МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ им. Н. Э. Баумана

КАФЕДРА ПРОЕКТИРОВАНИЕ  И ТЕХНОЛОГИЯ ПРОИЗВОДСТВА ЭЛЕКТРОННОЙ АППАРАТУРЫ

Отчет о выполнении

практического задания №3

«Разработка и реализация алгоритма»

по курсу «Функциональная логика и теория алгоритмов»

Выполнили: студент гр. ИУ4-21б

Глухов Степан Андреевич

Проверил: д.т.н., профессор, профессор каф. ИУ-4

Л.А. Зинченко

Москва 2022

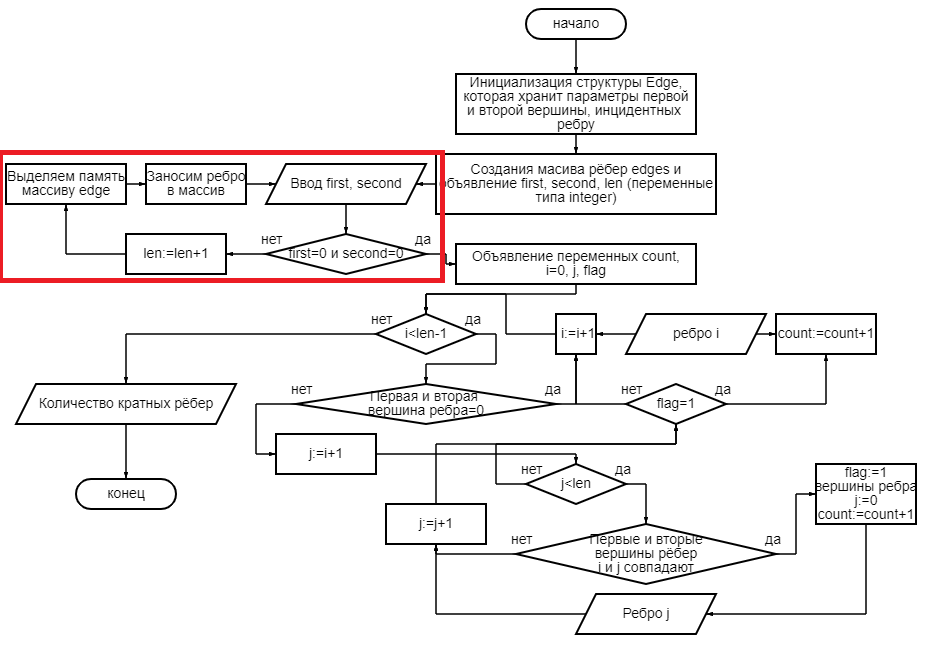
Цель: Разработать алгоритм, который находит и выводит все кратные рёбра, а также подсчитывает их общее количество. Реализовать его в виде программы на языке Си. Изучить временную сложность работы алгоритма.

1. Исходные данные:

Язык программирования – С.

2. Выполнение:

Данный алгоритм чем-то похож на сортировку пузырьком, в том смысле, что он пробегается по всем элементам массива несколько раз.



Красным квадратом отмечен алгоритма ввода рёбер. Далее начинает работу сам алгоритм.

Для начала выбираем самый первый элемент, а затем сравниваем со всеми остальными далее идущими элементами. Если элемент равен рассматриваемому, выводим его и зануляем. Затем выводим рассматриваемый элемент и меняем его на следующий. Нулевые элементы не берутся в качестве рассматриваемых.

#include <stdio.h>

#include <stdlib.h>

typedef struct Edge{

int firstTop, secondTop;

} Edge;

int main(){

Edge \*edges=NULL;

int first=0, second=0, len;

edges=(Edge\*)malloc(sizeof(Edge)\*len);

printf("Enter edges as '<First Top> <Second Top>'. Finish is '0 0'\n");

do{

scanf("%d %d", &first, &second);

edges=realloc(edges, sizeof(Edge)\*(++len));

edges[len-1].firstTop=first;

edges[len-1].secondTop=second;

}while(first!=0 || second!=0);

int count=0, i, j, flag;

for(i=0; i<len-1; i++){

if(edges[i].firstTop!=0 && edges[i].secondTop!=0){

for(j=i+1; j<len; j++)

if(edges[i].firstTop == edges[j].firstTop && edges[i].secondTop == edges[j].secondTop){

flag=1;

edges[j].firstTop=0;

edges[j].secondTop=0;

printf("%d, ", j+1);

count++;

}

if(flag){

flag=0;

printf("%d - multiple edges\n", i+1);

count++;

}

}

}

if(count>0)

printf("Count of multiple edges - %d", count);

else

printf("No multiple edges");

}

3. Результаты работы.

В результате был получен рабочий алгоритм, который выполняет свои функции и является достаточно эффективным. Это можно увидеть из графика:  
