ФГАОУВО «Южно-Уральский государственный университет (НИУ)»

Институт естественных и точных наук

Кафедра «Прикладная математика и программирование»

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №4

по дисциплине «Объектно-ориентированное программирование»

 Автор работы

 студент группы ЕТ-212

 \_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_А.А. Алексадрова

 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2017 г.

 Работа зачтена с оценкой

 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_А.К. Демидов

 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2017 г.

Челябинск, 2017

# 1 Постановка задачи

# I. Определить класс-шаблон с использованием динамического распределения памяти согласно варианту и необходимые конструкторы и операции, включая конструктор копий, операция присваивания и если указано операцию индексации. При выходе за границу, переполнении и т.п. вызвать исключительную ситуацию (определить собственные классы) для информирования программы, вызвавшей метод.

13. Класс стек элементов заданного типа, размером не более указанного в параметрах конструктора,
добавление << и извлечение >> элемента

II. Реализовать main c тестами (создание объектов и выполнение действий с ними, в т.ч. действие, приводящее к возникновению исключительной ситуации, которую необходимо перехватить)

# 2 Описание интерфейса класса

struct stackerror {

 virtual ~stackerror() {}

 virtual const char \*message() const=0;

};

struct stackempty: stackerror {

 const char \*message() const {return "Стек пуст";}

};

struct stackfull: stackerror {

 const char \*message() const {return "Стек полон";}

};

template <typename T>

class Stack {

 T \*a;

 int col,size;

public:

 Stack(int size) throw(bad\_alloc): col(0), size(size),a(new T[size]) {}

 ~Stack() throw() {delete []a;}

 Stack(const Stack<T> &) throw(bad\_alloc);

 Stack<T> &operator=(const Stack<T> &) throw();

 Stack<T> &operator<<(const T &) throw (stackfull);

 Stack<T> &operator>>(T &)throw (stackempty);

};

# 3 Описание тестов для проверки классов

# int main()

# {

#  Stack<int> obj(10);

#  try {

#  for (int i=0;; i++) {

#  obj<<i;

#  }

#  }

#  catch (stackerror &e) {

#  cout<<e.message();

#  }

#  cout<<"\nИзвлекаем из стека:\n";

#  try {

#  while (1) {

#  int temp;

#  obj>>temp;

#  cout<<temp;

#  }

#  }

#  catch (stackerror &e) {

#  cout<<"\n";

#  cout<<e.message();

#  }

#  return 0;

# }

**Полученные результаты:**

Тест 1. Добавление

Стек полон

Тест 2. Извлечение

Извлекаем из стека:
987654321
Тест 3.

Стек пуст

# 4 Листинг реализации класса

template <typename T>

Stack<T>:: Stack(const Stack <T> &c)throw(bad\_alloc):a(new T[c.size]),col(c.col),size(c.size) {

 for (int i=0; i<col; i++)

 a[i]=c.a[i];

}

template <typename T>

Stack<T> &Stack<T> ::operator<<(const T &x) throw(stackfull) {

 if (col==size) throw stackfull();

 a[col++]=x;

 return \*this;

}

template <typename T>

Stack <T> &Stack<T>::operator>>(T &x) throw(stackempty) {

 if (col==0) throw stackempty();

 x=a[--col];

 return \*this;

}