Department of Computer Engineering

University of Puerto Rico

Mayaguez Campus

# CASOLUS – Calculus Solver Programming Language

## *Project Plan*

Team Members:

Kevin Santiago Ortiz

Carlos J. Ojeda Malavé

Cristian Meléndez González

Christian J. Cancel Ramos

# INTRODUCTION

## Motivation

Due to the difficulty of the Calculus courses, most students drop out or end up failing. Two reasons for this are that it is too expensive to pay for a solutions manual or that it is difficult to find the solutions on the web. There are a lot of books that provide the solutions to some of the problems, but they are only for the odd numbered problems. These are the reasons why our team is motivated to create a tool that gives students the support they need when solving those mathematical problems. By creating a new programming language, we will be providing an irreplaceable tool for students, allowing them to place their equations, in a simple way, to receive a quick solution and thus ensuring that every student receives the correct answers to their problems.

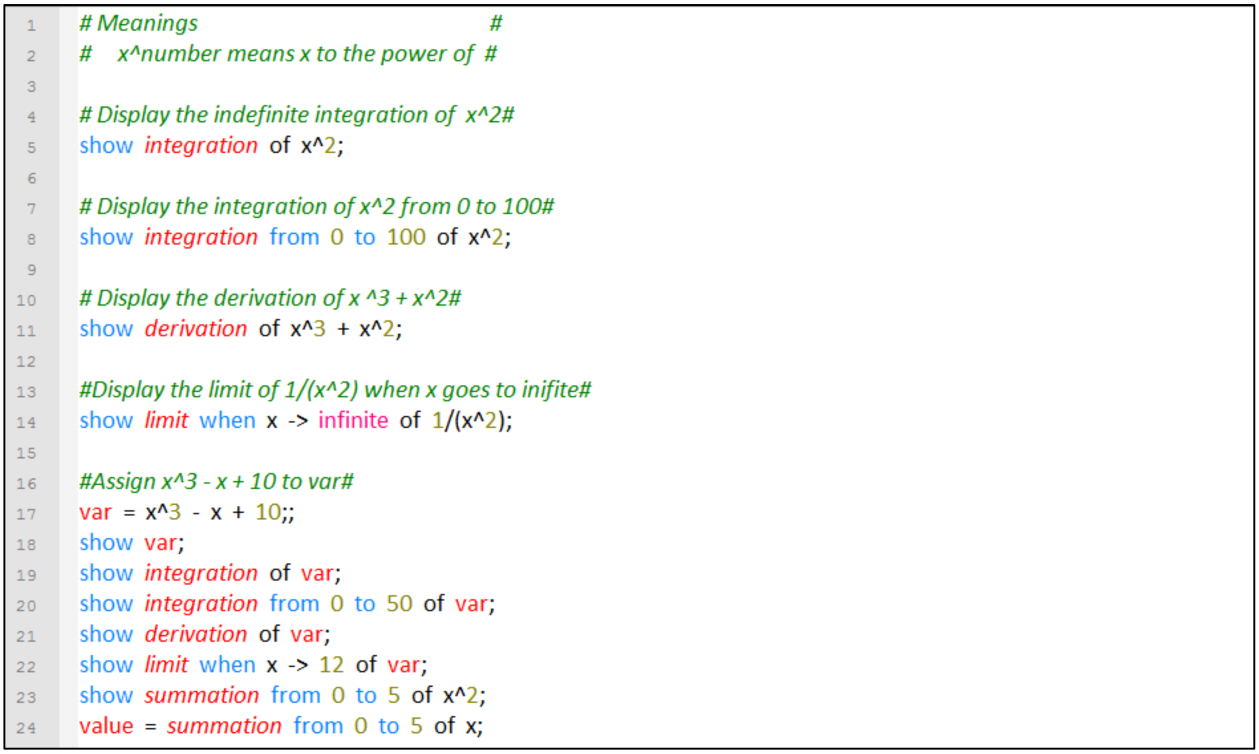
# FEATURES

## Concept

Casolus is a calculus solver programming language developed with the purpose of giving the students a simple tool to solve calculus problems. Our focus with this project is to provide a simple way to express mathematical equations such that everyone, even without knowing about programming, could learn it easily and quickly due to the language syntax is similar to a common language. Casolus will be able to solve simple operations and some complex operations. The simple operations will be sum, subtraction, division, multiplication, and powers while the complex operations will be integration, derivation, limits, summation notation, and product notation.

