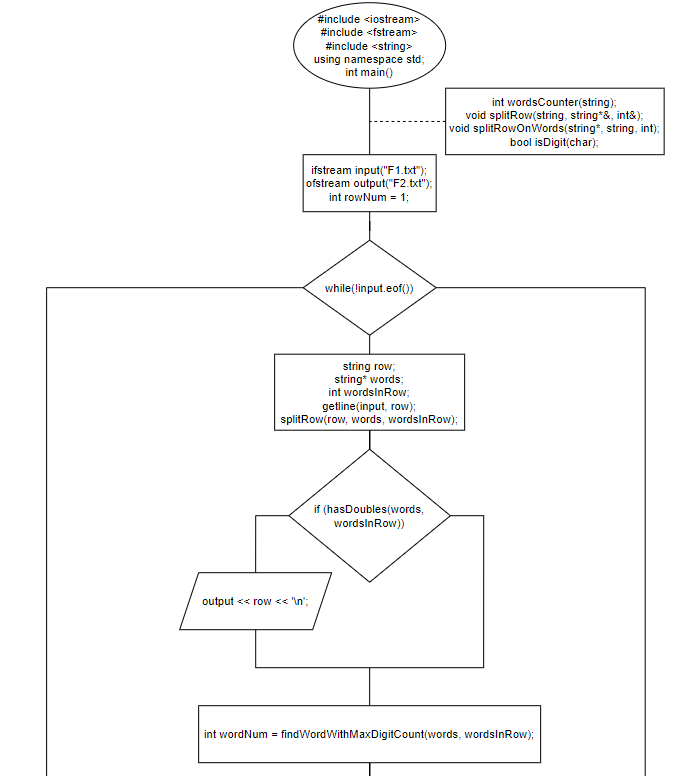
**5.2.** Будет использован код для сортировки методом сбалансированного слияния и результат его работы будет выведен на экран.

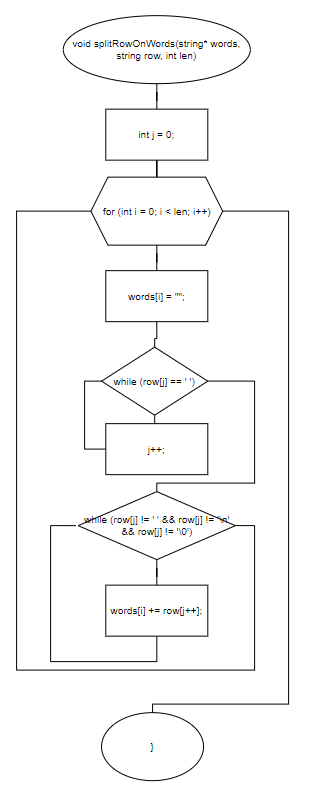
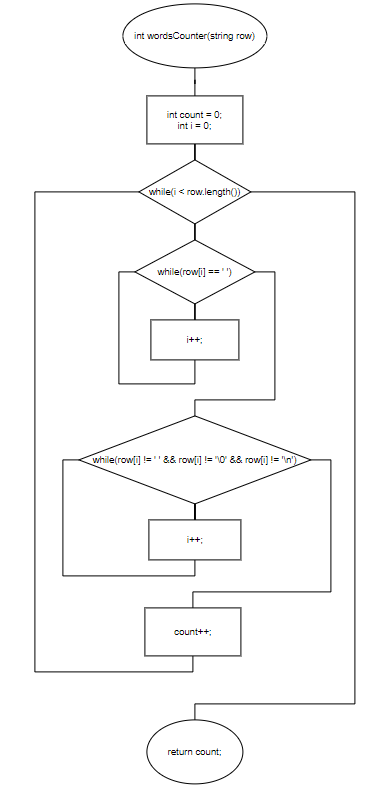


**5.3.** Будет использован код для сортировки методом многофазного слияния и результат его работы будет выведен на экран.



