



МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение  
высшего образования

«МИРЭА – Российский технологический университет»

**РТУ МИРЭА**

---

Институт Информационных технологий

Кафедра Математического обеспечения и стандартизации информационных  
технологий

## **ОТЧЕТ ПО ПРАКТИЧЕСКОЙ РАБОТЕ № 7**

**по дисциплине**

**«Структуры и алгоритмы обработки данных»**

**Тема: «Применение стека и очереди при преобразовании и  
вычислении арифметических выражений»**

Выполнил студент группы ИКБО-18-22

Ракитин В.А.

Принял преподаватель

Филатов А.С.

Лабораторная работа выполнена

«\_\_»\_\_\_\_\_202\_\_ г.

*(подпись студента)*

«Зачтено»

«\_\_»\_\_\_\_\_202\_\_ г.

*(подпись руководителя)*

Москва 2023

## 1. Цель работы

Получение знаний применения стека и очереди при преобразовании и вычислении арифметических выражений. Получение знаний и практических навыков по работе с различными нотациями арифметических выражений.

## 2. Постановка задачи

Вариант №3. Условие задания:

Упражнение 1	<p>1. Провести преобразование инфиксной записи выражения в префиксную нотацию, расписывая процесс по шагам</p> $S=a+(b-c*k)-d*e-f$ <p>2. Представить постфиксную нотацию выражений</p> $a+(c-b)/(b*d)$ $(a+b)*c-(d+e*f/((g/h+i-j)*k))/r$ <p>3. Представить префиксную нотацию выражений п.2</p> <p>4. Провести вычисление значения выражения, представленного в постфиксной форме, расписывая процесс по шагам</p> $7\ 2 - 3\ 2 + / 3 + 4 *$
Упражнение 2	<p>Выполнить программную реализацию следующих задач, используя структуру стек или очередь</p> <p>1. Реализовать класс стек, реализующий структуру и методы: втолкнуть элемент в стек, вытолкнуть элемент из стека, вернуть значение элемента в вершине стека, сделать стек пустым, определить пуст ли стек. Рассмотреть два варианта реализации: на массиве; на однонаправленном списке. Интерфейс программы должен обеспечивать непрерывную работу со структурой.</p>

	2. Разработать программу сложения двух больших целых чисел (не попадающих в диапазон стандартных типов), вводимых с клавиатуры, как последовательность символов.
--	--

### 3. Решение

#### Задание 1

##### Пункт 1

Преобразуем выражение  $S = a + (b - c * k) - d * e - f$  в префиксную запись. Развернём выражение и получим  $f - e * d - (k * c - b) + a = S$ .

№	Выражение	Стек	Префикс
0		(	
1	f	(	f
2	-	(-	f
3	e	(-	fe
4	*	(-*	fe
5	d	(-*	fed
6	-	(--	fed*
7	(	(--(	fed*
8	k	(--(	fed*k
9	*	(--(*	fed*k
10	c	(--(*	fed*kc
11	-	(--(-	fed*kc*
12	b	(--(-	fed*kc*b
13	)	(--	fed*kc*b-
14	+	(--+	fed*kc*b-
15	a	(--+	fed*kc*b-a
16	=	(--+	fed*kc*b-a=
17	S	(--+	fed*kc*b-a=S
18	)		fed*kc*b-a=S=+--

Разворачиваем выражение обратно. Ответ:  $--+S = a - b * ck * def$

##### Пункт 2

Представляем выражение  $a + (c - b) / (b * d)$  в постфиксном виде

№	Выражение	Стек	Постфикс
---	-----------	------	----------

0		(	
1	a	(	a
2	+	(+	a
3	(	(+(	a
4	c	(+(	ac
5	-	(+(-	ac
6	b	(+(-	acb
7	)	(+	acb-
8	/	(+(/	acb-
9	(	(+/(	acb-
10	b	(+/(	acb-b
11	*	(+/(*	acb-b
12	d	(+/(*	acb-bd
13	)	(+(/	acb-bd*
14	)		acb-bd*/+

Ответ: acb-bd\*/+

Представляем выражение  $(a+b)*c-(d+e*f/((g/h+i-j)*k))/r$  в постфиксном виде

№	Выражение	Стек	Постфикс
0		(	
1	(	((	
2	a	((	a
3	+	((+	a
4	b	((+	ab+
5	)	(	ab+
6	*	(*	ab+
7	c	(*	ab+c
8	-	(-	ab+c*
9	(	(- (	ab+c*
10	d	(- (	ab+c*d
11	+	(- (+	ab+c*d
12	e	(- (+	ab+c*d e
13	*	(- (+*	ab+c*d e
14	f	(- (+*	ab+c*d e f
15	/	(- (+(/	ab+c*d e f*
16	(	(- (+/(	ab+c*d e f*
17	(	(- (+/((	ab+c*d e f*
18	g	(- (+/((	ab+c*d e f* g
19	/	(- (+/((/	ab+c*d e f* g
20	h	(- (+/((/	ab+c*d e f* g h
21	+	(- (+/((+	ab+c*d e f* g h/

