|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | |  |  | | МИНОБРНАУКИ РОССИИ | | | | Федеральное государственное бюджетное образовательное учреждение  высшего образования  **«МИРЭА – Российский технологический университет»**  **РТУ МИРЭА** | | | |  | | | | | |
|  | | |
|  | Институт информационных технологий (ИТ) |
|  | Кафедра математического обеспечения и стандартизации информационных технологий (МОСИТ) |

|  |  |
| --- | --- |
| **ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №2** | |
| **по дисциплине** | |
| **«Структуры и алгоритмы обработки данных»**  **По теме: «Структура данных - стек»**  **Вариант 4** | |
|  | |
| Выполнил студент группы ИВБО-06-17 | *Галеев Д.Р.* |
| Принял преподаватель | *Скворцова Л.А.* |

Москва 2019

Оглавление

[1. Задание 1. Разработать класс Стек в соответствии задачей варианта, с реализацией на одномерном динамическом массиве. 3](#_Toc27519108)

[1.1. Вариант задания 3](#_Toc27519109)

[1.2. Абстрактный тип данных (далее АТД) для варианта задания, включая список общих функций из задания. 3](#_Toc27519110)

[1.3 Список модулей реализации АТД (или описать где расположена 4](#_Toc27519111)

[реализация АТД) 4](#_Toc27519112)

[1.4. Таблица тестов 4](#_Toc27519113)

[1.5. Текст исходного кода (листинг) программы 4](#_Toc27519114)

[2. Задание 2. Разработать класс Стек в соответствии задачей варианта, с реализацией на однонаправленном списке 4](#_Toc27519115)

[2.1. Вариант задания 4](#_Toc27519116)

[2.2. Реализация АТД 4](#_Toc27519117)

[2.3 Список модулей реализации АТД (или описать где расположена 5](#_Toc27519118)

[реализация АТД) 5](#_Toc27519119)

[2.4. Таблица тестов 5](#_Toc27519120)

[2.5. Текст исходного кода (листинг) программы 5](#_Toc27519121)

[3. Дополнительные задания 5](#_Toc27519122)

[3.1. Вариант задания 5](#_Toc27519123)

[3.2. Вариант задания 5](#_Toc27519124)

[3.3. Таблица тестов 6](#_Toc27519125)

[3.4. Текст исходного кода (листинг) программы 6](#_Toc27519126)

[4. Общие контрольные прогоны программы 6](#_Toc27519127)

[Выводы 8](#_Toc27519128)

[Список использованных информационных источников 8](#_Toc27519129)

[Приложение 1 9](#_Toc27519130)

1. Задание 1. Разработать класс Стек в соответствии задачей варианта, с реализацией на одномерном динамическом массиве.

1.1. Вариант задания

Вычислить значение арифметического безскобочного выражения, введенного в виде строки, предварительно преобразовав его в постфиксную форму. Операнды операций – это целые числа. Перед преобразованием выражения следует проверить его на корректность записи операций и операндов.

Требования:

* 1. Определение класса реализовать в отдельном заголовочном файле, реализация методов в файле cpp.
  2. В классе определить: конструктор без параметров и деструктор, а также другие методы управления структурой, для решения задачи.
  3. Разработать программу, тестирования класса.

## 1.2. Абстрактный тип данных (далее АТД) для варианта задания, включая список общих функций из задания.

Абстрактный тип данных Stack

T\* arr;

int elements;

Узнать, пустой ли массив:

empty();

Добавить в конец элемент:

push(T el);

Удалить элемент:

pop();

Получить значение вершины:

top();

1.3. Реализация АТД

АТД реализуется на одномерном динамическом массиве.

template <typename T>

class AStack

{

public:

AStack();

~AStack();

bool empty();

void push(T el);

T top();

void pop();

T poll();

private:

T\* arr;

int elements;

};

1.3 Список модулей реализации АТД (или описать где расположена

реализация АТД)

