Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное

учреждение

высшего образования

«**Пермский национальный исследовательский**

**политехнический университет»**

Кафедра «Информационные технологии и автоматизированные

системы»

**ОТЧЁТ**

**по лабораторной работе №19**

Дисциплина: «Информатика»

Тема: “Объектно-ориентированное программирование.

Основные алгоритмы работы со стеками.”

Вариант 13

Выполнил:

Студент группы ИВТ-20-2б

Сафронов Владислав Владиславович

Проверила:

Доцент кафедры ИТАС

Полякова О.А.

**Пермь, 2021**

**Постановка задачи**

Реализовать собственный класс, отражающий работу стека.

**Анализ задачи**

1. Для решения задачи необходимо:
   1. Организовать перегруженный класс Stack с полем vector<T> values, хранящее значения элементов стека.
2. В ходе работы были использованы следующие типы данных:
   1. Перегруженный класс Stack
3. template<typename T>
4. class Stack
5. {
6. vector<T> values;
   1. Конструктор класса Stack с параметром size
7. template <typename T>
8. Stack<T>::Stack(int size)
9. {
10. for (int i = 0; i < size; i++)
11. {
12. values.push\_back(0);
13. }
14. }
    1. Метод top() возвращает значение верхнего элемента стека.
15. template <typename T>
16. T Stack<T>::top()
17. {
18. if (values.size() == 0) throw ("Stack is empty");
19. return values[values.size() - 1];
20. }
    1. Метод push() для добавления элемента в стек.
21. template <typename T>
22. void Stack<T>::push(T& elem)
23. {
24. values.push\_back(elem);
25. }
    1. Метод pop() для удаления верхнего элемента из стека.
26. template <typename T>
27. void Stack<T>::pop()
28. {
29. if (values.size() == 0) throw ("Stack is empty");
30. values.pop\_back();
31. }
    1. Метод size() возвращает размер стека.
32. template <typename T>
33. int Stack<T>::size()
34. {
35. return values.size();
36. }
37. Для операций ввода и вывода использовались следующие операторы и функции.
    1. Операторы << и >>.

**Блок-схема программы**

**Код программы**

**Файл Stack.h:**

1. #pragma once
2. #include <vector>
3. using namespace std;
5. template<typename T>
6. class Stack
7. {
8. vector<T> values;
9. public:
10. Stack() {}
11. Stack(int size);
12. T top();
13. void push(T& elem);
14. void pop();
15. bool empty();
16. int size();
17. };

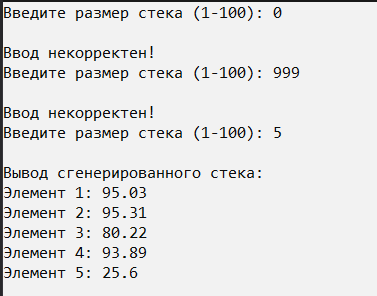
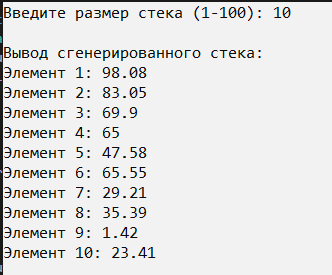
**Файл Stack.cpp:**

1. #include "Stack.h"
3. template <typename T>
4. Stack<T>::Stack(int size)
5. {
6. for (int i = 0; i < size; i++)
7. {
8. values.push\_back(0);
9. }
10. }
12. template <typename T>
13. T Stack<T>::top()
14. {
15. if (values.size() == 0) throw ("Stack is empty");
16. return values[values.size() - 1];
17. }
19. template <typename T>
20. void Stack<T>::push(T& elem)
21. {
22. values.push\_back(elem);
23. }
25. template <typename T>
26. void Stack<T>::pop()
27. {
28. if (values.size() == 0) throw ("Stack is empty");
29. values.pop\_back();
30. }
32. template <typename T>
33. int Stack<T>::size()
34. {
35. return values.size();
36. }

**Файл main.cpp:**

1. #include <iostream>
2. #include "Stack.h"
3. #include "Stack.cpp"
4. #include <ctime>
5. using namespace std;
7. int main()
8. {
9. setlocale(LC\_ALL, "Russian");
10. srand(time(0));
12. int n;
13. Stack<double> stk;
14. cout << "Введите размер стека (1-100): "; cin >> n;
15. while (n < 1 || n > 100)
16. {
17. cout << "\nВвод некорректен!";
18. cout << "\nВведите размер стека (1-100): "; cin >> n;
19. }
20. for (int i = 0; i < n; i++)
21. {
22. double a = rand() % 10000 / 100.0;
23. stk.push(a);
24. }
25. cout << "\nВывод сгенерированного стека: \n";
26. for (int i = 0; i < n; i++)
27. {
28. cout << "Элемент " << i + 1 << ": " << stk.top() << endl;
29. stk.pop();
30. }
32. return 0;
33. }

**Скриншоты результатов работы программы**

****