22	i	(-(+/(	ab+c*def*gh/i
23	-	(-(+/(	ab+c*def*gh/i+
24	j	(-(+/(	ab+c*def*gh/i+j
25	)	(-(+/(	ab+c*def*gh/i+j-
26	*	(-(+/(	ab+c*def*gh/i+j-
27	k	(-(+/(	ab+c*def*gh/i+j-k
28	)	(-(+/(	ab+c*def*gh/i+j-k*
29	)	(-	ab+c*def*gh/i+j-k*/+
30	/	(-/	ab+c*def*gh/i+j-k*/+
31	r	(-/	ab+c*def*gh/i+j-k*/+r
32	)		ab+c*def*gh/i+j-k*/+r/-

Ответ:  $ab+c*def*gh/i+j-k*/+r/-$

### Пункт 3

Представляем выражение  $a+(c-b)/(b*d)$  в префиксном виде. Развернём выражение:  $(d*b)/(b-c)+a$

№	Выражение	Стек	Префикс
0		(	
1	(	((	
2	d	((	d
3	*	((*	d
4	b	((*	db
5	)	(	db*
6	/	(/	db*
7	(	(/(	db*
8	b	(/(	db*b
9	-	(/(-	db*b
10	c	(/(-	db*bc
11	)	(/	db*bc-
12	+	(+	db*bc-/
13	a	(+	db*bc-/a
14	)		db*bc-/a+

Теперь снова разворачиваем выражение. Ответ:  $+a/-cb*bd$

Представляем выражение  $(a+b)*c-(d+e*f/((g/h+i-j)*k))/r$  в префиксном виде. Для этого разворачиваем выражение:  $r/((k*(j-i+h/g))/f*e+d)-c*(b+a)$

№	Выражение	Стек	Постфикс
0		(	
1	r	(	r
2	/	(/	r

3	(	((	r
4	(	((	r
5	k	((	rk
6	*	((*	rk
7	(	((*(	rk
8	j	((*(	rkj
9	-	((*(-	rkj
10	i	((*(-	rkji
11	+	((*(+	rkji-
12	h	((*(+	rkji-h
13	/	((*(+/	rkji-h
14	g	((*(+/	rkji-hg
15	)	((*(	rkji-hg/+
16	)	((	rkji-hg/+*
17	/	((/	rkji-hg/+*
18	f	((/	rkji-hg/+*f
19	*	((*	rkji-hg/+*f/
20	e	((*	rkji-hg/+*f/e
21	+	(((+	rkji-hg/+*f/e*
22	d	(((+	rkji-hg/+*f/e*d
23	)	((/	rkji-hg/+*f/e*d+
24	-	((-	rkji-hg/+*f/e*d+/-
25	c	((-	rkji-hg/+*f/e*d+/-c
26	*	((-*	rkji-hg/+*f/e*d+/-c
27	(	((-*(	rkji-hg/+*f/e*d+/-c
28	b	((-*(	rkji-hg/+*f/e*d+/-cb
29	+	((-*(+	rkji-hg/+*f/e*d+/-cb
30	a	((-*(+	rkji-hg/+*f/e*d+/-cba
31	)	((-*	rkji-hg/+*f/e*d+/-cba+
32	)		rkji-hg/+*f/e*d+/-cba+*-

Развернём выражение. Ответ:  $-*+abc/+d*e/f*+/gh-ijrk$

Пункт 4

$$7\ 2 - 3\ 2 + / 3 + 4 *$$

1. Указываем на символ 7.
2. Помещаем его в стек.
3. Указываем на символ 2.
4. Помещаем его в стек.
5. указываем на оператор  $-$ .

