Министерство образования Республики Беларусь

Учреждение образования «Белорусский государственный университет информатики и радиоэлектроники»

Факультет компьютерных систем и сетей

Кафедра информатики и технологий программирования

Дисциплина: Программирование

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовому проекту

на тему

СТРОКОВЫЙ ИНЖЕНЕРНЫЙ КАЛЬКУЛЯТОР

Студент: гр. 553501 Козлова А.А.

Руководитель: Казак Д. Г.

Минск 2016

**СОДЕРЖАНИЕ**

Введение……………………………….…….…………………………………………...3

1. Анализ требований к ПС и разработка функциональных требований…………...4
   1. Требования к ПС………………………………………………………..………..4
   2. Разработка функциональных требований………………………………….…..4
2. Проектирование программного средства…………………………………………..5
   1. Особенности программного средства…………………………………………..5
   2. Интерфейс программы…………………………………………………………..6
3. Создание программного средства…………………………………………….……10
   1. Основные функции и процедуры……………………………………………...10
   2. Схема работы программы……………………………………………………...12

Заключение………………………………………………………………………….…..13

Список литературы……………………………………………………………………..14

**ВВЕДЕНИЕ**

Калькулятор — электронное вычислительное устройство для выполнения операций над числами или алгебраическими формулами. В зависимости от возможностей и целевой сферы применения калькуляторы делятся на простейшие, бухгалтерские, инженерные (научные), финансовые. В отдельные классы обычно выделяют программируемые калькуляторы, дающие возможность выполнения сложных вычислений по предварительно заложенной программе, а также графические — поддерживающие построение и отображение графиков.

Целью данного курсового проекта является создание программы "Строковый инженерный калькулятор".

Основной задачей программного этого проекта является выполнение математических вычислений, направленное на облегчение работы пользователя. Данная программа может быть использована как электронный калькулятор, предназначенный для вычисления различных математических функций. Программа преобразовывает введенную строку в выражение и выводит результат расчетов. Можно производить умножение и деление, сложение и вычитание, возведение в степень. Также программа способна вычислять целый ряд функций, таких как косинус, синус, тангенс, десятичный логарифм и другие функции.

Для получения результата необходимо, чтобы разрабатываемое программное средство выполняло следующие функции:

* Проверка введенного выражения
* Разбиение выражения на отдельные функции
* Подсчет результата с учетом приоритета операций
* Вывод результат

Для разработки приложения была выбрана среда программирования Visual Studio 2013, позволяющая разработать приложение с графическим интерфейсом с поддержкой технологии Windows Forms. Программа написана на языке программирования С++.

**1. АНАЛИЗ ТРЕБОВАНИЙ К ПС И РАЗРАБОТКА ФУНКЦИОНАЛЬНЫХ ТРЕБОВАНИЙ**

**1.1. Требования к ПС**

Работа в приложении осуществляется с помощью кнопок и поля ввода. Программа имеет инженерный режим работы. В нем выполняются такие функции как возведение в степень, факториал, логарифм, тригонометрические вычисления и т.п. Все введенные углы будут считываться в радианах.

Для работы приложения необходим только исполняемый файл (Проект1.exe).

**1.2. Разработка функциональных требований**

Проанализировав существующие программные решения, были определены следующие функциональные требования к разрабатываемой программе:

1. соблюдение правильности вычислений;
2. программа должна иметь простой, но в то же время понятный и наглядный интерфейс, который не должен перегружать ресурсы компьютера;
3. программа должна иметь возможность сброса полученного результата;
4. пользователь должен иметь возможность видеть выполняемые им действия и полученный результат;
5. программа не должна занимать большой объем памяти и не должна требовать установки на жесткий диск компьютера;
6. должна существовать возможность вычисления основных тригонометрических функций (синус, косинус, тангенс, котангенс), извлечение квадратного корня, а также возведение числа в целую степень;
7. работоспособность приложения в среде Windows.

Перед расчётом ответа, программа должна:

* проверять правильность написанного выражения
* удалять лишние скобки
* добавлять недостающие скобки
* при обнаружении ошибки в написании какой-либо функции останавливать работу и выдавать соответствующее сообщение.

Программа должна выдавать сообщение об ошибке, если строка заканчивается на:

**+ - \* / . (**

**2. ПРОЕКТИРОВАНИЕ ПРОГРАММНОГО СРЕДСТВА**

**2.1. Особенности программного средства**

Приложение «Инженерный калькулятор» имеет ряд требований, необходимых для удобства использования:

1. Использование поля для ввода
2. Проверка на логичность ввода, т.е. программа не должна позволять пользователю вводить подряд >1 математической операции; невозможность писать после “.” чего либо, кроме цифр и т.д. Это сделано для уменьшения ошибок, допускаемых пользователем.
3. Расширенный набор действий: вычисление тригонометрических функций, возведение в степень и т.п.
4. Использование различных типов переменных
5. Вывод сообщений при делении на 0, извлечении корня из отрицательного числа, а также вычислении логарифма отрицательного числа и т.д.
6. Необходимые формулы:

