Министерство науки и высшего образования Российской Федерации Федеральное государственное автономное учреждение высшего образования «Пермский национальный исследовательский политехнический университет»

Электротехнический факультет Кафедра «Информационные технологии и автоматизированные системы»

**О Т Ч Ё Т по лабораторной работе**

Дисциплина: «Основы алгоритмизации и программирования»

Выполнил студент группы ИВТ21-1Б Лебедев Т.А.

Проверил Яруллин Д.В.

Пермь, 2021

**Постановка задачи:**

1. Сформировать однонаправленный и двунаправленный списки или стек и очередь. Тип информационного поля указан в варианте.

2. Распечатать полученную структуру.

3. Выполнить обработку структуры в соответствии с заданием.

4. Распечатать полученный результат.

5. Удалить соответствующую структуру из памяти.

**Анализ решения задачи:**

функция типа void(ничего не возвращает) которая принимает указатель на вершину стека и переменную которая будет записываться в ячейку

Создаем новый указатель q типа структуры comp. По сути это и есть наш новый элемент

выделяем память для нового элемента

Записываем необходимое число в Data элемента

Если вершины нет, то есть стек пустой

вершиной стека будет новый элемент

Проводим связь от нового элемента, к вершине. Тоесть кладем книжку на вершину стопки.

Обозначаем, что вершиной теперь является новый элемент

функция которая принимает вершину top и число которое нужно удалить

создаем указатель типа comp и приравниваем(ставим) его на вершину стека

создаем указатель на предыдуший элемент, с начала он будет пустым

пока указатель q не пустой, мы будем выполнять код в цикле, если он все же пустой цикл заканчивается

если Data элемента равна числу, которое нам нужно удалить

если такой указатель равен вершине, то есть элемент, который нам нужно удалить - вершина

передвигаем вершину на следующий элемент

очищаем ячейку

если элемент последний или находится между двумя другими элементами

Проводим связь от предыдущего элемента к следующему

очищаем ячейку

обнуляем переменные

если Data элемента НЕ равна числу, которое нам нужно удалить

запоминаем текущую ячейку как предыдущую

перемещаем указатель q на следующий элемент

принимает указатель на вершину стека

устанавливаем q на вершину

пока q не пустой

выводим на экран данные ячейки стека

после того как вывели передвигаем q на следующий элемент(ячейку)

