**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. 1384 |  | Усачева Д.В. |
| Преподаватель |  | Жангиров Т. Р. |
|  |  |  |

Санкт-Петербург 2022

**Цель работы.**

Научиться работать со структурами данных в языке C++.

**Задание.**

(Вариант 3)

**Моделирование стека.**   
Требуется 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.

**Примечания:**

1. Указатель на массив должен быть protected.
2. Подключать какие-то заголовочные файлы не требуется, всё необходимое подключено.
3. Предполагается, что пространство имен std уже доступно.
4. Использование ключевого слова using также не требуется.
5. Методы не должны выводить ничего в консоль.

**Выполнение работы.**

Класс CustomStack содержит три поля - текущий размер массива, ссылка на начала массива и текущий размер буфера массива. Этот класс также содержит методы:

* push
* pop
* top
* size
* empty
* extend

Пользователь не имеет прямого доступа к полям стека. В функции main программа создает стек и строку для команд str. Далее в цикле программа считывает команду и выполняет одно из 5 заданных действий. В случае некорректной команды (команда введена с ошибкой, такой команды нет или попытка вывести элемент пустого стека) программа выведет error и завершится.

Разработанный программный код см. в приложении 2.

Результаты тестирования см. в приложении 1.

**Выводы.**

Были освоены основы работы со структурами данных в языке С++

**ПРИЛОЖЕНИЕ 1**

**ТЕСТИРОВАНИЕ**

Таблица 1 - Пример тестовых случаев

|  |  |  |
| --- | --- | --- |
| № п/п | Входные данные | Выходные данные |
| 1. | cmd\_push 1  cmd\_push 2  cmd\_push 3  cmd\_pop  cmd\_pop | ok  ok  ok  3  2 |
|  | 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 |

**ПРИЛОЖЕНИЕ 2**

**ИСХОДНЫЙ КОД ПРОГРАММЫ**

Название файла: main.cpp

class CustomStack {

public:

CustomStack() {

sz = 0;

buf = 100;

mData = new int[buf];

}

~CustomStack() {

delete[] mData;

}

void push(int val) {

mData[sz] = val;

sz++;

if (sz >= buf)extend(100);

}

void pop() {

sz--;

}

int top() {

return mData[sz - 1];

}

size\_t size() {

return sz;

}

bool empty() {

if (sz == 0) {

return true;

}

else {

return false;

}

}

void extend(int n) {

buf += n;

mData = (int\*)realloc(mData, sizeof(int) \* buf);

}

private:

size\_t sz;

int buf;

protected:

int\* mData;

};

int main() {

char\* str = new char[256];

CustomStack st;

for (;;) {

std::cin >> str;

if (!strcmp(str, "cmd\_push")) {

int n;

cin >> n;

st.push(n);

cout << "ok" << endl;

}

else if (!strcmp(str, "cmd\_pop")) {

if (st.empty()) {

cout << "error" << endl;

break;

}

else {

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

st.pop();

}

}

else if (!strcmp(str, "cmd\_top")) {

if (st.empty()) {

cout << "error" << endl;

break;

}

else cout << st.top() << endl;

}

else if (!strcmp(str, "cmd\_size")) {

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

}

else if (!strcmp(str, "cmd\_exit")) {

cout << "bye";

break;

}

else {

cout << "error" << endl;

break;

}

}

return 0;

}