Реализация АТД состоит из файла заголовка AStack.h и файла AStack.cpp

## 1.4. Таблица тестов

|  |  |  |  |
| --- | --- | --- | --- |
| Номер теста | Исходные данные | Эталон результата | |
| Выражение в постфиксной форме | Результат выражения |
| 1 | 2+2\*2 | 2 2 2\*+ | 6 |
| 2 | 4+5\*3-2/1 | 4 5 3 \*+2 1/- | 17 |
| 3 | 1+2\*2+3\*6/9 | 1 1 2 2 \*+3 6 \*9/+ | 7 |

## 1.5. Текст исходного кода (листинг) программы

Листинг программы представлен в приложении 1.

2. Задание 2. Разработать класс Стек в соответствии задачей варианта, с реализацией на однонаправленном списке

2.1. Вариант задания

Требования

* 1. Структуру узла определить отдельным классом с набором необходимых методов для инициализации переменных.
  2. Определить класс Стек со структурой узла, реализованного согласно п.1. В классе определить: конструктор без параметров и деструктор для удаления списка, а также другие методы управления структурой, для решения задачи.
  3. Для тестирования созданного класса используйте программу, созданную в задании 1.

## 2.2. Реализация АТД

АТД реализуется на однонаправленном списке.

template <typename T>

class Node

{

public:

Node();

~Node();

void set\_data(T d);

void set\_next(Node<T>\* n);

T get\_data();

Node<T>\* get\_next();

private:

Node\* next;

T data;

};

template <typename T>

class NStack

{

public:

NStack();

~NStack();

bool empty();

void push(T el);

T top();

void pop();

T poll();

private:

Node<T>\* node;

int elements;

};

## 2.3 Список модулей реализации АТД (или описать где расположена

реализация АТД)

Реализация АТД расположена в файлах NStack.h и NStack.cpp

## 2.4. Таблица тестов

## 2.5. Текст исходного кода (листинг) программы

Листинг программы представлен в приложении 1.

3. Дополнительные задания

3.1. Вариант задания

Разработайте программу тестирования созданных классов, управляемую консольным меню.

3.2. Вариант задания

Реализуйте задачу варианта, используя контейнер stack из STL. Добавьте тестирование в созданную программу.

3.3. Таблица тестов

|  |  |  |  |
| --- | --- | --- | --- |
| Номер теста | Исходные данные | Эталон результата | |
| Выражение в постфиксной форме | Результат выражения |
| 1 | 2+2\*2 | 2 2 2\*+ | 6 |
| 2 | 4+5\*3-2/1 | 4 5 3 \*+2 1/- | 17 |
| 3 | 1+2\*2+3\*6/9 | 1 1 2 2 \*+3 6 \*9/+ | 7 |

## 3.4. Текст исходного кода (листинг) программы

Листинг программы представлен в приложении 1.

4. Общие контрольные прогоны программы

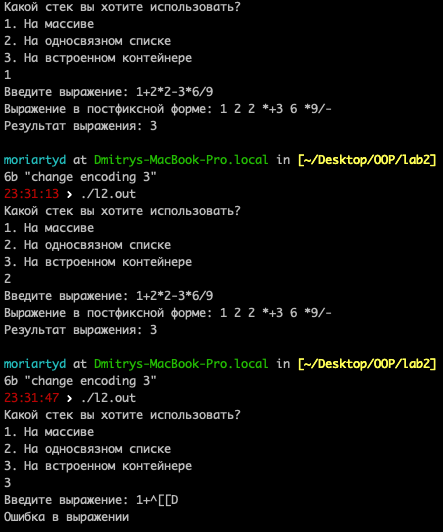


Рис.1 Контрольные прогоны программы

Выводы

В результате выполнения лабораторной работы были получены навыки по созданию структуры данных стек и ее применению.

