B.TECH.

SAMPLE PAPER

As per New Exam. Pattern w.e.f. 2009 - 10

OBJECT ORIENTED SYSTEMS

Time : 3 Hours *Total Marks : 100*

Note. Attempt ALL questions.

SECTION - A

(1×20=20)

Q.1. Attempt all the parts of this questions. All parts of the question carry equal marks. This question contains 20 objectives/fill in the blanks type questions.

(*i*) **Visualizing program components as objects is characteristics of which of the following language types?**

(a) Object oriented programming language
(b) Machine language
(c) Command line operating system
(d) Procedural language

Ans. (a) Object oriented programming language

(*ii*) **What will be the result of compilation for the following code?**

```
Public class MyClass
{
    final int i;
    public static void
main(String[ ] arguments)
    {

system.out.println(new
MyClass( ).i);
    }
}
```

(a) Will print 0
(b) Will give compile error
(c) Will give runtime error
(d) Will give syntax error

Ans. (b) Will give compile error

(*iii*) **A subclass is also called as**

(a) inner class
(b) nested class
(c) derived class
(d) hidden class

Ans. (c) derived class

(*iv*) **Command to execute a compiled java program is**

(a) run (b) execute
(c) javac (d) java

Ans. (d) java

(*v*) **Attribute of an Object is also known as its**

(a) State (b) Method
(c) Behavior (d) Procedures

Ans. (c) Behavior

(*vi*) **Which of the following method is not defined by the Applet class?**

(a) init( )
(b) paint( )
(c) start( )
(d) None of the above

Ans. (b) paint( )

(*vii*) **Java language has support for which of the following types of comments**

(a) block, line and javadoc
(b) javadoc, literal and string
(c) javadoc, char and string
(d) single, multiple and quote

Ans. (a) block, line and javadoc

(*viii*) **Which class is used for character stream in java . io?**

(a) FileOutputStream
(b) ByteArrayOutputStream
(c) FileWriter
(d) PrintStream

Ans. (c) FileWriter

(ix) **Which of the following object is a contained or transport mechanism to send/execute (normally) SQL statements and retrieve any results via its associated Connection?**

(a) DriverManager
(b) Statement
(c) DatabaseMetaData
(d) Driver

Ans. (b)

(x) **Swing components that don't rely on native GUI are referred to as**

(a) heavyweight components
(b) GUI components
(c) non-GUI components
(d) lightweight components

Ans. (d)

(xi) **Which of the following gives a technical definition of the language that includes the syntax and semantics of the Java programming language?**

(a) Java language specification
(b) Java API
(c) Java JDK
(d) Java IDE

Ans. (a) Java language specification

(xii) **A coloured image can be converted into a grayscale by using**

(a) CropImagerFilter
(b) RGBImageFilter
(c) ImageConsumer
(d) ImageProducer

Ans. (d) ImageProducer

(xiii) **If not assigned a value, a variable of type *char* has which default value?**

(a) '\u0001'　　(b) '\uffff'
(c) " " (space)　　(d) '\u0000'

Ans. (a) '\u0001'

(xiv) **When is an object eligible for garbage collection?**

(a) When an object becomes unreachable by any code
(b) When the "finalize" method is called on the object
(c) When an object goes out of scope
(d) When the system runs out of virtual memory

Ans. (a) When an object becomes unreachable by any code

(xv) **A class needs to be created that will store unique object elements. The elements need not be sorted but they must be unique. What interface might be most suitable to meet this need?**

(a) Set　　(b) List
(c) Map　　(d) Vector

Ans. (a) Set

(xvi) Class String provides ______ static method that takes an argument of any type and converts the argument to a String object.

Ans. valueOf

(xvii) The _____ flag causes output to be left justified in a field.

Ans. – (minus)

(xviii) A(n) ______ object is used to submit a query to a database.

Ans. Statement

(xix) The GridbagConraints's instance variable _____ is set to CENTER, by default for component.

Ans. anchor

(xx) The most common reason for using the ______ keyword is, because a field is shadowed by a method or constructor parameter.

Ans. This

SECTION - B

Note: Attempt any three parts of the following: (10×3=30)

Q.2.(*a*) What is object -Oriented approach? Explain.

