**DHARMSINH DESAI UNIVERSITY, NADIAD**
**FACULTY OF TECHNOLOGY**
**B.TECH. SEMESTER VI [INFORMATION TECHNOLOGY]**
**SUBJECT: (IT 608) LANGUAGE TRANSLATOR**

| | | | |
|---|---|---|---|
| **Examination** | **: First Sessional** | **Seat No.** | **:** |
| **Date** | **: 18/01/2016** | **Day** | **: Monday** |
| **Time** | **: 12.30 to 1.45** | **Max. Marks** | **: 36** |

**INSTRUCTIONS:**
1. Figures to the right indicate maximum marks for that question.
2. The symbols used carry their usual meanings.
3. Assume suitable data, if required & mention them clearly.
4. Draw neat sketches wherever necessary.
5. Here | is rule separator and ^ stands for NULL/ Empty string.

**Q.1 Do as directed.**

(a) Consider a program P that consists of two source modules M1 and M2 contained in two different files. If M1 contains a reference to a function defined in M2 the reference will be resolved at **[1]**
(A) Run time (B) Compile time (C) Link time (D) Load time

(b) Which of the following suffice to convert an arbitrary CFG to LL (1) Grammar? **[1]**
(A) Removing left recursion
(B) Factoring the grammar
(C) Removing left recursion and Factoring the grammar
(D) None of these

(c) How many steps are required to parse the string 'abdbdcde' using following grammar? (Use Brute Force Approach) **[1]**
S→ aABe A→ bdA |c| de B→ d
(A) 5 (B) 6 (C) 9 (D) 11

(d) The context free grammar is ambiguous if **[1]**
(A)It produces more than one leftmost derivation for the same sentence.
(B)It produces more than one rightmost derivation for the same sentence.
(C)It produces more than one parse tree for the same sentence.
(D)All of the above.

(e) Is the given grammar valid to be LL (1)? If Not, identify the problem and resolve it. **[2]**
**Grammar G1**: S → T + S | T, T→U*T | U, U→(S) | V, V→0|1|2|..|9

(f) Which are the phases of compiler? Explain each phase in brief. **[2]**

(g) What is back-patching? When it is needed? **[2]**

(h) Explain in detail with example: Token, Pattern and Lexemes. **[2]**

**Q.2** Attempt ***Any Two*** from the following questions. **[12]**

(a) For the following 'C' fragment, identify and list the lexemes that make up tokens. [6]

```
int MyX(int *E, unsigned int size)
{
int Y = 0, Z, i, j, k;
for(i = 0; i < size; i++)
   for(j = i; j < size; j++)
   {
      Z = 0;
      for(k = i; k <= j; k++)
      Z = Z + E[k];
      if (Z > Y)
        Y = Z;
   }
return Y;
}
```

(b) Find out FIRST and FOLLOW set for all the Nonterminals for following **grammar G2** : S→ABCDE A→a | ^ B→b | ^ C→c D→d | ^ E→e | ^ [6]

(c) Write RDP for the following **grammar G3** : S→dA | aB A→ bA | c B→ bB | c [6]

**Q.3** (a) Construct the non recursive predictive parser for the following **grammar G4**: **[8]**
**S→ ACB | CbB | Ba A→da | BC B→g|^ C→h|^.** Trace the behavior of parser on the input "ghhg" using the grammar. Note: - if the parser cannot work directly on the given grammar, you can replace it with equivalent grammar.

(b) Write LEX program for the **Language L1.** **[4]**
**Whitespaces**- blanks, tabs, newlines. **Keywords**- if, else, then, end, start
**User identifiers**- start with capital letters, followed by one or more letters.
**Comments**- start with **<! - -** Anything that is comment **- - >**
**Character constants**- anything between ` `.

**OR**

**Q.3** (a) Consider the **grammar G5:**S→**iEtSS' | a S'→eS|^ E→b**. Construct a predictive parsing table for the grammar given above. Justify the grammar is LL(1) or Not. **[6]**

(b) Draw transition diagram for the rules given in **Language L1 Q.3 (b) above**. **[6]**