National University of Computer and Emerging Sciences, Lahore Campus

	Course Name:	Compiler Construction	Course Code:	CS-402
	Program:	BS (CS)	Semester:	Spring 2018
	Duration:	150 min	Total Marks:	50
	Paper Date:	25-May-2018	Weight	
	Section:	ALL	Page(s):	2
	Exam Type:	Final		

Student : Name:________________________________ Roll No._________________ Section:________

Question 1 (5+5 marks) [Solve on page 1 and 2]

Consider the following two examples of a figure element defined in a script:

```
\begin{figure}
     \caption{animals}
     \include{\pic\lion.png}
     \include{\pic\cow.bmp}
\end{figure}
```

```
\begin{figure}
     \caption{plants}
     \include{\root\images\rose.bmp}
     \include{\root\images\sunFlower.png}
     \include{\root\images\palm.bmp}
\end{figure}
```

As you can see, each figure element has a starting and ending tag; it has a caption (or title), and can have multiple images from various files.

Now answer the following questions:

a) List down all the tokens. Give regular definition for any complex token (having more than one lexeme).

b) Give a CFG for the figure element.

Question 2 (5+10 marks) [Solve on page 3 and 4]

Suppose a high-level programming language has a three-way selection statement "cmp". The cmp compares two given operands, and executes one out of the three statements/blocks. Following is an example:

```
cmp (x, y) {
     EQ: print(“x == y”)
     LT: print(“x < y”)
     GT: print(“x > y”)
}
```

a) Give three-address code for the above example. Note that only one out of the three blocks is executed.

b) Give a translation scheme to generate three-address code for this statement. Use the following grammar for your translator. You do not need to give semantic actions for the assignment statement etc.

```
S -> cmp ( id , id ) { EQ : L  LT : L  GT : L }
L -> L ; S | S
```

Note that the symbol "L" may generate multiple statements.

Question 3 (5+10 marks) [solve on page 5 and 6]

Like three-address code, syntax trees is a form of intermediate representation. The following translation scheme generates syntax tree for a given assignment statement:

```
S -> id = E      {tmp = Leaf(id.lex);
                  S.v = Node('=', tmp, E.v)}
E -> E1 + T      {E.v = Node('+', E1.v, T.v)}
E -> T           {E.v = T.v}
T -> T1 * F      {T.v = Node('*', T1.v, F.v)}
T -> F           {T.v = F.v}
F -> id          {F.v = Leaf(id.lex)}
```

a) Give output of the above translation scheme for the following statement:

```
x = a + b * c
```

Draw the syntax tree; not the parse tree!

b) Now remove left recursion from the above translation scheme. Rewrite the actions as well.

Question 4 (10 marks) [Solve on page 7]

Consider a virtual machine that executes three-address code. All variables are global and are stored in a data section. The only data type available is Integer. Following is its code skeleton:

```
int *ds = new int[..]; // data section
int quad[..][4]; // three-address code stored in quadruple
int pc = 0; // program counter
...
for (int pc = 0; quad[pc][0] != HALT; ++pc) {
     switch (quad[pc][0]) {
          case '+': ...
          case '-: ...
          case MIN: // Add code here!
          case SKIP: // Add code here!
          ...
     }
}
```

The machine supports several instructions. Give C/C++ code for the following two instructions:

```
MIN  X, Y, Z
SKIP X
```

The first instruction compares the contents of x and y, and stores the minimum value into z. While the second instruction jumps forwards skipping X number of instructions. For example, skip 1 means that the instruction after the skip-instruction will be skipped; similarly skip 2 means that the two instructions after the skip-instruction will be skipped.