# PROGRAM EXAMPLE

The grammar for the code will looks like the following image.



```
# Meanings                                #
#    x^number means x to the power of  #

# Display the indefinite integration of  x^2#
show integration of x^2;

# Display the integration of x^2 from 0 to 100#
show integration from 0 to 100 of x^2;

# Display the derivation of x ^3 + x^2#
show derivation of x^3 + x^2;

#Display the limit of 1/(x^2) when x goes to inifite#
show limit when x -> infinite of 1/(x^2);

#Assign x^3 - x + 10 to var#
var = x^3 - x + 10;;
show var;
show integration of var;
show integration from 0 to 50 of var;
show derivation of var;
show limit when x -> 12 of var;
show summation from 0 to 5 of x^2;
value = summation from 0 to 5 of x;
```

Running the code above will result in several outputs showing the answer for the operations requested with the “show” command. But, this is not every command in the Casolus grammar, so following is a table with all commands.

| Command | Command Attributes |
|---|---|
| show | Display the result of an operation if the operation returns a value |
| integration | Could have two attributes, the range and the equation to or only the equation to integrate, this is for indefinite integration and definite integration respectively |
| derivation | Have one attribute and is the expression to be derivate |
| summation | Have two attributes, the range and the equation |
| product | Known as product notation, have two attributes, the range and the equation |
| limit | Have two attributes, the approach and the equation |

| Operators | Operators Attributes |
|---|---|
| **difference/minus** | **Will be able to handle multiple numbers. It will return the difference of the numbers that are inputted.** |
| **sum/plus** | **Will be able to handle multiple numbers. It will return the sum of the numbers that are inputted.** |
| **symbolic** | **Will have one attribute. It will be the expression with only symbolic expressions.** |
| **multiplication** | **Will be able to handle multiple numbers. It will return the multiplication of the numbers that are inputted.** |
| **divide** | **Will be able to handle multiple numbers. It will return the result of the division of the numbers that are inputted.** |
| **a^b** | **Returns the result of a to the power b** |

# IMPLEMENTATION REQUIREMENTS AND TOOLS

## Software Requirements

- The language should receive as input Calculus mathematical operation
- The language should solve differentiation problems
- The language should solve limit problems
- The language should solve definite integrals problems
- The language should solve indefinite integrals problems
- The language should solve simple mathematics problems ( + , - , /  and *)
- The language should solve polynomials
- The language should read from left to right
- The language should follow PEMDAS (Parentheses, Exponent, Multiplication, Division, Addition, Subtraction)
- The language should associate words with mathematical expressions (i.e the “derivative”)

## Tools

- **Python -** the user will need at least python version 2.7 installed in their computers to be able to run our calculus language.
- **PyCharm -** is an Integrated Development Environment(IDE) used to program Python. We will be using this program as our main IDE when creating our programming language.
- **SymPy -** is a Python library for symbolic mathematics. We will be using Sympy in our intermediate code so that the language can handle complex calculus equations.
- **Lex -** Lex is a software that generates the lexical analyzer designed for processing of character input streams.
- **Yacc -** Yacc is a LALR(Look Ahead Left-to-Right) parser generator that will be used for our syntax analyzer.
- **PLY -** We will be using PLY, and implementation of Lex and Yacc, for our lexical and syntax analyzers.

# PROJECT PLAN AND TIMELINE

| Task Name | Duration | Start | Finish |
|---|---|---|---|
| Design | 3 days | 21/09/2016 | 23/09/2016 |
| Lexical Analyzer | 10 days | 26/09/2016 | 07/10/2016 |
| Syntax Analyzer | 10 days | 07/10/2016 | 20/10/2016 |
| Intermediate Code implementation | 9 days | 20/10/2016 | 01/11/2016 |
| Testing | 5 days | 01/11/2016 | 07/11/2016 |
| Documentation | 34 days | 26/09/2016 | 07/11/2016 |
| Final Report | 6 days | 07/11/2016 | 13/11/2016 |