МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МО ЭВМ

ОТЧЕТ

по лабораторной работе №4
по дисциплине «Программирование»
Тема: Динамические структуры данных

Студент гр. 1304 _____ Павлов Д.Р.

Преподаватель _____ Чайка К.В.

Санкт-Петербург
2022

Цель работы.

Требуется написать программу, которая последовательно выполняет подаваемые ей на вход арифметические операции над числами с помощью стека на базе массива..

Задание.

1) Реализовать **класс** CustomStack, который будет содержать перечисленные ниже методы. Стек должен иметь возможность хранить и работать с типом данных **int.**

Объявление класса стека:

```
class CustomStack {

    public:

    // методы push, pop, size, empty, top + конструкторы, деструктор

    private:

    // поля класса, к которым не должно быть доступа извне

    protected:  // в этом блоке должен быть указатель на массив данных

       int* mData;
    };
```

Перечень методов класса стека, которые должны быть реализованы:

- **void push(int val)** - добавляет новый элемент в стек
- **void pop()** - удаляет из стека последний элемент
- **int top()** - доступ к верхнему элементу
- **size_t size()** - возвращает количество элементов в стеке
- **bool empty()** - проверяет отсутствие элементов в стеке
- **extend(int n)** - расширяет исходный массив на n ячеек

2) Обеспечить в программе считывание из потока **stdin** последовательности (не более 100 элементов) из чисел и арифметических операций (+, -, *, / (деление нацело)) разделенных пробелом, которые программа должна интерпретировать и выполнить по следующим правилам:

- Если очередной элемент входной последовательности - число, то положить его в стек,
- Если очередной элемент - знак операции, то применить эту операцию над двумя верхними элементами стека, а результат положить обратно в стек (следует считать, что левый операнд выражения лежит в стеке глубже),
- Если входная последовательность закончилась, то вывести результат (число в стеке).

Если в процессе вычисления возникает ошибка:

- например вызов метода **pop** или **top** при пустом стеке (для операции в стеке не хватает аргументов),
- по завершении работы программы в стеке более одного элемента,

программа должна вывести "**error**" и завершиться.

Тестирование.

Результаты тестирования представлены в табл. 1.

Таблица 1 – Результаты тестирования

№ п/п	Входные данные	Выходные данные	Комментарии
1.	1 -10 - 2 *	22	Ответ правильный

Выводы.

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: main.cpp

```
class CustomStack{
    public:
        CustomStack(){
            id = 0;
            mData = (int*)malloc(sizeof(int));
        }

        ~CustomStack(){
            free(mData);
        }

        void push(int val){
            mData[id] = val;
            id++;
        }

        void pop(){
            if (this->empty() == false){
                id--;
            }
            else{
                cout<<"error";
                exit(0);
            }
        }

        int top(){
            if (this->empty() == false){
                return mData[id-1];
            }else{
                cout<<"error";
                exit(0);
            }
        }

        int size(){
            return id;
        }
```

```
        bool empty(){
            return id == 0;
        }

        void extend(int n){
            mData = (int*)realloc(mData, n * sizeof(int));
        }

    size_t id;
    protected:
        int *mData;
};


int main(){
    string input;
    getline(cin, input, '\n');
    int numb = 0, a, b, sign = 1, n = 0;
    int check = 0;
    int i;
    CustomStack Stack;
    for(i = 0; i<input.length(); i++){
        if (input[i] == '+'){
            a = Stack.top();
            Stack.pop();
            b = Stack.top();
            Stack.pop();
            Stack.push(a + b);
            continue;
        }

        if (input[i] == '*'){
            a = Stack.top();
            Stack.pop();
            b = Stack.top();
            Stack.pop();
            Stack.push(a * b);
            continue;
        }

        if (input[i] == '/'){
            a = Stack.top();
            Stack.pop();
            b = Stack.top();
```

```
            Stack.pop();
            Stack.push(a / b);
            continue;
        }
        if (input[i] == '-'){
            if (i < input.length() - 1 && isdigit(input[i + 1])){
                sign = -1;
            }else{
                a = Stack.top();
                Stack.pop();
                b = Stack.top();
                Stack.pop();
                Stack.push(b - a);
            }
            continue;
        }

        if (input[i] == ' '){
            if (check == 1){
                n++;
                Stack.extend(n);
                Stack.push(numb*sign);
                numb = 0;
                sign = 1;
                check = 0;
            }
            continue;
        }
        check = 1;
        numb = 10 * numb + (int)(input[i] - '0');
    }
    if (check == 1){
        n++;
        Stack.extend(n);
        Stack.push(numb*sign);
    }
    if (Stack.size() > 1){
        cout<<"error";
    }else{
        cout<<Stack.top();
    }
    return 0;
}
```