Ans. 'Object Oriented' means the software components are organized as a collction of discrete objects to incoportes both data and possible actions over it. The four generally adopted aspects for Object-Oriented approach are: Identity, Classification, Polymorphism and Inheritance.

'Object Oriented Approach' is a different kind of expressive mdium. As a result of using OOAD, the tools starts to look less like machines and more like parts of our minds, and also like other forms of expression such as writing, painting, sculpture, animation, and filmmaking. Object-oriented programming (OOP) is part of this movement toward using he computer as an expressive medium.

Object-Oriented
= Objects + Communication
+ Classification + Inheritance

Following charactristics represent a pure approach to Object-oriented programming:

- *Everything is an object:* Assume object as a fancy variable; that stores data, but you can 'make requests' to the object, and can tell an Object to perform operations on itself.
- *A program is a bunch of objects telling each other what to do by sending message:*
- *Each object has its own memory mde up of other objects:* i.e., you can create a new generation of object by making a class containing existing objects. One can also say it like that you can build complexity in a program while hiding it behind the simplicity of objects.
- *Every object has a type:* Each object is an instance of a class, in which 'class' is synonymous with 'type'.
- *All objects of a particular type can receive the same type of messages.* This is actually a loaded statement, as you will see later. Because an object of type 'Square' is also an object of type 'Rctangle', a square is guaranteed to accept Rectangles messages. This means you can write code that talks to Rectangle and automatically handle anything that fits the description of a Rectangle. This substitutability is one of the most powerful concepts in OOP.

2. (*b*) What do you mean by conditional state transitions and state action?

Ans. Condition : A conditions is a Boolean function of object values. It is important to distinguish conditions from events, which have no time duration. A state can be defined in terms of condition, conversely, being in a state is a condition.

Condition can be used as guard on transitions. A guarded transition fires when its events occurs, only if the guard condition is true. For example, "When you go out in the morning (event) if the temperature is below freezing (condition), then put on your gloves (next state)." A guard condition on a transition is shown as a Boolean expression in bracket following event name. Following figure showing a state diagram with guarded transitions for traffic lights at an inter section. One pair of electric eyes checks the north-south left turn lanes, another pair checks the east-west turn lanes. If no car is in the north-south and/or east-west turn lanes then the traffic light control logic is smart enough to skip the left turn position of the cycle.

Operation:

- State diagrams would be of little use if they just described pattern of events.

- A behavioral description of an object must specify what the object does in response to events.
- An activity is an operation that takes time to complete.
- Activities include continuous operation.
- A state may control a continuous activity, such as ringing a telephone bell, that persist until an event terminates.
- The notation "do : A" with in state box indicate that activity.
- An action is an instantaneous operation. An action is associated with an event.
- An action represents and operation whose duration is insignificant compared to the resolution of the state diagram.
- i.e. disconnect phone line might be an action in response to an event an on-hook event.
- Action can also represent internal control operation such as setting attributes or generating other events.
- The notation for an action on a transition is a slash ('/') and the name (or description) of the action, following the name of the event that causes it.

2. (*c*) What is Jackson Structured Development methodologies to software engineering?

Ans. Jackson Structured Development: Jackson Structured Development (JSD) is another mature methodology which has a different style than SA/SD or OMT. The JSD methodology was developed by Michael Jackson and is especially popular in Europe. JSD does not distinguish between analysis and design and instead lumps analysis and design and instead lumps both phases together as specification. JSD divides system development into two stages; specification, then implementation. JSD first determines the "what" and then the "how". JSD is intended especially for applications in which timing is important.

A JSD model begins with consideration of the real world. The purpose of a system is to provide functionality, but Jackson feels that one must first consider how this functionality fits in with real world. A JSD model describes the real world in terms of entities, actions and ordering of actions. Entities usually appear as nouns in requirement statements and actions appear as verbs. JSD software development consist of six sequential steps: entity action step, entity structure step, initial model step, and implementation step.

Jackson presents several examples, one of which is the design of an elevator control system. The elevator control system controls two elevators which services six floors. Each elevator has six inside buttons (one for each floor). Each floor has Up and Down buttons in the waiting area. Jackson identifies two entities for elevator control example: button and elevator.

