# Compiler Design

Milestone 1

April 5, 2019

## Group Members:


1. Nidanshu Arora (160443) - nidanshu@iitk.ac.in
2. Niket Agrawal (160446) - niketagl@iitk.ac.in
3. Siddharth Chinmay (160685) - schinmay@iitk.ac.in


## Language :

1. The Source Language **S** for our compiler is **C+**.
2. Some features that distinguishes **C+** from vanilla "C" language are:
   (a) Function Overloading
   (b) Class Implementation (Basic)
3. The Implementation Language **I** is **C++** and the Target Language is **x86-64**.

# Project Desciption :

We have constructed a scanner and a parser for our source language **C+** that outputs the abstract syntax tree (AST) for each external declaration (function, struct, etc.) of input **C+** program in a graphical form. We have used Graphviz tool to draw the tree.

1. We are tokenizing the input code and sending the tokens to the parser. We have referred to this lexer - ANSI C grammar, Lex specification
2. We have modified this grammar - ANSI C Yacc grammar
3. We have added actions corresponding to each rule in our grammar to generate nodes and make the Abstract Syntax tree.
4. We have processed the trees generated by the actions added in the grammar to output the postscript of the dot file to build the tree.

# Steps to build and run the Project:

$ cd <project-top>
$ make
$ ./myASTGenerator <input file path> –out=<output dot file path>
$ dot -Tps <dot file path> -o <output ps file path>

Note: Please make sure you have correct versions of Flex and Bison installed :
flex : 2.6.0
bison : 3.0.4