Daffodil International University

Department of Computer Science and Engineering

Faculty of Science and Information Technology

Mid Term Examination, Semester: Spring 2017

Course Code: CSE331 **Course Title: Compiler Design** **Time: 90 minutes**

Course Teacher: All **Sections: All** **Campus: All** **Total Marks: 25**

Answer any three including question 1

1. Analyze the following code to answer the questions:

```
01    #include <stdio.h>
02    int main {
03    int n[10]; i, j;
04    for ( i = 0; i < 10; i++;){
05    n[i] = i + 100; }
06    for (j = 0; j < 10; j++;){
07    printf("Element[%d] = %d %d\n", j, n[j] ) }
08    return 0;}
09    }
```

a) Compilation of the above code produces lexemes at its first step. Show the all possible set of lexemes you may get. **2**

b) Few compiler errors may produce from the block of code. Identify, which lines have lexical and/or syntax errors (you need not to correct any code). **4**

c) Form the accurate token from the lexemes you made assuming that, the user has corrected all compiler errors. **2**

2. a) Explain with example, how a grammar becomes ambiguous and how to remove the ambiguity? **3**

b) Suppose you have a string like: **a + (b*a) - b + (a)**

Now- answer the following questions:

i) What grammar would be necessary to produce the above string? **3.5**

ii) How the parse tree would form to produce the same string. **2**

3. Suppose a Language $L = \{ P, Q, R \}$

a) Now, produce the Regular Expression: that is a string, starting with either ***P*** or ***R*** with at least one possibility of execution which is followed by an occurrence of ***Q*** with a confirm possibility and another ***Q*** with at least no possibility of occurrence. **3**

b) Produce the NFA from the above Regular Expression. **2**

c) Convert the produced NFA to DFA. **3.5**

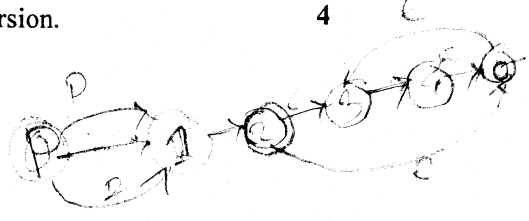
4. Consider the production E → E+T|T and answer the following:

a) The above production is left recursive or not? **1**

b) Why we eliminate left recursion? **1.5**

c) Is there any rule for elimination of left recursion? **2**

d) Draw the parse tree for before and after the elimination of left recursion. **4**



E → E+T/T

A → Aα/β

-- *Good luck* ☺ --