Entity structure step: Action occurs in the real world and is not an artifact of the system. Action takes place at a point in time, are atomic, and not decomposable. The entity structure step partially orders the actions of each entity by time. The elevator control system illustrates the importance of ordering actions. It is permissible for an elevator to arrive at floor 3, leaving floor 3, arrive at floor 2, leave floor 2 and so on. It does not make sense for two arrive actions to occur in succession; arrive and leave operations must alternate.

Initial Order: The initial model step states how the real world connects to the abstract model. JSD supports state-vector and data stream connection. The elevator control system illustrates state-vector connection. The elevator user dose not want the control system to remember each button pressed and send an elevator fine times to service request. The JSD model of the computer system is unaware of the number of presses and only communicates with the real world in the "up-flag".

Function step: It uses pseudocode to state output of actions. At the end of this step the developer has a complete specification of the required system in the elevator example, turning the display panel lights on or off as an elevator arrives at each floor is a function that must be specified.

System Timing step: This step considers how mush the model is permitted to lag the real world. For the most part, the result of the timing step is a set of informal notes on performance constraints. For example, an elevator control system must detect when up and down buttons are pressed.

Implementation Step: This focuses on the problems of process scheduling and allocates processors to processes. The number of processes may be different from the number of processors. Jackson's elevator control model has 50 processes. The developer must decide whether to match each process to one or 50CPU's or how to get several processes to share the same CPU. After the six JSD steps comes writing of code and database design.

2. (*d*) Write short nore on Double Buffering.

Ans. Images are not only useful for string pictures, but it can also be used as offscreen drawing surfaces. The allows programmers to render any image, including text and graphics, to an offscreen buffer that can be displayd at later time. The advantage to doing this is that the image is seen only when it is complete.

Drawing a complicated image could take several milliseconds or more, which results in fishing or flickering.This flashing is distracting. Use of an offscreen image to reduce flashing ro flictiering is calleddouble buffering, because the screen is considered a buffer for pixel, and the offscreen image is the second buffer, where programmer can prepare pixel for display.

2. (*e*) What is a Java servlet ? Discuss the life cycle of a servlet.

Ans. Servlets are small programs that execute on the server side of a web connection. Servlets dynamically extend the functionality of a web server servlets offer several advantages in comparison with CGI. First performance is significantly better. Second servlets are platform - independent because they are written in Java. Third, the Java security manager on the server enforces a set of restrictions to protect the resources on a server machine.

Life Cycle of Servlet : Three methods are central to the life cycle of a servlet. These are init ( ), service ( ), and destroy ( ).

The server invokes the init ( ) method of the servlet. This method is invoked only when the servlet is first loaded into memory.

The server invokes the service ( ) method of the servlet. This method is called to process the HTTP request. It may also formulate an HTTP response for the client. Service ( ) method is called for each HTTP request. The sever calls the destroy ( ) method to relinquish any resource such as file handles that are associated for the servlet.

SECTION - C

(10×5=50)

Note: Attempt all the questions. All questions carry equal marks.

Q.3. Attempt any one part of the following:

(*a*) Differentiate between generalization and aggregation.

Ans. Aggregation Versus Generalization: Aggregation is not the same thing as generalization. Aggregation relates instances. Two distinct objects are involved: one of them is a part of the other. Generalization relates classe and is a way of structuring the description of a single object. Both superclass and subclass refer to properties of a single objct. With generalization, an object is simultaneously an instance of the superclass and instance of the subclass. Confusion arises because both aggregation and generalization give rise to trees through transitive closure. An aggregation tree is composed of object instances that are all part of a composite object, a generalization tree is composed of classes that describe an object. Aggregation is often called "a-part-of" relationship; generalization is often called "a-kind-of" or "is-a" relationship.

Figure illustrates aggregation and generalization for the case of a desk lamp. Parts explosions are the most compelling examples of aggregation. Base, cover, switch, and wiring are all part of a lamp. Lamps may be classified into seveal different subclasses: flourescent and incandescent, for example. Each subclass may have its own distinct parts. For example, a fluorescent lamp has a ballast, twist mount, and starter, an incandescent lamp has a socket.

