Reg.No. \_\_\_\_\_\_\_\_\_\_\_\_



**UNIVERSITY**

(Karunya Institute of Technology & Sciences)

(Declared as Deemed-to-be University under Sec.3 of the UGC Act, 1956)

**Supplementary Examination – June – 2017**

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
| **Code :** | **14CS2038** | **Duration :** | **3hrs** |
| **Sub. Name :** | **PRINCIPLES OF COMPILER DESIGN** | **Max. marks :** | **100** |

**ANSWER ALL QUESTIONS (5 x 20 = 100 Marks)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Q. No. | Sub Div. | Questions | Course  Outcome | Marks |
| 1. |  | Elaborate on the various phases of compiler. Show the translation for an assignment statement “Interest = Principal \* 10 / 100”. Clearly indicate the output of each phase. | CO1 | 20 |
| (OR) | | | | |
| 2. | a. | Discuss the requirement of buffering technique while recognizing tokens? Summarize the use of sentinels in this process. | CO1 | 10 |
| b. | Write a regular definition for the following specifications:   1. Floating point numbers in C language. 2. Identifiers of C language 3. All string of lower case letters in which the letters are in ascending lexicographic order | CO2 | 10 |
| 3. |  | Convert the following regular expression to DFA by constructing a syntax tree and computing NULLABLE(), firstpos(), lastpos() and followpos() of each node: b (ab+aa)\* a | CO1 | 20 |
| (OR) | | | | |
| 4. | a. | Write a LEX program for token separation.(Tokens to be identified are: keywords, operators, delimiters, identifiers, constants) | CO3 | 10 |
|  | b. | Draw the transition diagram to recognize the tokens given below:   1. Relational operator 2. Identifiers 3. Floating point numbers | CO1 | 10 |
| 5. |  | Construct the predictive parsing (LL(1)) table by making necessary changes to the grammar given below:  E 🡪E + T | T  T 🡪T \* F | F  F 🡪 (E) | id  Parse the string “id + id \* id” using the parsing table. | CO3 | 20 |
| (OR) | | | | |
| 6. |  | Obtain LR(0) items for the following grammar.  S 🡪 L = R | R  L 🡪 \* R | id  R 🡪 L  Justify whether the above grammar is SLR or not. | CO3 | 20 |
| 7. |  | Apply CLR parsing algorithm to construct the parsing table for the following grammar.  S 🡪 id | V = E  V 🡪 id  E 🡪 V | n  Parse the string “id = n” using the parsing table. | CO3 | 20 |
| (OR) | | | | |
| 8. | a. | Develop the following intermediate representations for the expression z = (-x / y) + (-x / y)   1. Postfix notation 2. Syntax tree 3. DAG 4. Three address code 5. Quadruples 6. Triples 7. Indirect Triples | CO2 | 10 |
|  | b. | Write syntax directed definition for simple type declaration and construct dependency graph for the declaration, int id1, id2. | CO2 | 10 |
|  | | **Compulsory:** |  |  |
| 9. | a. | Examine the general structure of activation record. | CO2 | 10 |
|  | b. | Illustrate peephole optimization with appropriate example. | CO1 | 10 |

ALL THE BEST