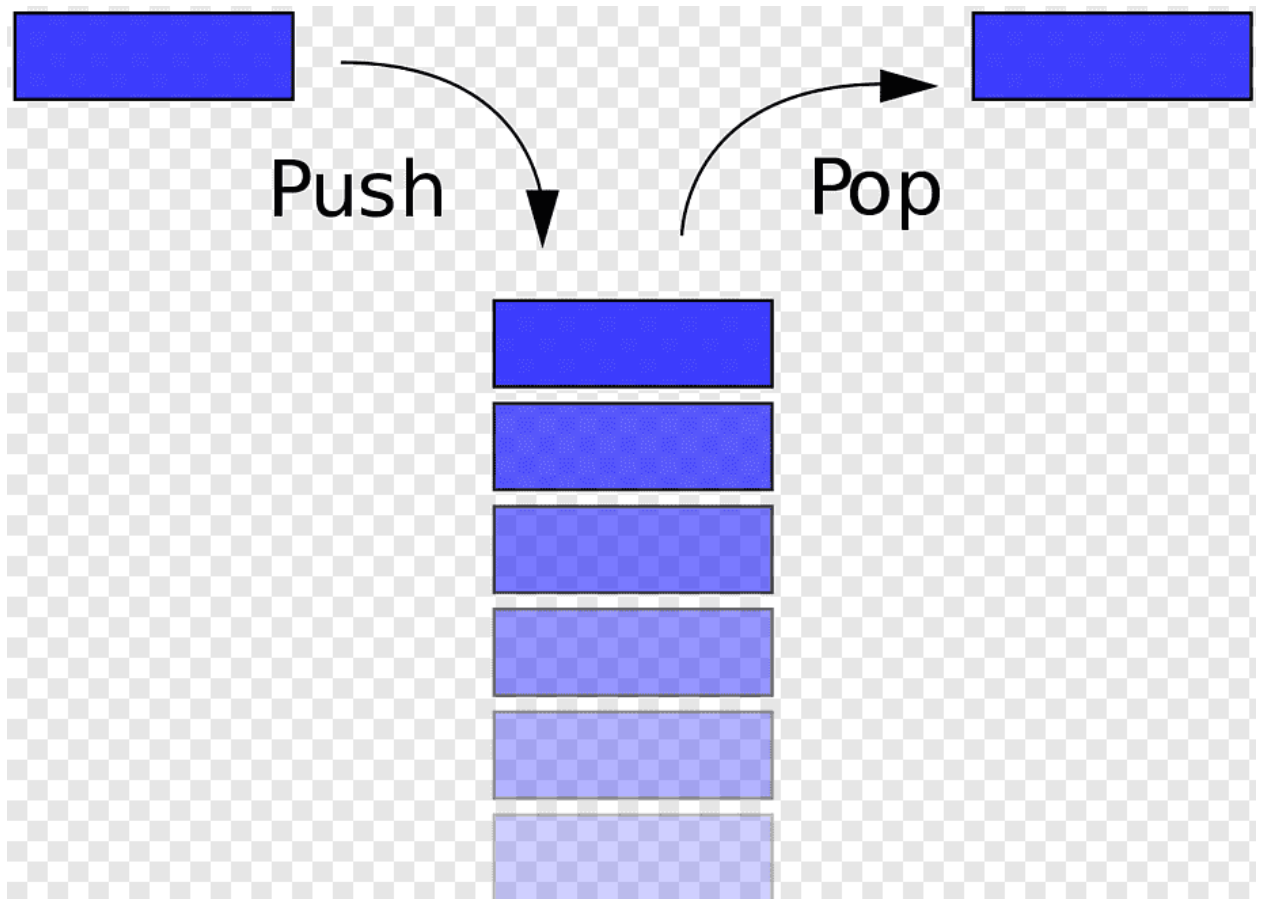
6. Извлекаем из стека два операнда (7 и 2) и выполняем операцию. Результат (5) помещаем в стек.
7. Указываем на символ 3.
8. Помещаем его в стек.
9. Указываем на символ 2.
10. Помещаем его в стек.
11. Указываем на оператор +.
12. Извлекаем из стека два операнда (3 и 2) и выполняем операцию. Результат (5) помещаем в стек.
13. Указываем на оператор /.
14. Извлекаем из стека два операнда (5 и 5) и выполняем операцию. Результат (1) помещаем в стек.
15. Указываем на символ 3.
16. Помещаем его в стек.
17. Указываем на оператор +.
18. Извлекаем из стека два операнда (1 и 3) и выполняем операцию. Результат (4) помещаем в стек.
19. Указываем на символ 4.
20. Помещаем его в стек.
21. Указываем оператор \*.
22. Извлекаем из стека два операнда (4 и 4) и выполняем операцию. Результат (16) помещаем в стек.

Ответ: 16.

## **Задание 2.**

### **Пункт 1.**

Стек – структура данных, представляющая из себя упорядоченный набор элементов, в которой добавление новых элементов и удаление существующих производится с одного конца, называемого вершиной стека. При этом первым из стека удаляется элемент, который был помещен туда последним, то есть в стеке реализуется стратегия «последним вошел — первым вышел».