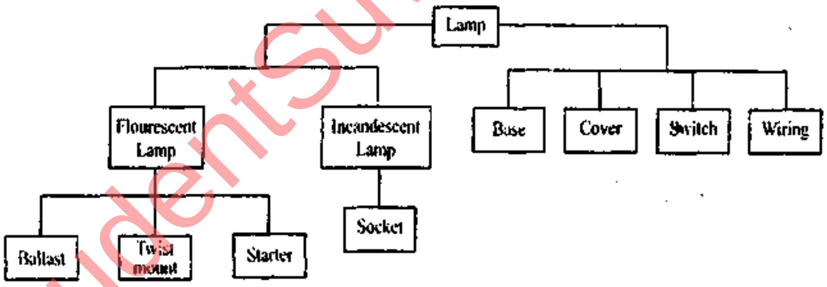


Fig. *Aggregation and generalization.*

Aggregation is sometimes called an "and-relationship" generalization an "or-relationship." A lamp is made of a base and a cover and a switch and wiring and so on. A lamp is a fluorescent lamp or an incandescent lamp.

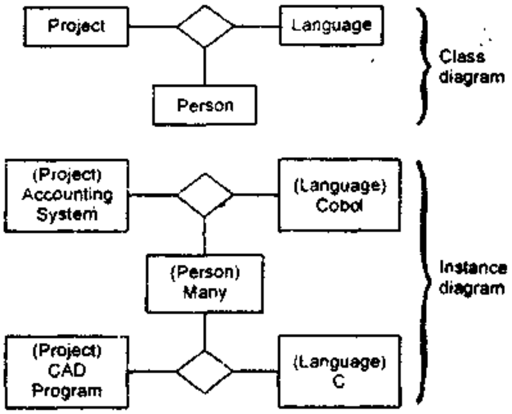
Q.3.(*b*) Explain the following terms :

(*i*) Links **(*ii*) Associations** **(*iii*) Link Attribute**

(*iv*) Multiplicity **(*v*) Role Name**

Ans. (*i*) Links : A link is a physical or conceptual connection between object instance. For example, *Joe Smith Works for Simplex Company*. Mathe-matically a link is defined as a tuple that is an ordered list of object instances. A link is an instance of an association.

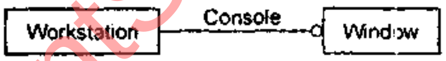
(*ii*) Association : An association describes a group of links with common structure and common semantics. For example, a *person works for a company*. All links in an association connect objects from the same classes. Association may be binary, ternary or higher order.



Ternary association and links

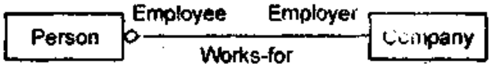
***(iii)* Link attributes :** A link attribute is a property of the links in an association. Each link attribute has a value for each link. The DMT notation for a link attribute is a box attached to the association by a loop, one or more link attributes may appear in the 2nd region of the box. This notation emphasizes the similarity between attributes for objects and attributes for links.

***(iv)* Multiplicity :** It specifies how many instances of one class may relate to a single instance of an associated class. Multiplicity constrains the number of related objects. It is often described as being "*one*" or "*many*" but more generally it is a subset of the non-negative integers. *A solid ball* is OMT symbol for "*many*" meaning zero or more. A *hollow ball* indicates "*optional*" meaning zero or one.



Zerp-or-One-multiplicity

***(v)* Role Name :** A role is one end of an association. A binary association has two roles, each of which may have a role name. A role name is a name that unequally identifies one end of association.



Employes	Employer
Joe Doe	Simplex
Mary Brown	Simplex
Jean Smith	United Widgets

Role names for an association.

Q.4. Attempt any one part of the following:

***(a)* (i) What do you understand by dynamic modelling? Explain.**

(ii) What is diffrence between state and eent?

Ans. (i) The dynamic model represents control information: The sequences of events, states, and operations that occur within a system of objects. Like the object model, the dynamic model is a pattern that specifies the allowable scenarios that may occur. the notation for the dynamic model represents a compromise between simplicity and expressiveness; there are some meaningful constraints that cannot be expressed bythe noation we present. As with the object model, these constraints must be expressed in natural language.

