Birla Institute of Technology & Science, Pilani
Work-Integrated Learning Programmes Division
Second Semester 2015-2016

Comprehensive Examination
(EC-3 Regular)

Course No.	: SS ZG514
Course Title	: OBJECT ORIENTED ANALYSIS & DESIGN
Nature of Exam	: Open Book
Weightage	: 50%
Duration	: 3 Hours
Date of Exam	: 10/04/2016 (FN)

No. of Pages = 3
No. of Questions = 5

Note:

1. Please follow all the *Instructions to Candidates* given on the cover page of the answer book.
2. All parts of a question should be answered consecutively. Each answer should start from a fresh page.
3. Assumptions made if any, should be stated clearly at the beginning of your answer.

Q.1. SOLID is an acronym of acronyms.

(a) Write the full form of each of the 5 acronyms which collect to form SOLID. [2.5]

(b) Explain each in one line. [2.5]

(c) Give an example to explain each. [5.0]

Q.2. Described below are 5 creational patterns.

(a) Identify each. [2.5]

(b) Explain your choice. [2.5]

(c) Explain the key feature of the mechanism that achieves the requirements of GRASP under the heading "creator" in each of the patterns. [5.0]

Description of 5 Creational Patterns

I. It define an interface for creating an object, but let subclasses decide which class to instantiate. A Method lets a class defer instantiation to subclasses (dependency injection).

II. It provide an interface for creating families of related or dependent objects without specifying their concrete classes.

III. Ensure a class has only one instance, and provide a global point of access to it.

IV. Specify the kinds of objects to create using a prototypical instance, and create new objects by copying this prototype.

V. Separate the construction of a complex object from its representation allowing the same construction process to create various representations.

Q.3. Draw the class diagram for the code given below:

(a) Identify the Classes. [2.5]

(b) Identify tthe attributes and methods. [2.5]

(c) Draw the diagram clearly showing the associations between classes. [5.0]

```
/**
 * @opt operations
 * @opt attributes
 * @opt types
 * @hidden
 */
class UMLOptions {}
/* Define some types we use */
/** @hidden */
class Name {}
/** @hidden */
class Number {}
/**
 * @has 1..* Member * Student
 * @composed 1..* Has 1..* Department
 */
class School {
      Name name;
      String address;
      Number phone;
      void addStudent() {}
      void removeStudent() {}
      void getStudent() {}
      void getAllStudents() {}
      void addDepartment() {}
      void removeDepartment() {}
      void getDepartment() {}
      void getAllDepartments() {}
}
/**
 * @has 1..* AssignedTo 1..* Instructor
 * @assoc 1..* - 1..* Course
 * @assoc 0..* - "0..1 chairperson" Instructor
 */
class Department {
      Name name;
      void addInstructor() {}
      void removeInstructor() {}
      void getInstructor() {}
      void getAllInstructors() {}
}
/**
 * @assoc * Attends * Course
 */
class Student {
      Name name;
      Number studentID;
}

class Course {
      Name name;
      Number courseID;
}
/**
 * @assoc 1..* Teaches * Course
 */
class Instructor {
      Name name;
}
```

Q.4. You have a Square class that derives from a Rectangle class. Assuming getter and setter methods exist for both width and height. The Square class always assumes that the width is equal to the height. If a Square object is used in a context where a Rectangle is expected, unexpected behavior may occur because the dimensions of a Square cannot (or rather should not) be modified independently.

(a) Comment on the application of LSP in this case. [2.5]

(b) Comment on the above if there were no setter methods. Parameters can only be passed by constructors [2.5]

(c) What are the constraints specified in LSP with respect to precondition, postcondition and invariants. Explain with an example. [5.0]

Q.5. The Gang of four patters support the principles indicated in GRASP.

(a) In the box given below mark a tick in the squares to show a relation between the 5 patterns in the x-axis with 6 patterns in the y-axis. [2.5]

(b) Explain the problem addressed by each of the 5 patterns to show how the 6 patterns on the y-axes help to address these problems. [5 x 1.5 = 7.5]

	Adapter	Factory	singleton	composite	Strategy
Low coupling					
High cohesion					
Polymorphism					
Pure fabrication					
Indirection					
Protected Variation					

****