Для решения первого упражнения была написана функция `push_back`, которая помещает элемент в стек. На вход функция получает целое число – элемент, который будет помещён в стек. Далее программа добавляет этот элемент в наш стек и информирует об этом пользователя. Если в нашем стеке, реализованном на массиве, не будет места, то программа также сообщит пользователю о невозможности добавить элемент в стек.

```
void Stack::push_back(int data) {  
    Node* newNode = new Node;  
    newNode->data = data;  
    newNode->next = nullptr;  
  
    if (head == nullptr)  
        head = newNode;  
    else {  
        Node* current = head;  
  
        while (current->next != nullptr)  
            current = current->next;  
  
        current->next = newNode;  
    }  
    cout << "Элемент " << data << " помещён в стек" << endl;  
}
```



Для решения второго упражнения была написана функция `pull_out`, которая удаляет последний элемент из стека. После удаления элемента, программа информирует пользователя об удалении элемента и скажет, какой элемент удален. Если стек пуст, то программа сообщит, что нечего удалять.

```
void Stack::pull_out() {
    Node* current = head;
    Node* temp = nullptr;
    if (current == nullptr) {
        cout << "Стек пуст. Нечего выталкивать" << endl;
    }
    else if (current->next == nullptr) {
        cout << "Элемент " << current->data << " вытолкнут из стека" <<
endl;
        head = nullptr;
        free(current);
    }
    else {
        while (current->next != nullptr) {
            temp = current;
            current = current->next;
        }
        cout << "Элемент " << current->data << " вытолкнут из стека" <<
endl;
        temp->next = nullptr;
        free(current);
    }
}
```

Для решения третьего упражнения была написана функция `return_meaning`, которая возвращает последний элемент стека. Программа сообщит пользователю, какой последний элемент, а также сообщит, если стек пуст.

```
int Stack::return_meaning() {
    Node* current = head;
    if (current == nullptr) {
        cout << "Стек пуст" << endl;
    }
    else {
        while (current->next != nullptr) {
            current = current->next;
        }
        cout << "Элемент в вершине стека: " << current->data << endl;
        return current->data;
    }
}
```

Для решения четвертого упражнения была написана функция `make_empty_stack`, которая делает стек пустым. Программа удаляет каждый элемент стека, а далее сообщает пользователю об отсутствии элементов в стеке.

```

void Stack::make_empty_stack() {
    Node* current = head;
    Node* next;
    while (current != nullptr) {
        next = current->next;
        delete current;
        current = next;
    }
    head = nullptr;
    cout << "Стек пуст" << endl;
}

```

Для решения пятого упражнения была написана функция `is_empty`, которая определяет, пустой стек или нет. Далее программа проверяет первый элемент стека. Если в стеке имеется первый элемент, то программа сообщит пользователю, что стек не пустой.

```

void Stack::is_empty() {
    Node* current = head;
    if (current == nullptr) {
        cout << "Стек пуст" << endl;
    }
    else {
        cout << "Стек не пустой" << endl;
    }
}

```

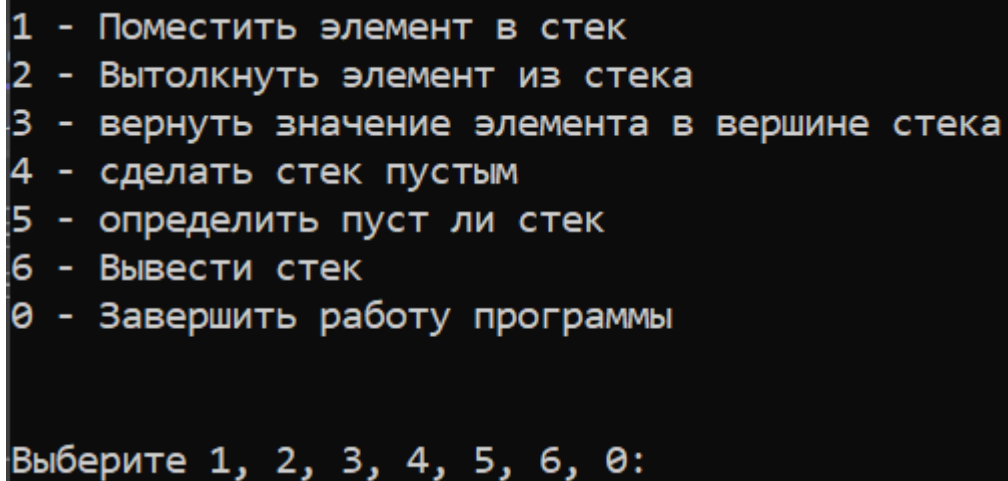
Также для удобства была написана функция `print_Stack`, которая выводит все элементы стека. Если в стеке нет элементов, то программа сообщает пользователю об отсутствии элементов в стеке.

```

void Stack::print_Stack() {
    Node* current = head;
    if (current == nullptr) {
        cout << "Стек пуст" << endl;
    }
    else {
        cout << "Ваш стек: ";
        while (current != nullptr) {
            cout << current->data << " ";
            current = current->next;
        }
        cout << endl;
    }
}

```

При запуске программы пользователь видит пользовательское меню, где пользователю надо выбрать, какую задачу надо решить.



```
1 - Поместить элемент в стек
2 - Вытолкнуть элемент из стека
3 - вернуть значение элемента в вершине стека
4 - сделать стек пустым
5 - определить пуст ли стек
6 - Вывести стек
0 - Завершить работу программы

Выберите 1, 2, 3, 4, 5, 6, 0:
```

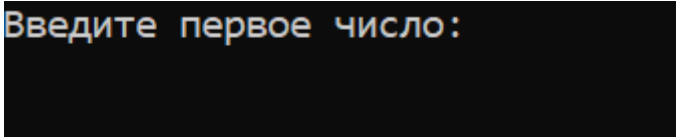
Рисунок 1. Интерфейс программы

Пункт 2.