An event is a single that something has happened state represents the interval between events and specifies the context in which evens are interpreted. A tradisition between states represents the response to an event including the next state and possible actions and events sent to other objects. A condition is a Boolean function that controls whether a transition is allowed to occu. A state diagram is a graph of states and transition labeled by events.

An action is an instantaneous operation in response to an event, often purely formal or internal. One kind of action is sending an event to another object. Action can be attached to transitions or to entering or exiting a state. An activity is a sequence of actions that taken time to complete. An activity can be equated with state or an entire state diagram. The result of an activity can be used as a decision to choose the next state.

States and events canboth be expanded into nested stated diagrams to show greater detail. Events and state can both be organized into inheritance hierarchies. Substates inherit the transitions of their superstate. Sbevents trigger the same transitions asthe superevents.

Objects are inherently concurrent. Each object is collection that has its own state. State diagrams show concurrency as an aggregation of concurrent states, each operating independently. Concurrent objects interact by exexhanging events and by testing conditions of other objects, including state. Transitios can split or merge flow of control.

Entry and exist actions permit actions to be associated with a state, to indicate all the transitions entering or exiting the state.They make self-contained state diagrams possible for use in multiple contexts. Internal actions represent transitios that do not leavethe state.Automatic transitions fire when the their conditions are satisfied and any activity in the souce state has terminated.

States are really restriction generalizations on a class and are complementary to ordinary extension generalizatins. A subclass inherits the state diagrams of its ancestors, to be concurrent with any state diagram that it defines. It is also possible tio refine an inherited state diagram by expanding states into substates or concurrent subdiagrams.

A realistic model of a programmable thermostat takes three pages and illustrates subtleties of behavior that are not apparent from the instruction manual or from everyday operation.

(ii) State: A state is an abstraction of the attribute values and links of an objects. Sets of values are grouped together into a state according to properties that affect the gross behaviour of the object. For example the State of Bank is either solvent or insolvent depending on whether its assets exceed its liabilities.

Event: An event is something that happens at a point in time, such as user depresses left button or flight 123 depart from India.

Q.4.(*b*) What is dynamic model? What is the difference between dynamic model and object model?

Ans. Dynamic Model: Temporal relationships ae difficult to understand. A system can best to understood by first examining its static structure i.e., structure of its objcts and their relationships to each other at a single moment in time. Thus we examine changes to the objects and their relationships over time. Those aspects of a system that are concerned with time and changes are the dynamic model control is that aspects of a system that describe the sequences of operations that occur in response to external stimuli without consideration of what the oeprations do, what they operate on, or now they are implemented.

The major dynamic modelling concepts are events, which represents external stimuli and states which represents values of objects. The state diagram is standard computer science concept that has been handled in different ways depending on its use.

Relation of object and dynamic model: he dynamic model specifies allowable sequence of changes to object from the object model. A state diagram describes all or part of the behaviour of one object of given class. States are equivalence class of attribute and link values for the object. Events can be represented as operations are th eobject model.

Dynamic model structure is related to and constrained by object model structure. substate refines the attribute and link values that the object can hav. Each substate restricts the values that the object can have. But this refinement of object values is exactly generalization by restriction. A hierarchy of states of an object is equivalent to a restriction hierarchy of the object class. Object oriented models nd languages do not support restriction in the generalization hierarchy.

So the dynamic model is the proper place to reprsent it.

A single object can have different states over time the object preserves its identify, but it cannot hve different classes. Interiet differences are properly modeled as different classes, temporary differences are modeled as different states of the same class. A composite state is the aggregation of more than one of current substate.

The dynamic model of a class is inherited by its subclasses. The subclasses inherit both the states of the ancestor and the transitions. The subclasses can have their own state diagrams. The event hierarchy is independent of the class hierarchy. Evens can be defined across different classes of objects. Events are moe fundamental han states and more parallel o classes. States are defined by he interaction of objects and events. Transitions more expressive than operations, because the effect of anevent dependsnot only on the class of an object but also on its state.

Q.5. Attempt any one part of the following:

(a) What is Jackson Structured Development methodologies to software engineering?

