This question paper contains **3** printed pages]

ND—05—2023

FACULTY OF SCIENCE AND TECHNOLOGY

B.Sc. (Second Year) (Fourth Semester) EXAMINATION

NOVEMBER/DECEMBER, 2023

(CBCS/Revised Pattern)

COMPUTER SCIENCE

(Principle of Compiler Design)

(Thursday, 30-11-2023) **Time : 2.00 p.m. to 5.00 p.m.**

Time—Three Hours *Maximum Marks*—75

N.B. :— (*i*) *All* questions are compulsory.

(*ii*) Figures to the right indicate full marks.

(*iii*) Assume suitable data, if required.

(*iv*) Use of any electronic media such as mobile phone, digital diary and electronic calculator is not permitted.

1. Attempt any *five* of the following (**3** marks each) : 15

(*a*) Explain need of translator in detail.

(*b*) Define programming languages.

(*c*) Explain regular expression.

P.T.O.

(*d*) Explain errors in compiler designing.

(*e*) Explain data elements.

(*f*) Explain semantic errors.

(*g*) Explain context free grammar.

2. Attempt any *three* of the following (**5** marks each) : 15

(*a*) Explain lexical and syntactic structure of language.

(*b*) Describe different data structures used in compiler designing.

(*c*) Explain capabilities of context free grammar.

(*d*) Explain minimization of number of states of DFA.

(*e*) Explain operator precedence parsing.

3. Attempt any *three* of the following (**5** marks each) : 15

(*a*) Explain implementation of syntax directed translator.

(*b*) Describe sources of optimization.

(*c*) Explain phases of compiler.

(*d*) Explain one pass and multi pass compiler.

(*e*) Explain predictive parsers.

4. Attempt any *three* of the following (**5** marks each) : 15

(*a*) Explain lexical and syntactic structure of language.

(*b*) Explain predictive parsers and LR parsers.

(*c*) Explain evaluation of postfix notation.

(*d*) Explain conversion of regular expression to finite automata.

(*e*) Explain role of lexical analyzer and input buffering.

5. Write short notes on any *three* of the following (**5** marks each) : 15

(*a*) Bootstrapping

(*b*) Parse tree and syntax tree

(*c*) Finite automata

(*d*) Operator precedence parsing

(*e*) Syntactic phase errors.