Для решения упражнения была написана функция `add`, которая складывает два числа. На вход функция получает массивы типа `char`. В двух массивах хранятся наши числа, а третий массив – будущая сумма двух чисел. Далее программа складывает числа и передаёт число в массив.

```
void add(char* num1, char* num2, char* result) {
    int carry = 0;
    int i = strlen(num1) - 1;
    int j = strlen(num2) - 1;
    int k = 0;
    while (i >= 0 || j >= 0) {
        int sum = carry;
        if (i >= 0) {
            sum += num1[i--] - '0';
        }
        if (j >= 0) {
            sum += num2[j--] - '0';
        }
        result[k++] = sum % 10 + '0';
        carry = sum / 10;
    }
    if (carry != 0) {
        result[k++] = carry + '0';
    }
    result[k] = '\0';
    int len = strlen(result);
    for (int i = 0; i < len / 2; i++) {
        swap(result[i], result[len - i - 1]);
    }
}
```

При запуске программы пользователю надо будет ввести число, а затем надо будет ввести второе число.



Введите первое число:

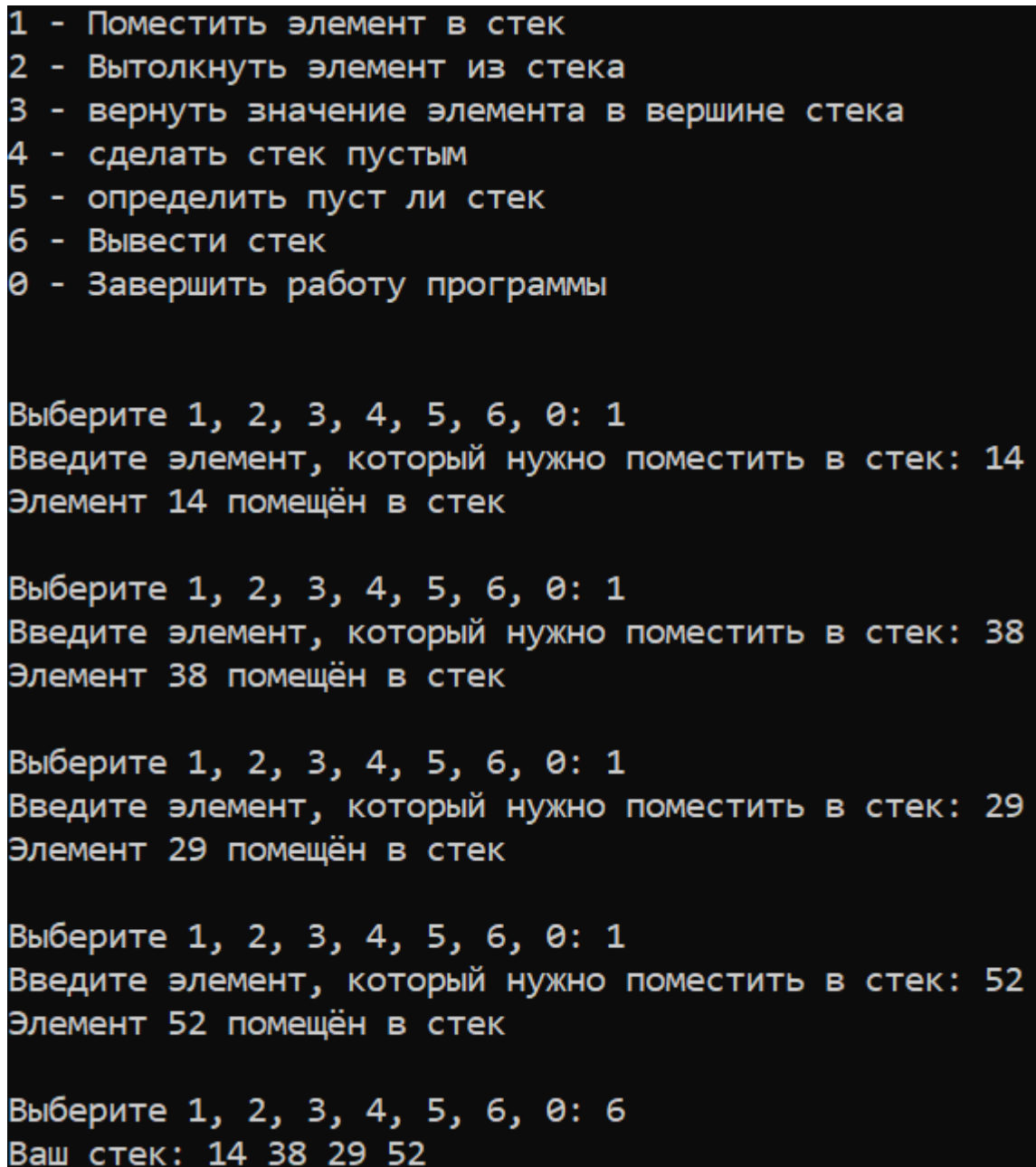
Рисунок 2. Интерфейс программы

#### 4. Тестирование

Пункт 1.

Протестируем программой выполнение первого упражнения. Добавим следующие элементы в стек: 14, 38, 29, 52. Далее выведем элементы стека.

На рисунке 3 программа вывела верный результат.



```
1 - Поместить элемент в стек
2 - Вытолкнуть элемент из стека
3 - вернуть значение элемента в вершине стека
4 - сделать стек пустым
5 - определить пуст ли стек
6 - Вывести стек
0 - Завершить работу программы

Выберите 1, 2, 3, 4, 5, 6, 0: 1
Введите элемент, который нужно поместить в стек: 14
Элемент 14 помещён в стек

Выберите 1, 2, 3, 4, 5, 6, 0: 1
Введите элемент, который нужно поместить в стек: 38
Элемент 38 помещён в стек

Выберите 1, 2, 3, 4, 5, 6, 0: 1
Введите элемент, который нужно поместить в стек: 29
Элемент 29 помещён в стек

Выберите 1, 2, 3, 4, 5, 6, 0: 1
Введите элемент, который нужно поместить в стек: 52
Элемент 52 помещён в стек

Выберите 1, 2, 3, 4, 5, 6, 0: 6
Ваш стек: 14 38 29 52
```

