MODULE 9

OBJECT-ORIENTED SYSTEM MODELLING

OBJECTIVE QUESTIONS

There are 4 alternative answers to each question. One of them is correct. Pick the correct answer. Do not guess. A key is given at the end of the module for you to verify your answer

LEARNING UNIT 1

9.1.1 Computer systems are designed by

(a) simplifying requirements of system
(b) breaking of the system into smaller self-contained co-operating subsystems
(c) breaking up the systems into independent parts
(d) modular design

9.1.2 Functions and procedures are

(a) not useful in designing computer systems
(b) old fashioned and they are not useful
(c) useful in designing computer systems
(d) have side effects which require special care if they are used as subsystems

9.1.3 A subsystem of a complex system must specify

(a) what task it performs
(b) how it performs a task
(c) with which subsystems it co-operates
(d) how it co-operates with other systems

9.1.4 A subsystem of a complex system must

(i) know how other subsystems perform their task
(ii) know what tasks other subsystems perform
(iii) know what task it performs and other subsystems perform to access its data
(iv) know how to send requests to other systems for getting tasks done by them

(a) i, ii (b) ii, iii
(c) ii, iv (d) iii, iv

9.1.5 A subsystem of a complex system

(i) should be reusable in other complex system
(ii) must not be able to inherit the properties of other subsystems
(iii) must have clearly specified responsibilities
(iv) must know the stimuli to which it should respond

(a) i, ii, iii (b) ii, iii, iv
(c) i, iii, iv (d) i, ii, iv

9.1.6 By polymorphism of a subsystem we mean

(a) it should be reusable
(b) it should have polymorphic data types
(c) it should accept generic commands and interpret appropriately
(d) it should morph polygons

9.1.7 The advantages of object-oriented modelling are

(i) it allows easy integration of subsystems
(ii) it promotes reuse of code written earlier
(iii) it allows modification of some objects by other objects
(iv) it allows data structures in objects to be modified by other objects

(a) i, ii (b) i, iii
(c) ii, iii (d) i, iv

9.1.8 Objects are

(i) tangible entities
(ii) intangible entities
(iii) transient entities
(iv) uniquely identifiable

(a) i, ii (b) i, ii, iii
(c) i, ii, iii, iv (d) i, ii, iv

9.1.9 A class is

(a) a group of objects
(b) template for objects of a particular type
(c) a class of objects
(d) a classification of objects

9.1.10 All objects have

(i) attributes
(ii) states
(iii) a set of operations
(iv) a unique identity

(a) i, ii, iii (b) ii, iii, iv
(c) i, iii, iv (d) i, ii, iii, iv

9.1.11 In UML diagram of a class

(a) state of object cannot be represented
(b) state is irrelevant
(c) state is represented as an attribute
(d) state is represented as a result of an operation

9.1.12 Attributes are assigned value

(a) when operations are performed on an object
(b) when instances of objects are defined
(c) when methods are invoked
(d) when classes are identified

9.1.13 The following are intangible entities which can be defined as objects

(i) a motor car
(ii) a bank account
(iii) an aircraft
(iv) a linked list

(a) i, ii (b) ii, iv
(c) iii, iv (d) ii, iii, iv

9.1.14 A query operation on a object

(a) has side effect
(b) has no side effects
(c) changes the state of an object
(d) is not allowed

9.1.15 An instance of an object is created by a

(a) query operation
(b) update operation
(c) constructor operation
(d) open operation

9.1.16 An update operation in an object instance

(a) updates the class
(b) has no side effects
(c) deletes an instance
(d) alters values of attribute(s) of an object instance

9.1.17 In object-oriented design

(a) operations and methods are identical
(b) methods specify algorithms whereas operations only state what is to be done
(c) methods do not change values of attributes
(d) methods and constructor are same

9.1.18 By abstraction in object-oriented modelling we mean picking

(a) only attributes appropriate to model an object
(b) only operations
(c) both operation and attributes with operations appropriate to model an object
(d) the appropriate abstract data type

9.1.19 By encapsulation in object-oriented modelling we mean

(a) encapsulating data and programs
(b) hiding attributes of an object from users
(c) hiding operations on object from users
(d) hiding implementation details of methods from users of objects

9.1.20 Encapsulation in object-oriented modelling is useful as

(a) it allows improving methods of an object independent of other parts of system
(b) it hides implementation details of methods
(c) it allows easy designing
(d) encapsulates attributes and operations of object

9.1.21 Objects may be viewed as

(a) clients in a system
(b) servers in a system
(c) as both clients and servers in a system
(d) neither as clients nor as servers in a system