Ans. Jackson Structured Development: Jackson Structured Development (JSD) is another mature methodology which has a different style than SA/SD or OMT. The JSD methodology was developed by Michael Jackson and is especially popular in Europe. JSD does not distinguish between analysis and design and instead lumps analysis and design and instead lumps both phases together as specification. JSD divides system development into two stages; specification, then implementation. JSD first determines the "what" and then the "how". JSD is intended especially for applications in which timing is important.

A JSD model begins with consideration of the real world. The purpose of a system is to provide functionality, but Jackson feels that one must first consider how this functionality fits in with real world. A JSD model describes the real world in terms of entities, actions and ordering of actions. Entities usually appear as nouns in requirement statements and actions appear as verbs. JSD software development consist of six sequential steps: entity action step, entity structure step, initial model step, and implementation step.

Jackson presents several examples, one of which is the design of an elevator control system. The elevator control system controls two elevators which services six floors. Each elevator has six inside buttons (one for each floor). Each floor has Up and Down buttons in the waiting area. Jackson identifies two entities for elevator control example: button and elevator.

Entity structure step: Action occurs in the real world and is not an artifact of the system. Action takes place at a point in time, are atomic, and not decomposable. The entity structure step partially orders the actions of each entity by time. The elevator control system illustrates the importance of ordering actions. It is permissible for an elevator to arrive at floor 3, leaving floor 3, arrive at floor 2, leave floor 2 and so on. It does not make sense for two arrive actions to occur in succession; arrive and leave operations must alternate.

Initial Order: The initial model step states how the real world connects to the abstract model. JSD supports state-vector and data stream connection. The elevator control system illustrates state-vector connection. The elevator user dose not want the control system to remember each button pressed and send an elevator fine times to service request. The JSD model of the computer system is unaware of the number of presses and only communicates with the real world in the "up-flag".

Function step: It uses pseudocode to state output of actions. At the end of this step the developer has a complete specification of the required system in the elevator example, turning the display panel lights on or off as an elevator arrives at each floor is a function that must be specified.

System Timing step: This step considers how mush the model is permitted to lag the real world. For the most part, the result of the timing step is a set of informal notes on performance constraints. For example, an elevator control system must detect when up and down buttons are pressed.

Implementation Step: This focuses on the problems of process scheduling and allocates processors to processes. The number of processes may be different from the number of precessors. Jackson's elevator control model has 50 processes. The developer must decide whether to match each process to one or 50CPU's or how to get several processes to share the same CPU. After the six JSD steps comes writing of code and database design.

Q.5.(*b*) Compare between SA/SD and OMT modelling.

Ans. Similarities :

1. The OMT and SA/SD methodologies both incorporate similar modelling components.

2. Both methodologies support three orthogonal views of a system – *the object, dynamic and functional models.*

Differences:

1. The OMT and SA/SD methodologies differ in the relative emphasis that they place on the various modeling components.

2. *In the SA/SD approach* the functional model dominates, the dynamic model is next important and the object model least important. OMT modelling regards object model as most important then the dynamic model and finally the functional model.

3. *SA/SD* organises a system around procedures. *An object oriented design technique* organises a system around real-world object or conceptual objects that exist in the user's view of the world.

4. An SA/SD design has a clearly defined system boundary, across which the software procedures must communicate with real world. It is difficult to extend a SA/SD design to a new boundary. It is much easier to extend *an object-oriented* design, one merely adds object and relationship near the boundary to represent objects that existed previously only in the outside world.

Q.6. Attempt any one part of the following:

(a) Write a Java program to read in two matrices from the keyboard and compute their sum. Overload toString( ) method to display the result matrix in row and column form.