### Рисунок 3. Решение программой первого упражнения

Протестируем программой выполнение второго упражнения. Добавим в стек элементы 11, 55, 28. Далее удалим последний элемент из стека (28) и выведем наш стек (11, 55). На рисунке 4 программа вывела верный результат.

```
1 - Поместить элемент в стек
2 - Вытолкнуть элемент из стека
3 - вернуть значение элемента в вершине стека
4 - сделать стек пустым
5 - определить пуст ли стек
6 - Вывести стек
0 - Завершить работу программы

Выберите 1, 2, 3, 4, 5, 6, 0: 1
Введите элемент, который нужно поместить в стек: 11
Элемент 11 помещён в стек

Выберите 1, 2, 3, 4, 5, 6, 0: 1
Введите элемент, который нужно поместить в стек: 55
Элемент 55 помещён в стек

Выберите 1, 2, 3, 4, 5, 6, 0: 1
Введите элемент, который нужно поместить в стек: 28
Элемент 28 помещён в стек

Выберите 1, 2, 3, 4, 5, 6, 0: 2
Элемент 28 вытолкнут из стека

Выберите 1, 2, 3, 4, 5, 6, 0: 6
Ваш стек: 11 55
```

### Рисунок 4. Решение программой второго упражнения

Протестируем программой выполнение четвёртого упражнения. Введём в стек элементы 68, 77, 32, 51. Выведем наш стек. Далее удалим все элементы из стека и снова выведем наш стек. Программа должна сообщить нам, что в стеке нет элементов. На рисунке 5 программа вывела верный результат.

```
1 - Поместить элемент в стек
2 - Вытолкнуть элемент из стека
3 - вернуть значение элемента в вершине стека
4 - сделать стек пустым
5 - определить пуст ли стек
6 - Вывести стек
0 - Завершить работу программы

Выберите 1, 2, 3, 4, 5, 6, 0: 1
Введите элемент, который нужно поместить в стек: 68
Элемент 68 помещён в стек

Выберите 1, 2, 3, 4, 5, 6, 0: 1
Введите элемент, который нужно поместить в стек: 77
Элемент 77 помещён в стек

Выберите 1, 2, 3, 4, 5, 6, 0: 1
Введите элемент, который нужно поместить в стек: 32
Элемент 32 помещён в стек

Выберите 1, 2, 3, 4, 5, 6, 0: 1
Введите элемент, который нужно поместить в стек: 51
Элемент 51 помещён в стек

Выберите 1, 2, 3, 4, 5, 6, 0: 6
Ваш стек: 68 77 32 51

Выберите 1, 2, 3, 4, 5, 6, 0: 4
Стек пуст

Выберите 1, 2, 3, 4, 5, 6, 0: 6
Стек пуст
```

