Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное

образовательное учреждение высшего образования

Национальный исследовательский Нижегородский государственный университет им. Н.И. Лобачевского

Институт информационных технологий, математики и механики

**Отчет по лабораторной работе**

**«Вычисление арифметических выражений»**

**Выполнил**:

студент группы 382003-1

Семенова В. А.

**Проверил**:

ассистент каф. МОСТ,

Волокитин В. Д.

Нижний Новгород

2021

Содержание

[Постановка задачи 3](#_Toc91271053)

[Метод решения 4](#_Toc91271054)

[Руководство пользователя 5](#_Toc91271055)

[Описание программной реализации 6](#_Toc91271056)

[Подтверждение корректности 7](#_Toc91271057)

[Заключение 8](#_Toc91271058)

[Приложение 9](#_Toc91271059)

# Постановка задачи

Необходимо разработать программу, выполняющую вычисление арифметического выражения с вещественными числами. Выражение в качестве операндов может содержать переменные и вещественные числа. Допустимые операции известны: +, -, /, \*. Допускается наличие знака "-" в начале выражения или после открывающей скобки. Опционально - наличие математических функций (sin, соs, ln, exp, и т.д.) Программа должна выполнять предварительную проверку корректности выражения и сообщать пользователю вид ошибки и номера символов строки, в которых были найдены ошибки.

Этапы выполнения работы:

1. Разработка шаблонного класса TStack
2. Разбиение исходного арифметического выражения на лексемы (т.е. выделить операнды, операции, переменные)
3. Проверка корректности выражения:

* правильность расстановки скобок (с использованием стека (TStack) для хранения индексов скобок).
* пропущены ли операнды или знаки операций
* недопустимые символы

1. Перевод выражения в постфиксную (польскую) запись
2. Вычисление выражения по постфиксной записи
3. Создание консольного интерфейса пользователя
4. Написание отчета

Все функции должны тестироваться с помощью Google test-ов.

# Метод решения

Класс Arithmetic предназначен для хранения операндов и операций.

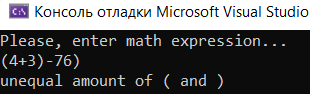
* 1. Конструктор по умолчанию, конструктор копирования и деструктор предназначены для создания и удаления объектов класса соответственно.
  2. Переопределен оператор = c учетом входных параметров.
  3. Функции double GetX() const и char GetOp() const возвращают операнд и операцию соответственно. Функция bool GetF() const возвращает true, если объект класса является операндом, и false, если операцией.
  4. Функция int GetPriority() возвращает приоритет операции, коей является данный объект класса.
  5. Функция double res(Arithmetic\* m, int n) вычисляет значение выражения, записанного в обратной польской записи по следующему алгоритму:
     + Если на вход подано число, оно помещается в стек.
     + Если на вход подан знак операции, соответствующая операция выполняется над требуемым количеством операндов, извлекаемых из стека, взятых в порядке добавления, а результат данной операции кладется обратно в стек.
     + Предыдущие два шага выполняются, пока входной набор символов обработан не полностью.
     + После полной обработки выражения результат вычислений лежит на вершине стека.
  6. Функция double result(string s) переводит введенную строку в массив символов (вещественных чисел – операндов и операций). Допускает работу с переменными, значения которых могут быть введены только единожды и проверяются на корректность. Производится проверка на ввод недопустимых символов. После преобразования строки в массив символов и проверки его корректности функцией bool aaa(Arithmetic\* m, int n) происходит перевод выражения в инфиксной форме записи в постфиксную форму записи с использованием стека. Перевод производится по следующим правилам:
     + Если символ – число, помещаем его в массив pol.
     + Если символ « ( », помещаем его в стек.
     + Если символ « ) », извлекаем из стека все операции до « ( » и помещаем в массив pol, саму « ( » просто извлекаем из стека.
     + Если символ – операция, выталкиваем из стека все операции, приоритет которых не ниже приоритета данной операции, после чего саму операцию записываем в стек.
     + При достижении конца массива m, если стек не пуст, извлекаем все элементы из него в массив pol.

Далее вызывается функция double res(Arithmetic\* m, int n), выдающая результат вычислений.

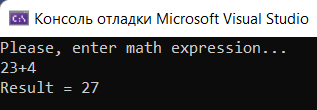
* 1. Функция bool aaa(Arithmetic\* m, int n) проверяет корректность введенного выражения на соответствие математической логике и, соответственно, возможность вычисления. В случае, если найдена какая-либо ошибка, выводится сообщение о характере ошибки и, если ошибка заключается в вводе недопустимого символа, данный символ.