Стек был разработан на одномерном динамическом массиве, однонаправленном списке. Была разработана программа тестирования для проверки правильности вывода результата. Был изучен контейнер stack из STL.

Список использованных информационных источников

1. Скворцова Л.А. Структуры и алгоритмы ОД, лекции РТУ МИРЭА, 2019/2020 уч.г.

Приложение 1

AStack.cpp

#include "AStack.h"

#include <iostream>

template<typename T>

AStack<T>::AStack()

{

elements = 0;

arr = 0;

}

template<typename T>

AStack<T>::~AStack()

{

}

template<typename T>

bool AStack<T>::empty()

{

return (this->elements ? false : true);

}

template<typename T>

void AStack<T>::push(T el)

{

this->elements++;

if (elements == 1)

this->arr = (T\*)malloc(sizeof(T));

else

this->arr = (T\*)realloc(this->arr, sizeof(T) \* this->elements);

this->arr[this->elements - 1] = el;

}

template<typename T>

T AStack<T>::top()

{

return T(this->arr[this->elements - 1]);

}

template<typename T>

void AStack<T>::pop()

{

this->elements--;

if (!elements)

{

free(this->arr);

this->arr = NULL;

}

else

this->arr = (T\*)realloc(this->arr, sizeof(T) \* (this->elements));

}

template<typename T>

T AStack<T>::poll()

{

T tmp;

tmp = this->top();

this->pop();

return T(tmp);

}

AStack.h

template <typename T>

class AStack

{

public:

AStack();

~AStack();

bool empty();

void push(T el);

T top();

void pop();

T poll();

private:

T\* arr;

int elements;

};

NStack.cpp

#include "NStack.h"

#include <stdlib.h>

template<typename T>

Node<T>::Node()

{

}

template<typename T>

Node<T>::~Node()

{

}

template<typename T>

void Node<T>::set\_data(T d)

{

data = d;

}

template<typename T>

void Node<T>::set\_next(Node<T>\* n)

{

next = n;

}

template<typename T>

T Node<T>::get\_data()

{

return T(this->data);

}

template<typename T>

Node<T>\* Node<T>::get\_next()

{

return (next);

}

template<typename T>

NStack<T>::NStack() :

node(0), elements(0)

{

}

template<typename T>

NStack<T>::~NStack()

{

Node<T>\* tmp = node;

while (tmp)

{

node = tmp;

tmp = node->get\_next();

delete node;

}

}

template<typename T>

bool NStack<T>::empty()

{

return (this->elements ? false : true);

}

template<typename T>

void NStack<T>::push(T el)

{

Node<T>\* new\_node = new Node<T>();

new\_node = (Node<T>\*)malloc(sizeof(Node<T>));

new\_node->set\_data(el);

new\_node->set\_next(this->node);

this->node = new\_node;

this->elements++;

}

template<typename T>

T NStack<T>::top()

{

return this->node->get\_data();

}

template<typename T>

void NStack<T>::pop()

{

this->elements--;

Node<T>\* tmp = this->node->get\_next();

delete this->node;

this->node = tmp;

}

template<typename T>

T NStack<T>::poll()

{

T tmp;

tmp = this->top();

this->pop();

return tmp;

}

NStack.h

#pragma once

#ifndef NSTACK\_H

#define NSTACK\_H

template <typename T>

class Node

{

public:

Node();

~Node();

void set\_data(T d);

void set\_next(Node<T>\* n);

T get\_data();

Node<T>\* get\_next();

private:

Node\* next;

T data;

};

template <typename T>

class NStack

{

public:

NStack();

~NStack();

bool empty();

void push(T el);

T top();

void pop();

T poll();

private:

Node<T>\* node;

int elements;

};

#endif

Lab2.cpp

// lab2.cpp : Этот файл содержит функцию "main". Здесь начинается и заканчивается выполнение программы.

