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 №1

<Реализация очереди>

по курсу «Функциональная логика и теория алгоритмов»

Выполнили: студенты гр. ИУ4-33Б

Мадаминов Акромжон Тулкин угли

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

Терехов Владимир Владимирович

Москва 2020

Цель работы: написать программу на языке С, в которой реализована очередь.

1. Исходные данные:

Очередь-динамическая структура данных, состоящая из узлов, каждый из которых содержит значение и указатель на предыдущий узел.

2. Выполнение.

#include<stdio.h>

#include<stdlib.h>

#define MAX 100

struct queue{

int que[MAX];

int first;

int last;

};

void init(struct queue \*s){

s->first = 1;

s->last = 0;

return;

}

void add(struct queue \*s){

int element;

printf("New element: ");

scanf("%d", &element);

if(s->last < MAX-1) {

s->last++;

s->que[s->last] = element;

}

else

printf("The queue is crowded...\n");

return;

}

void del(struct queue \*s){

int element, i;

if(s->last < s->first) {

printf("The queue is empty...\n");

return;

}

element = s->que[s->first];

for(i = s->first; i < s->last; i++) {

s->que[i] = s->que[i+1];

}

s->last--;

return;

}

void show(struct queue \*s){

if(s->last < s->first){

printf("The queue is empty...\n");

return;

}

else{

int k;

for(k = s->first; k <= s->last; k++){

printf("%d ", s->que[k]);

}

printf("\n");

return;

}

}

int main(void){

struct queue \*s;

s = (struct queue \*)malloc(sizeof(struct queue));

init(s);

printf("1 - add an element\n");

printf("2 - delete first element\n");

printf("3 - show a queue\n");

printf("4 - exit program\n");

int a;

while(a!=4){

printf("Action: ");

scanf("%d", &a);

switch(a){

case 1:

add(s);

break;

case 2:

del(s);

break;

case 3:

show(s);

break;

case 4:

exit(0);

break;

default:

printf("Unknown operation...\n");

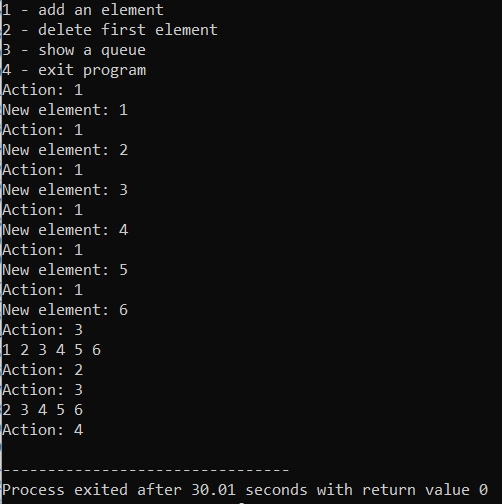
}

}

return 0;

}

3. Результаты работы.



4. Вывод:

Я создал программу, использующую принцип работы очереди, которая позволяет добавлять элементы в конец очереди, удалять начальные и выводить на экран всё содержимое очереди.