# Руководство пользователя

Пользователю предлагается ввести выражение в инфиксной форме записи. Предполагается, что выражение вводится без пробелов. Допустимые операции: + - \* / ( ). В случае синтаксической ошибки на экран выводится тип ошибки.



В случае корректности введенного выражения программа выводит результат вычислений на экран.



# Описание программной реализации

Программа представлена 4 проектами:

1. simple:

main\_arithmetic.cpp – реализация пользовательского интерфейса программы.

1. arithmetic:

arithmetic.h – объявление класса arithmetic, его полей и методов.

arithmetic.cpp – реализация методов класса Arithmetic.

stack.h – объявление класса шаблонного TStack, реализация его методов, необходимых для работы программы.

1. gtest:

библиотека Google test-ов

1. tests:

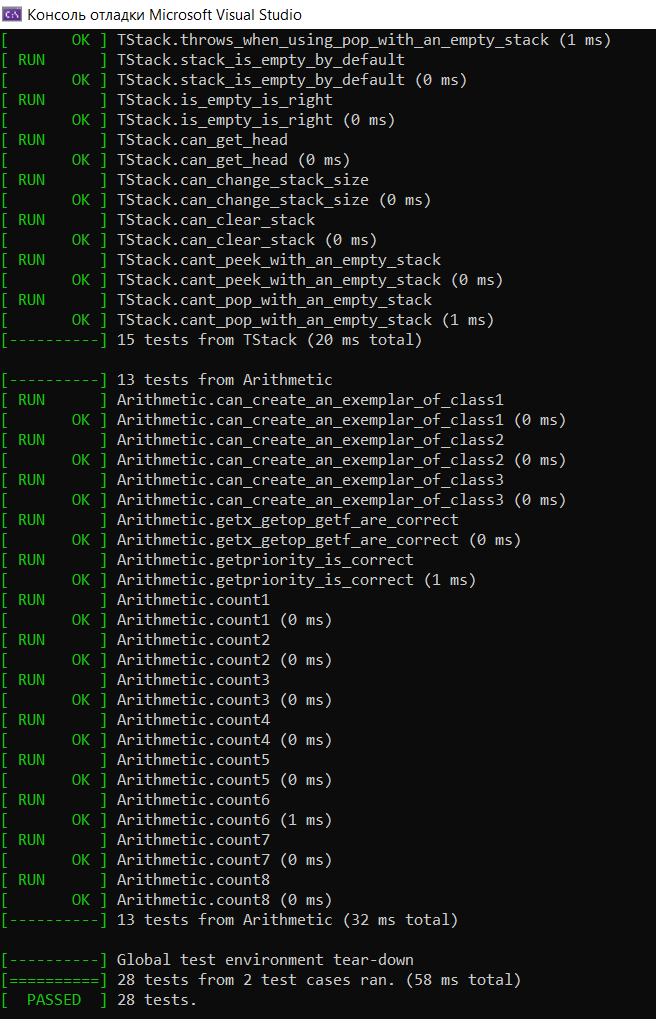
test\_arithmetic.cpp – тесты для класса Arithmetic.

test\_main.cpp – запуск работы тестов.

test\_stack.cpp - тесты для класса TStack.

# Подтверждение корректности

Для подтверждения корректности работы программы используются Google test-ы.