Ans.

```
import java . io . *;
class MyMatrix {
public static void main (string args [ ])
throws IOException
{
BufferedReader br = new
    BufferReader (new Input Stream Reader (System.in));
    String str;
    int i) j;
    int matrix1[ ][ ] = new int [4][5];
    int matrix2[ ][ ] = new int [4][5];
    int matrix3[ ][ ] = new int [4][5];
    System . out . printh ("Enter first matrix")
    for (i = 0; i < 4; i++)
        for(j = 0; j < 5; j++)
        {
            str = br.readLine( );
            try {
                matrix1[i][j] = Integer . parseInt(str);
            }catch (Number Format Exception e){
                System . out . println("Invalid format");
        }
    System . out . println("\n Enter Scond Matrix");
    for(i = 0; i < 4; i++)
        for(j = 0; j < 5; j++)
        {
            Str = br . read lin( );
            try{
                    matrix2[i][j] = Integer . parseInt(str);
                }   catch (Number Format Exception e){
                    system . out . println ("Invalid format");
                }
            for (i = 0; i < 4; i++)
            {
                for(j = 0; j < 5; j++)
                {
                    matrix3[i][j] = matrix1[i][j] + matrix2[i][j];
                    system . out . print
                                                                    (matrix3[i][j]+"\t");
                }
                    system . out . println("n");
            }
        }
    }
```

Q.6.(*b*) Write a short note on the following:

(a) Java virtual machine

(b) Multithreading

(c) Utility classes

(d) Applets

(e) Layout managers

Ans. (a) Java virtual machine: The java compiler produces an intermedia code known as byte code for a machine that doesn't text. This machine is called java virtual machine and it exists only inside the compute memory.

The process of compiling a java proramming into byte code which is also referred to as virtual mchine code.

Java	Java	Vitual
program	compiler	- lachine
Source code		Byte code
BYTE code	Java	Machine
Virtual machine	interpreter	code

(b) Multithreading: Multithreading means handling multiple tasks simultaneously. Java supports multithreaded programs. This means that we need not wait for the application tio finish one task befoe beginning another. This feature greately improves the interactive performance of graphical applications. Java is inherently multi-threaded. This makes java very responsive to user input. It also helps to contribute to java is rubustem and provides a mechanism whereby the java environment can ensure, that a malicions applet does not steal all of the host's CPU cycle.

(c) Utility classes: The java.util package contains some to the most exiciting enhancements added by java collections. The java.util classes are listed below:

- Abstract collection (Java 2)
- Abstract list (Java 2)
- Abstract map (Java 2)
- Abstract sequential list (Java 2)
- Abstract set (Java 2)

- Array list (Java 2)
- Arrays (Java 2)
- Bitset
- Clander
- Collections (Java 2)
- Date
- Dictionary
- Event object
- Gregerial calender
- Hash map (Java 2)
- Hash set (Java 2)
- Hash table
- Stuct
- String tokenizer
- Time zone etc.

(d) Applet: Java applet is a program that appars, embedded in a web document and is meant to be hosted by a web browser. which provides a home for the applet. Any applet depends on a java capable browser in order to run it. It can also be viewed usign a tool called the applet viewer. A java applet teceives input from th web browser regarding initialization, loading startin shopping. destroying etc.

(e) Layout manager: A layout manager automatically arranges your conrol within a window by using some type of alogorithm. Each container object has a layout manager associated with it. A layour manager is an instance of any class that implements the layout manager interface. The layout managers set by the set Layout( ) method. Whenever a contains is resized (or sized for the first time), the layout manager is used to position each of the components within it.

Types of Layout

* **Flow layout:** It is the default layout manager. This layout implement or simple layout style similar to how words flew by text edition.

* **Border layout:** The border layout class implements a commn layout style for top-level window.

Q.7. Attempt any one part of the following:

(a) What are the Java Bean? What are their advantage? What are various introspection Mechanisms to inter information about a Bean? Explain with a suiable exmple.

Ans. JavaBeans enables software developers to design and create reusable software components that inegrate with each other, with applications, and with development tools. JavaBeans, or 'Ban' can be an entire application used alone or embedded within other application or a component that can be used within an application.

JavaBeans is a set of java objects that provides the infrastructure for the components to be writtenin java Language. JavaBeans includes objects for Customizing, Introspecting and Event handling. It provides the component famework from which reusable components can be built.

JavaBeans provides the mans to get component information. Using the Bean Development Kit, the users can implement their own components as well as describe theinterface specificatios to their own components. It also enables the users to define the event mechanisms, properties and public methods.

Advantages of Java Beans:

(a) A bean inherits all the benefits of java's 'Write-once, run anywhere' paradigm.

(b) The properties, events and methods of a Bean can be controlld.

(c) Auxiliary software can be provided to help configure a Bean.

