**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра МОЭВМ**

отчет

**по лабораторной работе № 4**

**по дисциплине «Программирование»**

Тема: «Динамические структуры данных»

|  |  |  |
| --- | --- | --- |
| Студент гр. 3343 |  | Иванов П.Д. |
| Преподаватель |  | Государкин Я.С. |

Санкт-Петербург

2024

# Цель работы

Изучение особенностей применения классов в языке программирования C++ и освоение методов работы с ними. Также разработать динамическую структуру данных стек на основе массива с применением принципов объектно-ориентированного программирования.

# Задание

ТТребуется написать программу, моделирующую работу стека на базе массива. Для этого необходимо:

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()* - возвращает верхний элемент

*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 команд (каждая команда с новой строки), в зависимости от которых программа выполняет ту или иную операцию и выводит результат ее выполнения с новой строки.

Перечень команд, которые подаются на вход программе в stdin:

cmd\_push n - добавляет целое число n в стек. Программа должна вывести "ok"

cmd\_pop - удаляет из стека последний элемент и выводит его значение на экран

cmd\_top - программа должна вывести верхний элемент стека на экран не удаляя его из стека

cmd\_size - программа должна вывести количество элементов в стеке

cmd\_exit - программа должна вывести "bye" и завершить работу

Если в процессе вычисления возникает ошибка (например вызов метода pop или top при пустом стеке), программа должна вывести "error" и завершиться.

## Выполнение работы

*CustomStack*: Этот класс представляет стек. Он содержит приватное поле *mSize*, определяющее текущий размер стека, и защищенное поле *mData*, представляющее динамический массив данных, который служит основой для стека.

Конструктор инициализирует стек, устанавливая размер *mSize* в 0 и выделяя память для массива *mData*. Деструктор освобождает память, выделенную под массив mData.

Методы были реализованы согласно заданию:

*push(int obj):* добавляет элемент на вершину стека.

*pop():* удаляет элемент с вершины стека.

*top():* возвращает значение элемента на вершине стека.

*size():* возвращает текущий размер стека.

*empty():* возвращает true, если стек пуст, и false в противном случае.

Метод *extend*: этот метод используется для изменения размера массива *mData*, который используется для хранения элементов стека.

В функции *main* создается объект *CustomStack*. Затем программа ожидает ввода команды (например, cmd\_push). В зависимости от введенной команды выполняются соответствующие действия: добавление элемента, удаление элемента, получение верхнего элемента, вывод размера стека или завершение программы. При возникновении исключения выводится сообщение "error". Программа завершает свою работу при вводе команды cmd\_exit.

## Тестирование

Результаты тестирования содержатся в таблице 1.

Таблица 1.

|  |  |  |  |
| --- | --- | --- | --- |
| № | Входные данные | Выходные данные | Комментарии |
| 1. | cmd\_push 1  cmd\_top  cmd\_push 2  cmd\_top  cmd\_pop  cmd\_size  cmd\_pop  cmd\_size  cmd\_exit | ok  1  ok  2  2  1  1  0  bye | Вывод соответствует ожиданиям. |
|  |  |  |

# Выводы

В процессе выполнения лабораторной работы был изучен синтаксис языка C++ для работы с классами, а также была разработана программа, реализующая стек на основе массива.

ПРИЛОЖЕНИЕ А  
ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: main.c

#include <iostream>

using namespace std;

class CustomStack{

public:

CustomStack(){

this->mSize = 0;

this->mData = new int[this->mSize];

}

~CustomStack(){

delete this->mData;

}

void push(int obj){

this->extend(1);

this->mData[this->mSize-1] = obj;

}

void pop(){

if(this->mSize == 0){

throw 1;

}

this->extend(-1);

}

int top(){

if(this->mSize == 0){

throw 2;

}

return this->mData[this->mSize-1];

}

size\_t size(){

return (size\_t)this->mSize;

}

bool empty(){

return this->mSize == 0;

}

void extend(int n){

if(this->mSize+n < 0){

throw 3;

}

int\* new\_mData = new int[this->mSize+n];

if(n < 0) {

for (int i = 0; i < this->mSize-n; ++i) {

new\_mData[i] = this->mData[i];

}

} else {

for (int i = 0; i < this->mSize; ++i) {

new\_mData[i] = this->mData[i];

}

}

delete this->mData;

this->mSize += n;

this->mData = new\_mData;

}

private:

int mSize;

protected:

int\* mData;

};

int main(){

string command;

CustomStack stack = CustomStack();

do{

cin >> command;

try{

if(command == "cmd\_push"){

int value;

cin >> value;

stack.push(value);

cout << "ok" << endl;

} else if(command == "cmd\_pop"){

cout << stack.top() << endl;

stack.pop();

} else if(command == "cmd\_top"){

cout << stack.top() << endl;

} else if(command == "cmd\_size"){

cout << stack.size() << endl;

} else if(command == "cmd\_exit"){

cout << "bye" << endl;

exit(0);

} else {

throw 4;

}

}catch (int exp) {

cout << "error" << endl;

break;

}

}while(!command.empty());

}