* xy=exp(y\*ln(x))
* ctg(x)=1/tan(x)

Программа должна справляться с:

* lg
* ctg
* ln
* tg
* sin
* cos
* sh
* ch
* th
* arcsin
* arcos
* arctg
* сложением
* вычитанием
* умножением
* делением
* возведением в степень
* факториалом
* рассчитывать ответ с учетом ( )

**2.2. Интерфейс программного средства**

Для повышения удобства пользования программой разработан понятный графический интерфейс, то есть все возможные операции, которые пользователь может производить в программе, выведены непосредственно на экран пользователя.

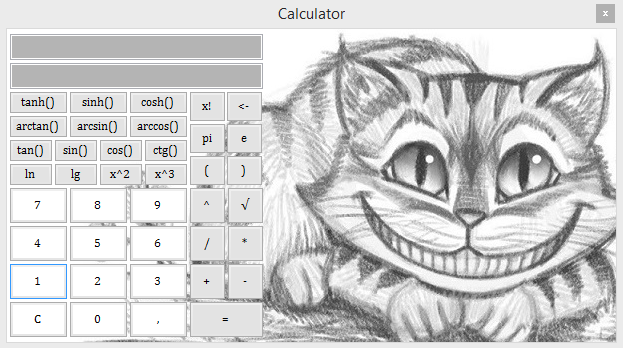
****

Рисунок 1.1 – Окно “Calculator”

**Поле 1**

****

**Поле 2**

* поле 1 отображает вводимую строку;
* поле 2 отображает результат введенного выражения;



* добавляет в поле1 **tanh(**
* рассчитывает гиперболический тангенс значения выражения в скобках



* добавляет в поле1 **sinh(**
* рассчитывает гиперболический синус значения выражения в скобках



* добавляет в поле1 **cosh(**
* рассчитывает гиперболический косинус значения выражения в скобках



* добавляет в поле1 **arctan(**
* рассчитывает арктангенс значения выражения в скобках



* добавляет в поле1 **arcsin(**
* рассчитывает арксинус значения выражения в скобках



* добавляет в поле1 **arccos(**
* рассчитывает арккосинус значения выражения в скобках



* добавляет в поле1 **tan(**
* рассчитывает тангенс значения выражения в скобках



* добавляет в поле1 **sin(**
* рассчитывает синус значения выражения в скобках



* добавляет в поле1 **cos(**
* рассчитывает косинус значения выражения в скобках



* добавляет в поле1 **ctg(**
* рассчитывает котангенс значения выражения в скобках



* добавляет в поле1 **lg(**
* рассчитывает десятичный логарифм значения выражения в скобках



* добавляет в поле1 **ln(**
* рассчитывает натуральный логарифм значения выражения в скобках



* добавляет в поле1 **^(2)**
* нужна для возведения значения, стоящего перед знаком степени, в квадрат



* добавляет в поле1 **^(3)**
* нужна для возведения значения, стоящего перед знаком степени, в куб



* добавляет в поле1 **!**
* нужна для расчета факториала выражения, стоящего перед знаком **!**



* удаляет последний элемент строки в поле1



* добавляет в поле1 **pi**



* добавляет в поле1 **e**



* добавляет в поле1 **(**
* нужна для возведения числа в степень



* добавляет в поле1 **)**
* нужна для осуществления операции деления



* добавляет в поле1 **^**
* нужна для возведения числа в степень



* добавляет в поле1 **√(**
* нужна для извлечения квадратного корня из выражения стоящего в скобках



* добавляет в поле1 **/**
* нужна для осуществления операции деления



* добавляет в поле1 **\***
* нужна для осуществления операции умножения



* добавляет в поле1 **+**
* нужна для осуществления операции сложения



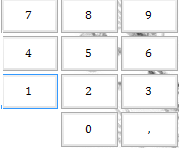
* добавляет в поле1 **─**
* нужна для осуществления операции вычитания



* подсчитывает результат и отображает его в поле2



* очищает поле1



* обеспечивает ввод цифр

**3. СОЗДАНИЕ ПРОГРАММНОГО СРЕДСТВА**

**3.1. Основные функции**

1. static String^ checkBrackets(String^& inputString)  
   Считает в строке **inputString** **“(“** и **“)”**. Если **“(“** больше, то к написанному выражению дописываются недостающие **“)”**. Если **“)”** больше, то в написанной строке удаляются лишние **“(“**. Возвращается исправленная строка.
2. static double lg(double x)  
   Сравнивает значение аргумента с 0, если меньше или равно, то останавливает вычисления и пишет сообщение об ошибке.
3. static double ln(double x)  
   Сравнивает значение аргумента с 0, если меньше или равно, то останавливает вычисления и пишет сообщение об ошибке.
4. static double ctg(double x)  
   Вычисляет котангенс.
5. static double arcsin(double x)

Проверяет принадлежит ли аргумент арксинуса промежутку [-1;1], если принадлежит вычисляет арксинус, если не принадлежит, то пишет сообщение об ошибке и останавливает расчет ответа.