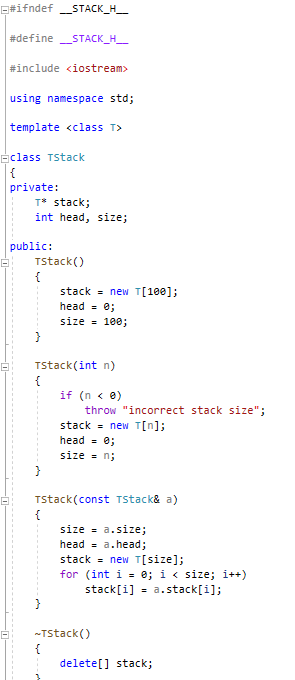


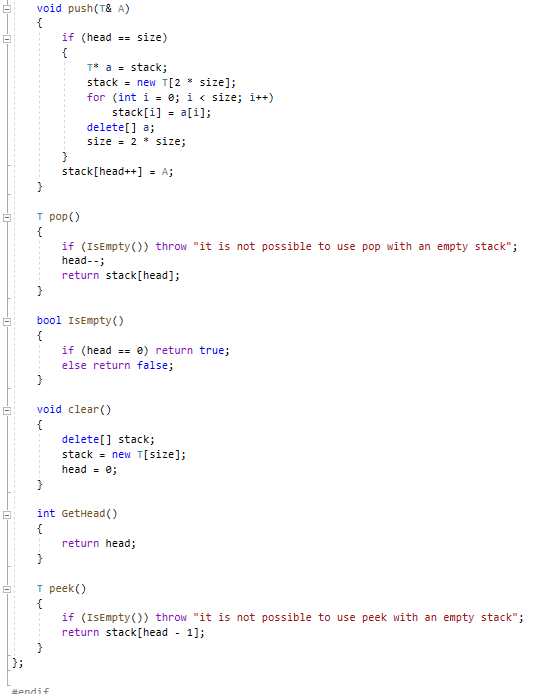
# Заключение

В результате выполнения лабораторной работы был реализован класс TStack для работы со стеком и класс Arithmetic для перевода выражения из инфиксной формы записи в обратную польскую запись и вычисления выражения, записанного в постфиксной форме записи.

