Total No. of Questions : 10] **SEAT No. :** □

P3221 **[Total No. of Pages : 3**

[5461]-262

B.E. (Computer Engg.)

PRINCIPLES OF MODERN COMPILER DESIGN

(2012 Pattern) (Semester - I) (410442)

Time :2 ½ Hours] ***[Max. Marks : 70***

Instructions to the candidates :

1) ***Answer Q.No.1 or Q.2, Q.3 or Q.4, Q.5 or Q.6, Q.7 or Q.8 and Q.9 or Q.10.***
2) ***Neat diagrams must be drawn wherever necessary.***
3) ***Black figures to the right indicate full marks.***
4) ***Assume Suitable data if necessary.***

Q1) a) Why compilation phases are divided into front - end and back - end? What are the advantages? **[4]**

b) Give syntax directed definition for constructing syntax tree for arithmetic expression. **[6]**

OR

Q2) a) Explain the meaning of following symbols used in LEX. **[4]**

i) / ii) $

iii) ^ iv) {}

v) [] vi) .

vii) | viii) \

b) Write the syntax directed translation scheme of generating intermediate code for assignment statement. **[6]**

Q3) a) Define the phase and pass related to compiler. **[2]**

b) Check whether the following grammar LL(1) or not.

E TE
E *TE /
T FT
T ^T/
F (E)/id **[8]**

OR

Q4) a) Explain the goto function used in LR parser design. **[2]**

b) Show that the following grammar is not SLR (1) **[8]**

S Aa Ab | B b Ba

A

B

Q5) a) Explain sources of code optimization. **[6]**

b) Show the steps involved on generating the code for the expression. **[6]**

k=(a+b)* c+d/ (a+b)+b

(Assuming there are only 2 registers available)

c) Explain the method for constructing DAG. Construct DAG for following code D = B*C

E = A + B

B = B* C

A = E-D **[6]**

OR

Q6) a) Discuss following optimizations with example **[6]**

i) Constant folding (compile time evaluation)

ii) Variable propogation

b) Discuss various issues in code generation phase. **[6]**

c) Explain the algorithm for simple code generation. **[6]**

Q7) a) Explain following related to Haskell program. **[6]**

i) Offside rule

ii) List

b) Explain following features of Object oriented languages related to compiler design.

i) Overloading

ii) Inheritance **[6]**

c) Explain how code is generated for control flow statements. **[4]**

OR

Q8) a) Discuss following with respect to Object oriented languages. **[6]**

i) Type checking

ii) Type coercion

b) Explain following with respect to Functional languages. **[6]**

i) Polymorphic typing

ii) Lazy evaluation

c) Discuss the structure of Java CC. **[4]**

Q9) a) Discuss following with respect to Parallel object oriented languages. **[6]**

i) Object location

ii) Object migration

b) Write short notes on. **[6]**

i) Dynamic compilation

ii) GCC

c) Explain following shared variable models. **[4]**

i) Locks

ii) Monitors

OR

Q10) a) Write short notes on. **[6]**

i) JIT

ii) nmake

b) Discuss the issues in Tuple Space implementation **[6]**

c) Explain cross compilation using XMLVM. **[4]**

❍❍❍❍