9.1.22 Inheritance in object-oriented system is used to

(a) create new classes from existing classes
(b) add new operations to existing operations
(c) add new attributes to existing attributes
(d) add new states to existing states

9.1.23 Inheritance in object-oriented modelling can be used to

(a) generalize classes
(b) specialize classes
(c) generalize and specialize classes
(d) create new classes

9.1.24 When a subclass is created using inheritance the resulting class

(a) may have only attributes of parent class
(b) may have only operations of parent class
(c) may have new operations only in addition to those in parent class
(d) may have new attributes and new operations in addition to those of the parent class

9.1.25 By polymorphism in object-oriented modelling we mean

(a) the ability to manipulate objects of different distinct classes
(b) the ability to manipulate objects of different distinct classes knowing only their common properties
(c) use of polymorphic operations
(d) use of similar operations to do similar things

9.1.26 A polymorphic operation

(a) has same name
(b) has same name but uses different methods depending on class
(c) uses different methods to perform on the same class
(d) uses polymorphic method

LEARNING UNIT 2

9.2.1 Given a word statement of a problem potential objects are identified by selecting

(a) verb phrases in the statement
(b) noun phrases in the statement
(c) adjectives in the statement
(d) adverbs in the statement

9.2.2 Given a word statement of problem potential operations appropriate for objects are identified by selecting

(a) verb phrases in the statement
(b) noun phrases in the statement
(c) adjectives in the statement
(d) adverbs in the statement

9.2.3 Objects selected to model a system

(i) must be essential for functioning of the system
(ii) must have all attributes which are invariant during operations of a system
(iii) must have attributes relevant for performing services of object
(iv) must be able to perform assigned services

(a) i, ii, iii (b) ii, iii, iv
(c) i, iii, iv (d) i, ii, iii, iv

9.2.4 An object is selected for modelling a system provided

(a) its attributes are invariant during operation of the system
(b) its attributes change during operation of the system
(c) it has numerous attributes
(d) it has no attributes relevant to the system

9.2.5 An object is considered an external entity in object-oriented modelling if

(a) its attributes are invariant during operation of the system
(b) its attributes change during operation of the system
(c) it has numerous attributes
(d) it has no attributes relevant to the system

LEARNING UNIT 3

9.3.1 Object-oriented system modelling using CRC method gives

(a) Java programs for the system
(b) C++ programs for the system
(c) Classes of the system, their responsibilities and collaborating classes
(d) Objective C programs for the system

9.3.2 The expansion of the acronym CRC is

(a) Collecting Responsibilities Classes
(b) Collaborating with Relevant Classes
(c) Class Responsibilities and Collaborators
(d) Creating Relevant Classes

9.3.3 In CRC based design a CRC team consists of

(i) one or two user's representatives
(ii) several programmers
(iii) project coordinators
(iv) one or two system analysts

(a) i, ii (b) i, iii
(c) i, iii, iv (d) i, ii, iii, iv

9.3.4 A class index card contains besides class name

(i) superclasses and subclasses
(ii) short description of class
(iii) collaborators
(iv) private responsibilities of class
(v) contract(s) with collaborators

(a) i, ii, iii (b) i, iii, iv, v
(c) i, ii, iii, iv (d) i, ii, iii, iv and v

9.3.5 The CRC modelling primarily requires

(i) identifying classes and their responsibilities

(ii) identifying collaborators of each class and their responsibilities

(iii) developing a collaboration graph

(a) i, ii (b) i, iii

(c) ii, iii (d) i, ii, iii

KEY TO OBJECTIVE QUESTIONS

9.1.1 b	**9.1.2 d**	**9.1.3 c**	**9.1.4 c**	**9.1.5 c**	**9.1.6 c**
9.1.7 a	**9.1.8 d**	**9.1.9 b**	**9.1.10 d**	**9.1.11 c**	**9.1.12 b**
9.1.13 b	**9.1.14 b**	**9.1.15 c**	**9.1.16 d**	**9.1.17 b**	**9.1.18 c**
9.1.19 d	**9.1.20 a**	**9.1.21 c**	**9.1.22 a**	**9.1.23 c**	**9.1.24 d**
9.1.25 b	**9.1.26 b**	**9.2.1 b**	**9.2.2 a**	**9.2.3 c**	**9.2.4 b**
9.2.5 a	**9.3.1 c**	**9.3.2 c**	**9.3.3 c**	**9.3.4 d**	**9.3.5 d**