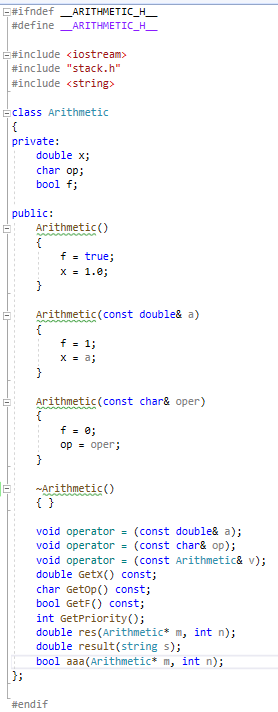
# Приложение

Класс TStack

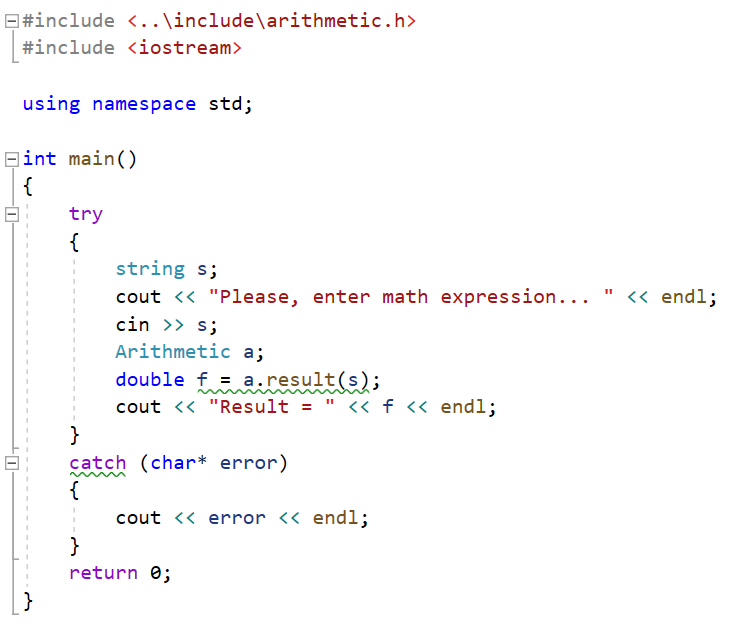




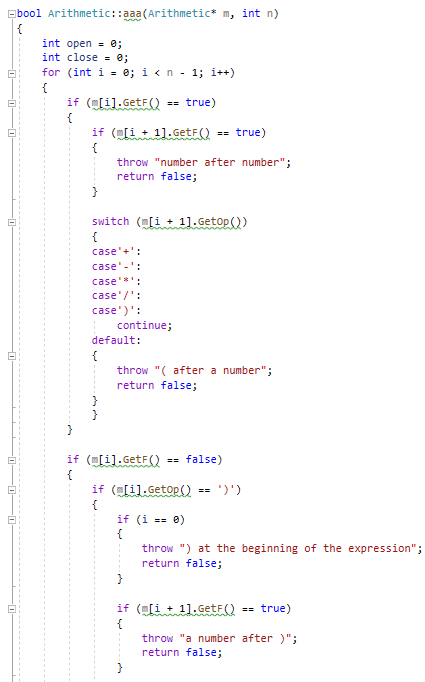
Класс Arithmetic

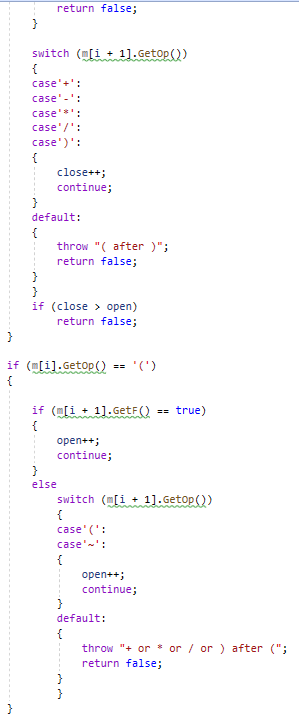


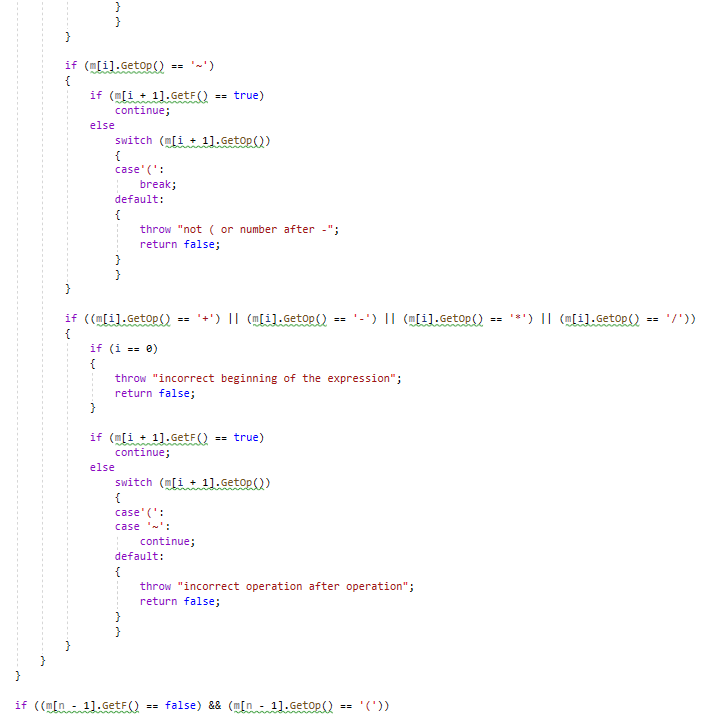
Пользовательское приложение

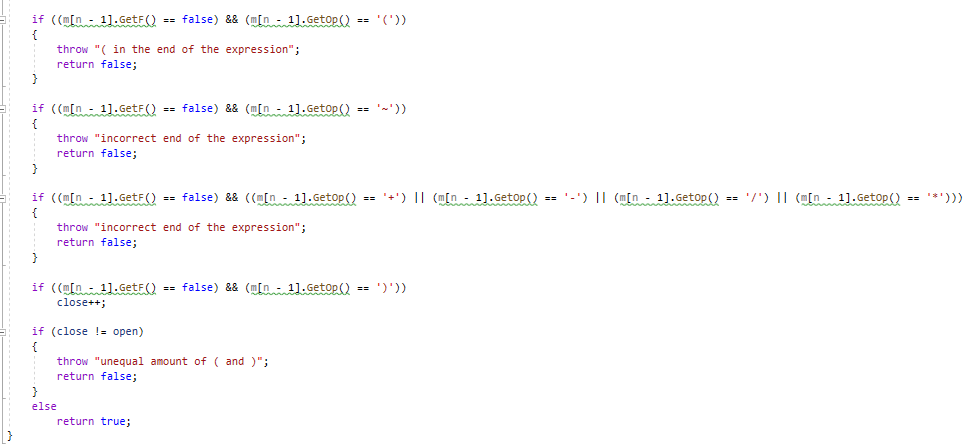


bool aaa(Arithmetic\* m, int n)

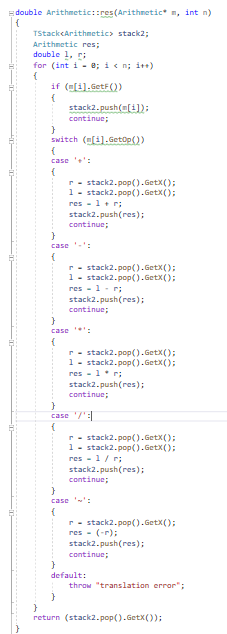








double res(Arithmetic\* m, int n)



double result(string s)