Рисунок 5. Решение программой четвёртого упражнения

Протестируем программой выполнение пятого упражнения. Введём в стек элементы 11 и 22. Далее выведем наш стек. Затем проверим наличие элементов в стеке. На рисунке 6 программа вывела верный результат.

```
1 - Поместить элемент в стек
2 - Вытолкнуть элемент из стека
3 - вернуть значение элемента в вершине стека
4 - сделать стек пустым
5 - определить пуст ли стек
6 - Вывести стек
0 - Завершить работу программы

Выберите 1, 2, 3, 4, 5, 6, 0: 1
Введите элемент, который нужно поместить в стек: 11
Элемент 11 помещён в стек

Выберите 1, 2, 3, 4, 5, 6, 0: 1
Введите элемент, который нужно поместить в стек: 22
Элемент 22 помещён в стек

Выберите 1, 2, 3, 4, 5, 6, 0: 6
Ваш стек: 11 22

Выберите 1, 2, 3, 4, 5, 6, 0: 5
Стек не пустой
```

Рисунок 6. Решение программой пятого упражнения

## Пункт 2.

Протестируем программой выполнения упражнения. Введём два числа: 53228 и 48712. Программа должна вывести новое число: 101940. На рисунке 7 программа вывела верный результат.

```
Введите первое число: 53228
Введите второе число: 48712
Ответ: 101940
```

Рисунок 7. Решение программой упражнения

## 5. Вывод

В результате работы я:

- 1) Получил знания применения стека и очереди при преобразовании и вычислении арифметических выражений.
- 2) Получил знания и практические навыки по работе с различными нотациями арифметических выражений.

## 6. Исходный код программы

```
#include <iostream>
using namespace std;

const int MAX_SIZE = 100;

class Stack {
private:
    int arr[MAX_SIZE];
    int k = -1;
public:
    void push_back(int x);
    void pull_out();
    void print_Stack();
    void is_empty();
    void make_empty_stack();
    int return_meaning();
};

void Stack::push_back(int x) {
    if (k != MAX_SIZE) {
        k += 1;
        arr[k] = x;
        cout << "Элемент " << x << " Помещён в стек" << endl;
    }
    else {
        cout << "Стек полон" << endl;
    }
}

void Stack::pull_out() {
    if (k == -1) {
        cout << "Стек пуст. Нечего выталкивать" << endl;
    }
    else {
        cout << "Элемент " << arr[k] << " вытолкнут из стека" << endl;
        k -= 1;
    }
}

void Stack::print_Stack() {
    if (k == -1) {
        cout << "Стек пуст" << endl;
    }
    else {
        int c = 0;
        cout << "Ваш стек: ";
```



```

        while (c != k+1) {
            cout << arr[c] << " ";
            c += 1;
        }
    }
}

void Stack::is_empty() {
    if (k == -1) {
        cout << "Стек пустой" << endl;
    }
    else {
        cout << "Стек не пустой" << endl;
    }
}

void Stack::make_empty_stack() {
    k = -1;
    cout << "Стек пуст" << endl;
}

int Stack::return_meaning() {
    if (k == -1) {
        cout << "Стек пуст" << endl;
    }
    else {
        cout << "Элемент в вершине стека: " << arr[k] << endl;
        return arr[k];
    }
}

int main() {
    setlocale(LC_ALL, "Rus");
    Stack s;
    int pointer = 1;
    cout << endl << "1 - Поместить элемент в стек" << endl
        << "2 - Вытолкнуть элемент из стека" << endl
        << "3 - вернуть значение элемента в вершине стека" << endl
        << "4 - сделать стек пустым" << endl
        << "5 - определить пуст ли стек" << endl
        << "6 - Вывести стек" << endl
        << "0 - Завершить работу программы" << endl << endl;
    while (pointer == 1) {
        int choise;
        cout << endl << "Выберите 1, 2, 3, 4, 5, 6, 0: "; cin >> choise;
        switch (choise)
        {
            case (1):
                int x;
                cout << "Введите элемент, который нужно поместить в стек: "; cin >> x;
                s.push_back(x);
                break;
            case (2):
                s.pull_out();
                break;
            case (3):
                s.return_meaning();
                break;
            case (4):
                s.make_empty_stack();
                break;
            case (5):
                s.is_empty();

```

```

        break;
    case (6):
        s.print_Stack();
        break;
    case (0):
        pointer = 0;
        break;
    default:
        break;
    }
}
return 0;
}

