Compilers

Phase 1

Team 10

Name	Sec	Bn	Email
Khaled Galal Helmy	1	16	khaled.elnomrosy98@eng-st.cu.edu.eg
Muhammad Ayman	2	11	muhamed.sadek97@eng-st.cu.edu.eg
Yousif Gamal	2	34	Yousif.Ahmed99@eng-st.cu.edu.eg
Youssef Mohamed Ahmed Dawood	2	35	yousef.dawood03@eng-st.cu.edu.eg

Assumptions:

- Lexical tokens:
  - Strings are quoted by double quotes only for example: 'ayman' is invalid
  - No double quote inside two double quotes for example: "ayman"mh" is invalid
  - Backslash at the end of the string is invalid for example: "ayman\"
  - Single quotes inside two double quotes is valid for example: “‘ayman’”
  - Newline,intend,carriage return characters are valid for example: “ayman\r\t\nmuhammad”
  - The following float structures are valid: 0.2,1.,.3
  - Invalid characters: {~,@,#,$,[,],’,\}
- Switch case:
  - Only such format accepted
    switch(x)
    case 0: //one statement only
    Break; //mandatory
    case 1: //one statement only
    Break; //mandatory
    default: //one statement only
- For loop, while loop, do while: like c++
- Func definition
  - func returnType FuncatioName(args){ // code}
- Function call: like c++;
- dataTypes: int, float, bool, string and void (only used with functions return type)
- If else and if like c++, but else if not supported
- g++ is used instead of gcc for compilation
- The OS is linux ubuntu

The language is mainly like c++ with some modifications stated below
Examples:
//function definition keyword func needs to be used - function //needs to be implemented when defined
Func int mul(int x, int y){ return x*y; }
//function call
int x = mul(x,y);

Switch case
int x = 0;
int y = 0;
switch(x)
case 0: y = 1;
break;
case 2: y = 3;
break;
default: y=3;

There is If() {} else {}
And if() {}
But if() {} else if() {} not supported

While - do while - for - variable declaration all like c++
Constants declared like variables with const keyword as a prefix before data type
---constconstconst
Arrays not supported

Unlike c++: & -> and, | -> or

Program	Empty | `statments`
`statments`	statement
	Statements statment
	Block_statment
	Statements block_statments

Block_statment	'{' '}'
	'{' statments '}'
statment	';'
	while_statment
	if_statment

	for_statment
	Do_while_statment ';'
	switch_statment
	Func_defintion_statment
	Var_declare_statment ';'

	expression_statment ';'
while_statment	WHILE '(' expression_statment ')' block_statment
if_statment	matched_if
	unmatched_if
matched_if	IF '(' expression_statment ')' '{' statments '}' ELSE '{' matched_if '}' %prec ifpred

	statments
	Empty
unmatched_if	IF '(' expression_statment ')' '{' statments '}' IF '(' expression_statment ')' '{' matched_if '}' ELSE '{' unmatched_if '}'
for_statment	FOR '(' for_begining ';' expression_statment ';' expression_statment ')' block_statment

for_begining	expression_statment
	var_declare_statment
do_while_statment	DO block_statment WHILE '(' expression_statment ')'
switch_statment	SWITCH '(' Identifiers ')' cases_statment
case_statment	CASE intType ':' statment BREAK ';'

cases_statment	DEFAULT
	case_statment cases_statment
func_defintion_statment	FUNC VOID Identifiers '(' func_def_arguments ')' block_statment
	FUNC data_type Identifiers '(' func_def_arguments ')' func_return_statments
return_statment func_return_statments	RETURN expression_statment ';'

	'{' return_statment '}'
	'{' statments return_statment '}'
data_type	INT FLOAT {;} BOOL {;} STRING {;}
func_def_arguments:	data_type Identifiers
	func_def_arguments ',' data_type Identifiers
func_def_arguments	Empty

	data_type Identifiers
	func_def_arguments ',' data_type Identifiers
arguments	expression_statment
	arguments ',' expression_statment
	empty

func_call_statment	Identifiers '(' arguments ')'
func_call_statment	Identifiers '(' arguments ')'
var_declare_statment	data_type Identifiers
	data_type Identifiers '=' expression_statment
	Constant data_type Identifiers '=' expression_statment

value	intType
	floatType
	boolType
	stringType
expression_statment	'(' expression_statment ')'

	Identifiers
	value
	func_call_statment
	expression_statment_lv0
expression_statment_lv0	expression_statment '=' expression_statment

	expression_statment '+' expression_statment
	expression_statment '-' expression_statment
	expression_statment '*' expression_statment
	expression_statment '/' expression_statment
	expression_statment '%' expression_statment

	expression_statment '^' expression_statment
	expression_statment AND expression_statment
	expression_statment OR expression_statment
	expression_statment GREATER_THAN expression_statment
	expression_statment GREATER_EQUAL expression_statment

	expression_statment LESS_THAN expression_statment
	expression_statment LESS_EQUAL expression_statment
	expression_statment EQUAL expression_statment
	expression_statment NOT_EQUAL expression_statment
	NOT expression_statment

Token	Description
INT FLOAT BOOL STRING VOID	Data Types
IF ELSE WHILE DO FOR SWITCH CASE DEFAULT BREAK FUNC RETURN	Keywords
([0-9]+\.[0-9]+)|(\.[0-9]+)|([0-9]+\.)	Float Regex
[0-9]+	Integer Regex
[a-zA-Z_][0-9a-zA-Z_]*	Identifier Regex
"true", "false"	Boolean
(\"\")|(\"("\n"|"\t"|"\r"|.*)\")|(\"\'("\n"|"\t"|"\r"|.*)\'\")	String Regex
"=","+", "-", "*", "/", "%", "&", "|", "^",">", ">=", "<", "<=", "==", "!=", "!",	Operators