1. static double arccos(double x)  
   Проверяет принадлежит ли аргумент арккосинуса промежутку [-1;1], если принадлежит вычисляет арккосинус, если не принадлежит, то пишет сообщение об ошибке и останавливает расчет ответа.
2. static double factorial(double x)  
   Сравнивает значение аргумента с нулем и проверяет на целое значение, если меньше 0 или аргумент не является целым числом, то останавливает вычисление и пишет соответствующее сообщение об ошибке. В случае успеха пройденных проверок, функция возвращает факториал числа.
3. static double sqrt(double x)  
   Сравнивает значение аргумента с нулем, если меньше 0, то останавливает вычисление и пишет сообщение об ошибке. В ином случае, функция возвращает квадратный корень из числа.
4. static double powerIt(double multiplyElement, String^& inputString)  
   Возводит в степень, если это возможно. В ином случае возвращает 0 и сообщение об ошибке.
5. static double convertFromStringToDouble(String^& inputString)  
   При возврате этот метод содержит число двойной точности с плавающей запятой, эквивалентное параметру **inputString**, если преобразование завершилось успешно, или нуль, если оно завершилось неудачно. Преобразование завершается неудачно, если значение параметра **inputString**  равно **null**, не является числом допустимого формата или представляет число меньше **MinValue** или больше **MaxValue**. Этот параметр передается неинициализированным; любое значение, первоначально предоставленное в объекте **result** будет перезаписано.
6. static double getNumberFromString(String^& inputString)  
   Считывает из строки число.
7. static bool isCharFoundInString(wchar\_t currentChar, String^ stringForSearching)  
   Функция ищет вхождение символа **с** в строку **stringForSearching**, если символ найден, то функция возвращает **true**, в ином случае – **false**.
8. static double getNextElement(String^& inputString);  
   Функция, которая находит самую вложенную функцию или выражение, рассчитывает его и заменяет числом. Если функция не была найдена, то возвращается соответствующее сообщение об ошибке.
9. static double multiplyIt(String^& inputString, double multiplyElement)  
   Выполняет операции умножения, деления. Если делитель равен 0, то выводится сообщение об ошибке.
10. static bool isPrevFunc(String^& inputString, String^ searchingFunction)  
    Сравнивает введенную функцию **inputString** с фиксированным названием прописанной функции **searchingFunction**. Если **inputString** сравнимо с **searchingFunction**, то возвращается **true**, в ином случае **false**.
11. static double mainFunction(String^& inputString);  
    Основная функция программы. Выполняет операции сложения, вычитания. Если нужно, запускает функции multiplyIt, powerIt. Возвращает конечный результат вычислений. Вызывается при нажатии на кнопку **“=”**.
12. static String^ checkError()

Если в ходе выполнения программы произошла ошибка, то эта функция вернет соответствующую ошибку.

**3.2. Схема работы программы**

Введена строка?

Вывод сообщения «write expretion»

Проверка строки

Завершен ввод строки?

Вывод сообщения «error in conditions»

Проверка скобок

Расчет результата

Ошибка в выражении?

Вывод обнаруженной ошибки

Вывод результата

Рисунок 1.2 Алгоритм обработки выражения после нажатия пользователем “=”

**Заключение**

В результате выполнения данного курсового проекта был разработан программный продукт “Calculator”. При помощи этой программы можно высчитывать определенные арифметические операции. Программа не занимает много места, не требовательна к установленному программному обеспечению.

“Calculator” обладает следующими достоинствами:

* Возможность ввода в строку всего выражения.
* Логические ограничения на начальном этапе ввода выражения, чтобы не допустить нежелательных ошибок при обработке выражения.
* Добавление или удаление открывающих и закрывающих скобок, в случае неравенства их количества
* При невозможности обработки результата, программа выводит ошибку с сообщением, где именно введено недопустимое выражение.

В целом, поставленная в начале курсового проекта цель была достигнута. В программе выполняются все необходимые функциональные требования.

**Список литературы**

1. Интерактивный учебник по Visual C++ (<https://msdn.microsoft.com/ru-ru/library/ms235630(v=vs.120).aspx)>
2. Шилдт, Герберт. Полный справочник по C++, 4-е издание. Пер. с англ. — М. : Издательский дом “Вильямс”, 2006. — 800 с. : ил. — Парад, тит. англ. ISBN 5-8459-0489-7 (рус.).
3. Visual C++ 2010. Полный курс. Автор: Айвор Хортон. Издательство: Вильямс Год: 2011
4. Microsoft Visual Studio 2010. Автор: Голощапов А.Л. Издательство: Вильямс. Год: 2011.