```

Таблица 1. Код программы задачи №1 на основе статического массива

```

#include <iostream>
using namespace std;

const int MAX_SIZE = 100;

class Stack {
private:
    struct Node {
        int data;
        Node* next;
    };
    Node* head;
    int arr[MAX_SIZE];
    int k = -1;
public:
    void push_back(int data);
    void pull_out();
    void print_Stack();
    void is_empty();
    void make_empty_stack();
    int return_meaning();
};

void Stack::push_back(int data) {
    Node* newNode = new Node;
    newNode->data = data;
    newNode->next = nullptr;

    if (head == nullptr)
        head = newNode;
    else {
        Node* current = head;

        while (current->next != nullptr) {
            current = current->next;
        }

        current->next = newNode;
    }
    cout << "Элемент " << data << " помещён в стек" << endl;
}

void Stack::pull_out() {
    Node* current = head;
    Node* temp = nullptr;
    if (current == nullptr) {
        cout << "Стек пуст. Нечего выталкивать" << endl;
    }
}

```

```

        else if (current->next == nullptr) {
            cout << "Элемент " << current->data << " вытолкнут из стека" <<
endl;
            head = nullptr;
            free(current);
        }
        else {
            while (current->next != nullptr) {
                temp = current;
                current = current->next;
            }
            cout << "Элемент " << current->data << " вытолкнут из стека" <<
endl;
            temp->next = nullptr;
            free(current);
        }
    }

void Stack::print_Stack() {
    Node* current = head;
    if (current == nullptr) {
        cout << "Стек пуст" << endl;
    }
    else {
        cout << "Ваш стек: ";
        while (current != nullptr) {
            cout << current->data << " ";
            current = current->next;
        }
        cout << endl;
    }
}

void Stack::is_empty() {
    Node* current = head;
    if (current == nullptr) {
        cout << "Стек пуст" << endl;
    }
    else {
        cout << "Стек не пустой" << endl;
    }
}

void Stack::make_empty_stack() {
    Node* current = head;
    Node* next;
    while (current != nullptr) {
        next = current->next;
        delete current;
        current = next;
    }
    head = nullptr;
    cout << "Стек пуст" << endl;
}

int Stack::return_meaning() {
    Node* current = head;
    if (current == nullptr) {
        cout << "Стек пуст" << endl;
    }
    else {
        while (current->next != nullptr) {
            current = current->next;
        }
    }
}

```

```

        cout << "Элемент в вершине стека: " << current->data << endl;
        return current->data;
    }
}

int main() {
    setlocale(LC_ALL, "Rus");
    Stack s;
    int pointer = 1;
    cout << endl << "1 - Поместить элемент в стек" << endl
        << "2 - Вытолкнуть элемент из стека" << endl
        << "3 - вернуть значение элемента в вершине стека" << endl
        << "4 - сделать стек пустым" << endl
        << "5 - определить пуст ли стек" << endl
        << "6 - Вывести стек" << endl
        << "0 - Завершить работу программы" << endl << endl;
    while (pointer == 1) {
        int choose;
        cout << endl << "Выберите 1, 2, 3, 4, 5, 6, 0: "; cin >> choose;
        switch (choose)
        {
            case(1):
                int x;
                cout << "Введите элемент, который нужно поместить в стек: ";
                cin >> x;
                s.push_back(x);
                break;
            case(2):
                s.pull_out();
                break;
            case(3):
                s.return_meaning();
                break;
            case(4):
                s.make_empty_stack();
                break;
            case(5):
                s.is_empty();
                break;
            case(6):
                s.print_Stack();
                break;
            case(0):
                pointer = 0;
                break;
            default:
                break;
        }
    }
    return 0;
}

```

Таблица 2. Код программы задачи №1 на основе однонаправленного списка

```

#include <iostream>
using namespace std;

const int MAX_SIZE = 100;

void add(char* num1, char* num2, char* result) {
    int carry = 0;
    int i = strlen(num1) - 1;
    int j = strlen(num2) - 1;
    int k = 0;

```

```

while (i >= 0 || j >= 0) {
    int sum = carry;
    if (i >= 0) {
        sum += num1[i--] - '0';
    }
    if (j >= 0) {
        sum += num2[j--] - '0';
    }
    result[k++] = sum % 10 + '0';
    carry = sum / 10;
}
if (carry != 0) {
    result[k++] = carry + '0';
}
result[k] = '\0';
int len = strlen(result);
for (int i = 0; i < len / 2; i++) {
    swap(result[i], result[len - i - 1]);
}
}

int main() {
    setlocale(LC_ALL, "Rus");
    char num1[MAX_SIZE];
    char num2[MAX_SIZE];
    char result[MAX_SIZE];
    cout << "Введите первое число: "; cin >> num1;
    cout << "Введите второе число: "; cin >> num2;
    add(num1, num2, result);
    cout << "Ответ: " << result << endl;
    return 0;
}

```

Таблица 3. Код программы задачи №2