# Assignment C1

## 1 Problem Definition

Code generation using iburg tool

## 2 Learning Objectives:

1. To understand code generation using tool

## 3 Prerequisites:

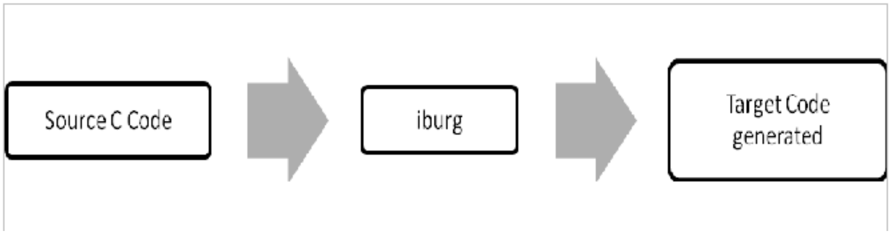
1. Concept of code generation
2. Concept of C compilation

## 4 S/W and H/W requirements:

1. Open source 64 bit OS.
2. 64-bit Intel-i5/ i7 or latest higher processor computers
3. iburg

## 5 Theory

In Compiler, many code generator generators use tree pattern matching and dynamic programming. Iburg accept tree patterns and associated costs and semantic actions that, for example, allocate registers and emit object code. Iburg produce tree matchers that make two passes over each subject tree. The first pass is bottom up and finds a set of patterns that cover the tree with minimum cost. The second pass executes the semantic actions associated with minimum-cost patterns at the nodes they matched. Code generator generators based on this model include BEG, Twig, and burg. BEG matchers are hard-coded and mirror the tree patterns in the same way that recursive-descent parsers mirror their input grammars. Iburg is developed in Ansi C and support C codes.



# 6 Related Mathematics

Let S be the solution perspective of the given problem.
The set S is defined as:
$S = \{\ s, e, X, Y, F, Su, Fl|\varnothing_s\}$
Where,
s= Start point
e= End point
F= Set of main functions
DD= set of deterministic data
NDD= set of non deterministic data
X= Input Set.
$X = \{\ L\}$
where,
L= { $[A - Z]U[a - z]U[0 - 9]U+, -, *, /, \hat{a}.\}$// Code Language
Y = $\{L\}$
Su = Students are able to generate code using iburg
Fl = Code generation get failed

# 7 Conclusion

We are able to understand process of code generation and can observe code generated by iburg and GCC.