**Описание переменных:**

n- количество элементов стека

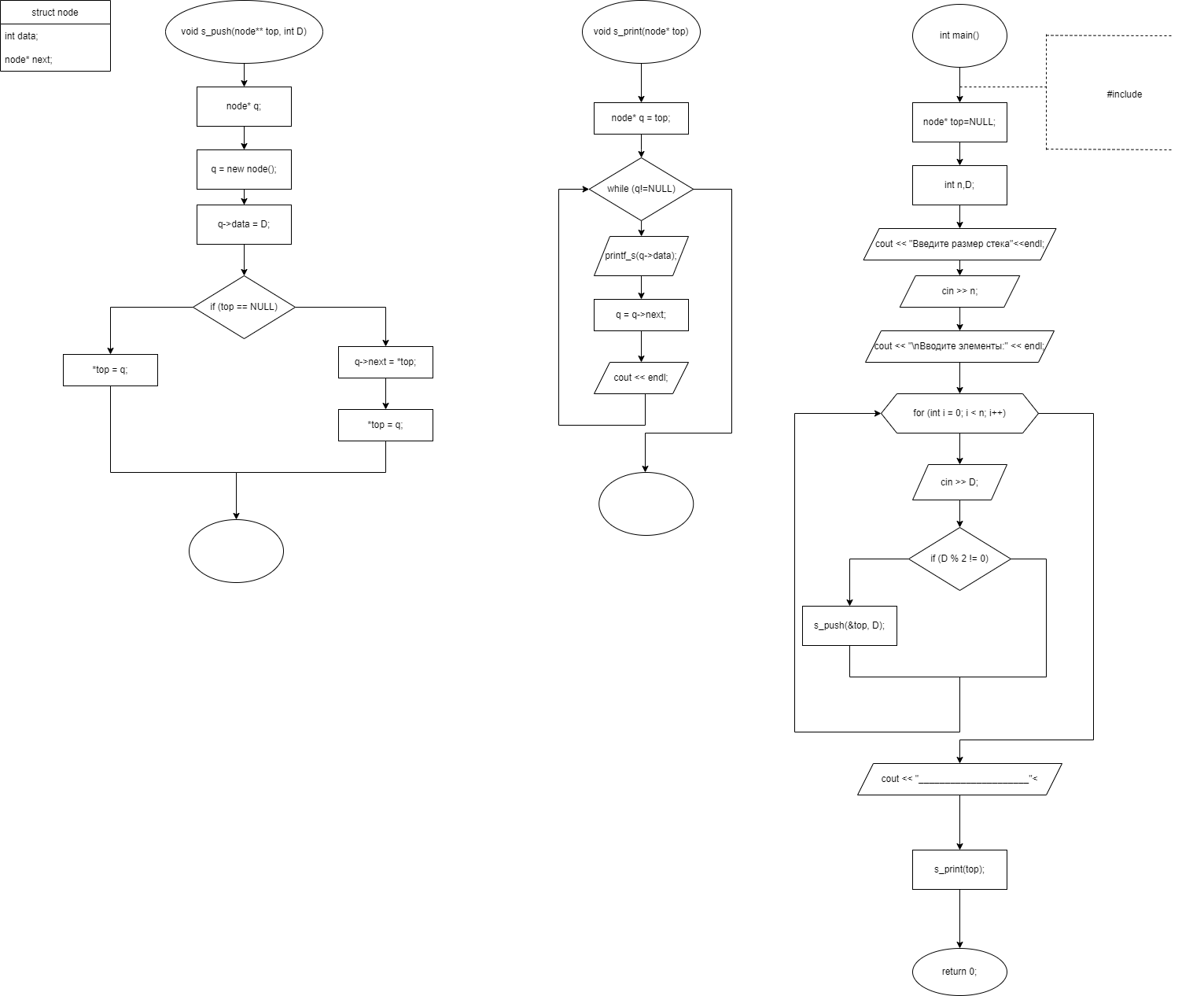
data-информационное поле структуры

\*next-указатель на следующий элемент

\*first-переменная, в которой будет хранится первый элемент списка

\*p,\*r-вспомогательные переменные

**Блок схема:**

****

**Код программы:**

#include<iostream>

using namespace std;

struct node

{

int data;

node\* next;

};

void s\_push(node\*\* top, int D) { //функция типа void(ничего не возвращает) которая принимает указатль на вершину стека и переменную которая будет записываться в ячейку

node\* q; //Создаем новый указатель q типа структуры comp. По сути это и есть наш новый элемент

q = new node(); //выделяем память для нового элемента

q->data = D; //Записываем необходимое число в Data элемента

if (top == NULL) { //Если вершины нет, то есть стек пустой

\*top = q; //вершиной стека будет новый элемент

}

else //если стек не пустой

{

q->next = \*top; //Проводим связь от нового элемента, к вершине. Тоесть кладем книжку на вершину стопки.

\*top = q; //Обозначаем, что вершиной теперь является новый элемент

}

}

void s\_delete\_key(node\*\* top, int N) {//функция которая принимает вершину top и число которое нужно удалить

node\* q = \*top; //создаем указатель типа comp и приравниваем(ставим) его на вершину стека

node\* prev = NULL;//создаем указатель на предыдуший элемент, с начала он будет пустым

while (q != NULL) {//пока указатель q не пустой, мы будем выполнять код в цикле, если он все же пустой цикл заканчивается

if (q->data == N) {//если Data элемента равна числу, которое нам нужно удалить

if (q == \*top) {//если такой указатель равен вершине, то есть элемент, который нам нужно удалить - вершина

\*top = q->next;//передвигаем вершину на следующий элемент

free(q);//очищаем ячейку

q->data = NULL; //Далее во избежание ошибок мы обнуляем переменные в удаленной ячейке, так как в некоторых компиляторах удаленная ячейка имеет переменные не NULL значения, а дословно "Чтение памяти невозможно" или числа "-2738568384" или другие, в зависимости от компилятора.

q->next = NULL;

}

else//если элемент последний или находится между двумя другими элементами

{

prev->next = q->next;//Проводим связь от предыдущего элемента к следующему

free(q);//очищаем ячейку

q->data = NULL;//обнуляем переменные

q->next = NULL;

}

}// если Data элемента НЕ равна числу, которое нам нужно удалить

prev = q; //запоминаем текущую ячейку как предыдущую

q = q->next;//перемещаем указатель q на следующий элемент

}

}

void s\_print(node\* top) //принимает указатель на вершину стека

{

node\* q = top; //устанавливаем q на вершину

while (q!=NULL) //пока q не пустой

{

printf\_s(q->data);//выводим на экран данные ячейки стека

q = q->next;//после того как вывели передвигаем q на следующий элемент(ячейку)

cout << endl;

}

}

int main()

{

node\* top=NULL;

int n,D;

setlocale(LC\_ALL, "ru");

cout << "Введите размер стека"<<endl;

cin >> n;

cout << "\nВводите элементы:" << endl;

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

{

cin >> D;

if (D % 2 != 0)

{

s\_push(&top, D);

}

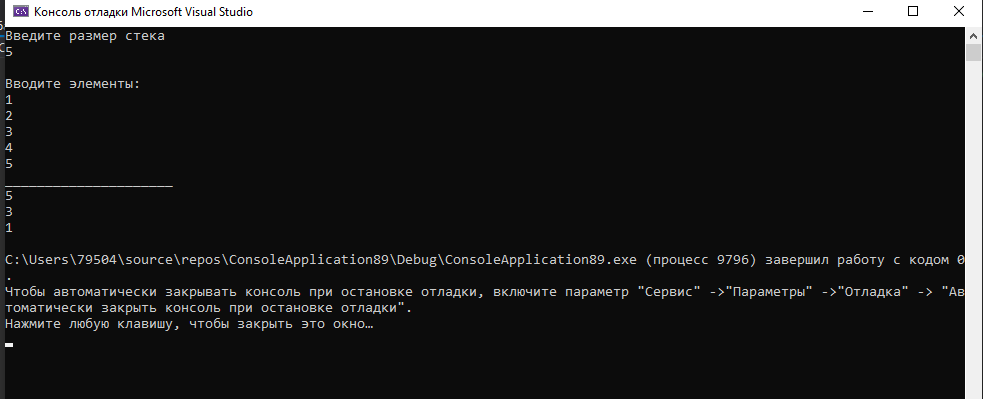
}

cout << "\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_"<<endl;

s\_print(top);

}

**Скриншоты результатов:**

****