Requirement Analysis Document

Note: all the highlighted part should be submitted by November 10th

# **Project name: Go language Parser**

# **Team members: 정재우(20215184), 남동준(20195053)**

# **Team name: Naming variable is too hard**

# **Scope of the project**

**It receives and interprets codes that fit part of the Go program grammar, and outputs execution results. If an error occurs, an appropriate error message is output.**

**- Keywords (20/25)**

ex)

import, package

var, const, (int, bool)

if else else if

switch case default

for, break, continue, range

func return

**- Operator (22/48)**

**ex) Arithmetric operation, Logical operation, Relational operation + a**

**ex) +,-,\*,/,%,(,) for Arithmetric / &&,||,! for Logical / <,>,<=,>=,==,!= for Relational**

**ex) =, +=, -=,\*=,/=,%= for Extra**

**- Types (5/11)**

**ex) Boolean, Integer(Not float), array, function, string**

**- Identifiers**

**- Constants**

**ex) Integer, true, false, string of character**

# **Representative diagram to explain your project**

Make a block diagram explaining modules of your project. In the modules, you can write

major functions of your code

**Tools used : Lex & Yacc**

**Input : String of code**

**Output (proper error message) : Error occured or not. If error occured, print proper error message**

**Ex)**

**Hello World**

**package main**

**import "fmt"**

**func main() {**

**fmt.Println("Hello World")**

**}**

Output

Input

Input

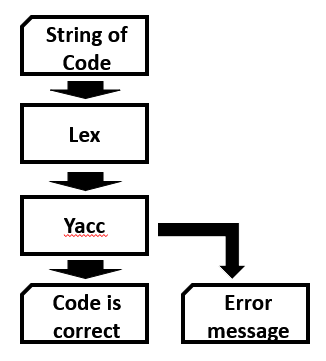


Fig 1. Diagram of Go parser

# Limitations of your project

If your project can not parse the standard input format you should write about that.

For ex: HTML parser -> this can parse only following tags and properties r

SQL parser -> Can only parse select and update statement without if conditions

# Sample input and output