//

#include <iostream>

#include "AStack.cpp"

#include "NStack.cpp"

#include <string>

#include <stack>

#define NUM 1

#define OP 2

#define ERROR "Ошибка в выражении"

using namespace std;

int is\_num(char c)

{

return (c >= '0' && c <= '9' ? 1 : 0);

}

int is\_op(char c)

{

return (c == '+' || c == '-' || c == '\*' || c == '/' || c == '%');

}

int my\_error()

{

cout << ERROR << endl;

return (-1);

}

int check\_str(string str)

{

int i;

int cur;

int prev;

i = 0;

cur = 0;

prev = 0;

while (str[i])

{

while (str[i] && str[i] == ' ')

i++;

if (str[i] && is\_num(str[i]))

{

while (is\_num(str[i]))

i++;

cur = NUM;

if (prev == cur)

return (my\_error());

prev = cur;

}

else if (str[i] && is\_op(str[i]))

{

cur = OP;

if (prev == cur)

return (my\_error());

prev = cur;

i++;

}

else

return (my\_error());

}

return (0);

}

int prior(char c)

{

switch (c)

{

case '+':

case '-': return 1;

case '\*':

case '%':

case '/': return 2;

}

return (0);

}

template <class T>

string opn(string s, T \*st)

{

string out;

int i;

int j;

i = 0;

j = 0;

while (s[i])

{

if (is\_num(s[i]))

{

out += s[i];

j++;

}

else if (s[i] != ' ')

{

if (j != 0 && is\_num(out[j - 1]))

{

out += ' ';

j++;

}

if (is\_op(s[i]))

{

if (st->empty() || prior(st->top()) < prior(s[i]))

st->push(s[i]);

else

{

while (!st->empty() && (prior(st->top()) >= prior(s[i])))

{

out += st->top(); st->pop();

j++;

}

st->push(s[i]);

}

}

}

i++;

}

while (!st->empty())

{

out += st->top(); st->pop();

}

out += '\0';

return (out);

}

template <class T>

int calculate(string s, T \*st)

{

int i;

int l, r;

char\* endptr;

endptr = NULL;

i = 0;

while (s[i])

{

while (s[i] == ' ')

i++;

if (is\_num(s[i]))

{

int num = strtol(&s[i], &endptr, 10);

st->push(num);

i += endptr - &s[i];

}

else

{

if (is\_op(s[i]))

{

r = st->top(); st->pop();

l = st->top(); st->pop();

switch (s[i])

{

case '+':

st->push(l + r);

break;

case '-':

st->push(l - r);

break;

case '\*':

st->push(l \* r);

break;

case '/':

st->push(l / r);

break;

case '%':

st->push(l % r);

break;

}

}

i++;

}

}

return (st->top());

}

int main()

{

int menu;

AStack<char>\* st1 = new AStack<char>();

NStack<char>\* st2 = new NStack<char>();

stack<char>\* st3 = new stack<char>;

cout << "Какой стек вы хотите использовать?\

\n1. На массиве\

\n2. На односвязном списке\

\n3. На встроенном контейнере" << endl;

cin >> menu;

string s;

cout << "Введите выражение: ";

getchar();

getline(cin, s);

if (check\_str(s) == -1)

exit (0);

string out;

switch (menu)

{

case (1):

out = opn(s, st1);

break;

case (2):

out = opn(s, st2);

break;

case (3):

out = opn(s, st3);

break;

default:

cout << "Что-то пошло нетак" << endl;

exit(-1);

break;

}

cout << "Выражение в постфиксной форме: " << out << endl;

AStack<int>\* st4 = new AStack<int>();

NStack<int>\* st5 = new NStack<int>();

stack <int>\* st6 = new stack<int>;

int res;

switch (menu)

{

case (1):

res = calculate(out, st4);

break;

case (2):

res = calculate(out, st5);

break;

case (3):

res = calculate(out, st6);

break;

}

cout << "Результат выражения: " << res << endl;;

return (0);

}