**Блок-схема программы**





**Решение**

#include <bits/stdc++.h>

using namespace std;

void mergeSortl(int\*, int, int);

void mergeSortRec(int\*, int, int);

void MergeSort(int\*, int, int);

void Merge(int\*, int, int);

int main()

{

int \*arr, i;

ifstream in ("File.txt");

setlocale(LC\_ALL, "PUS");

for (int i = 0; i < 10; i++)

{

in >> arr[i];

}

in.close();

cout<< "Ваш массив: ";

for (int i = 0; i < 10; i++)

{

cout << arr[i] << " ";

}

// Сортировка естественным слиянием

cout<<'\n'<<"Сортировка естественным слиянием: ";

mergeSortl(arr, 10, 0);

for (int i = 0; i < 10; i++)

{

cout << arr[i] << " ";

}

cout<<'\n';

//

// Cортировка сбалансированным слиянием

int n;

int\* A = new int[100];

n=10;

ifstream inp ("File.txt");

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

{

inp>> A[i];

}

inp.close();

MergeSort(A, 0, n - 1);

cout << "Cортировка сбалансированным слиянием: ";

for (i = 0; i < n; i++) cout << A[i] << " ";

delete[]A;

cout<<'\n';

//

// Сортировка многофазным слиянием

int\* b=new int[100];

ifstream inpu("File.txt");

for (i = 0; i < n; i++) inpu>> b[i];

inpu.close();

MergeSort(b, 0, n - 1);

cout << "Сортировка многофазным слиянием: ";

for (i = 0; i < n; i++) cout << b[i] << " ";

delete[]b;

//

}

// 1

void mergeSortl(int\* v, int r, int i)

{

if (i<r)

{

mergeSortRec(v, r, i);

mergeSortl(v, r, i + 1);

}

}

void mergeSortRec(int\* v, int r, int t)

{

int p = 1;

int k = 1;

int i = 0;

int num;

bool flag = false;

while (v[i] <= v[i + 1] && i < r - 1)

{

p++;

i++;

}

if (v[i] > v[i + 1] && i < r - 1)

{

i++;

flag = true;

while (v[i] <= v[i + 1] && i < r - 1)

{

k++;

i++;

}

}

num = i+1;

int\* a = new int[p];

int\* b = new int[k];

for (i = 0; i <p; i++)

{

a[i] = v[i];

}

int g = 0;

if (flag == true)

{

for (i = p; i < p+k; i++)

{

b[g] = v[i];

g++;

}

}

int j=0;

int l=0;

if (flag == true)

{

for (i = 0; i < num; i++)

{

if ((a[l] < b[j] || j>=k) && l < p)

{

v[i] = a[l];

l++;

}

else

if ((a[l] > b[j] || l >- p)&& j<k)

{

v[i] = b[j];

j++;

}

else

if (a[l] == b[j] && j < k && l < p)

{

v[i] = a[l];

v[i+l] = b[j];

l++;

j++;

i++;

}

}

}

delete[] a;

delete[] b;

}

// 2

void MergeSort(int\* A, int first, int last)

{

if (first < last)

{

MergeSort(A, first, (first + last) / 2);

MergeSort(A, (first + last) /2 + 1, last);

Merge(A, first, last);

}

}

void Merge(int\* A, int first, int last)

{

int middle, start, final, j;

int\* mas = new int[100];

middle = (first + last) / 2;

start = first;

final = middle + 1;

for (j = first; j <= last; j++)

if ((start <= middle) && ((final > last) || (A[start] < A[final])))

{

mas[j] = A[start]; start++;

}

else

{

mas[j] = A[final];

final++;

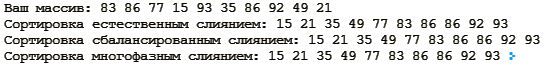
}

for (j = first; j <= last; j++) A[j] = mas[j];

delete[]mas;

}

**Скриншоты результатов работы программы**

****

**GitHub**

<https://github.com/sugarrrfqs/lab16/tree/main>