(d) the configuration setting of a Bean can be saved and restored at a later time.

(e) A can receive events from other objects and generate evens tha are sent to other objects:

Introspection is the process of analyzing a Bean tio detrmine is capabilities. This feature allows another application to obtain information about a component — Without introspection, the Java Bean technology would not operate.

There are two ways in which the developer of a bean can indicate which of its properties, events and method should be exposed. With the first name simple naming convesiios are used. these allow the introspection mchanisms to infer informations about a Bean. In the second way, an additional class that extends the Bean Infor in terface is provided that explicity supplies this information.

Design Patterns for Properties: A *property* i sa subset of a Bean's state. The values assigned to the properties determine the behavior and ap pearance of that component. A property is set throuigh a *setter* method. A property is obtained by a *getter* method. There are twotypes of properties: simple and indexed.

Simple Properties: A simple property has a single value. It can be identified by the followin design patterns where N is the name of the property and T is its type:

Public T getN( );

Public void setN(T arg);

Indexed Properties: An indexed property consists of multiple values. It can be identified by the following design patterns, where N is the name of the property and T is its type:

Puiblic T getN(int *index*);

Public void setN(int *index*, T *value*);

Public T[ ]getN( );

Public void setN(T values[ ]);

Design patterns for Events: beans use the delegation event model that was discussed earlier in this book. Beans can generate event s and send them to other objects. These canbe identified by the following design patterns, where T is the type of the event:

public void add TListener (TListener *eventListener*);

Public void remove TListener(TListener *eventListener)*;

These methods are used to add or removea listener forthe specfied event.

Using the Beaninfor Interface: The BeanInfor interface enablesyou to *explicitly* control what infomation is available.The beanInfor interface defines several methods, including these:

PropertyDescriptor[ ] getPropertyDescriptors( )

EventSetDescriptor[ ] getEventSetDescriptors( )

MethodDescriptor[ ] getMethodDescriptors( )

They return arrays of objects that provide information about the properties, events, andmethods of a Bean.

Q.7.(*b*) Write short note on any three of the following:

(a) JDBC

(b) Adjustment events

(c) FontMetrics class

(d) java constructs and features that facilitate reuse

Ans. (a) JDBC: JDBC stands for java database connectivity. It was developed by Java soft and is a part of th Java Enterprise API. JDBC is the first standardized API that allows the users to develop database front ends without continuously rewriting their code. JDBC provides the ability to create robust, platform-independent application and web-based applets, which can accers any database through a DBS-independent mechanism. The interactionbetween theseap-plication and applets with SQL data sources is done through a set of relational database objects and methods defind is the JDBC API.

(b) Adjustment events: An adjustment event is generated by a scroll bar. There are five types of adjustment events. he adjustment event class defines integer constants that can be used to identify them. the constants and their meaning are below:

1. **Block-Decrement:** The user clicked inside the scroll bar to decrease its value.
2. **Block-Increment:** The user clicked inside the scroll bar to increase its value.
3. **Track:** The slider was dragged.
4. **Unit-Decrement:** The button at the end of the scroll bar was clicked to decrease its value.
5. **Unit-Increment:** The button at the end of the scroll bar was clicked to increase its value.

(c) Front metrics class: It encapsulates various information about a font. There are common terminology used when descenting fonts.

- **Height:** The top-to-botton size of the tallest character in the font.
- **Baseline:** The line that the bottoms of characters are aligned to (not counting descent).
- **Ascent:** The distance from the baseline to the top of a character.
- **Descent:** The distance fromthe baseline to the bottom of a character.

- **Leading:** The distance between the bottom of one line of text and top of the next.

(d) Java constructors and eatures that facilitales reuse: The final stage of intialyation of a new object is to call a method calle a constructor.

Constructors are identified by the following rules:

- The method name must exactly match the class name.
- There must not be a return type declared for the method.

Default constructor: Every class hs at least one constructor. If you don't write any constructor for your class. java provides a constructor with no arguments and an empty body. This default constructor allows youto create anobject with the new operator by using syntax new ABC( ), where ABC is a class name.

Inheritance and interfaces is a way which faciliate reuseability in jave.

Inheritance is a technique to incorpoate code. This means that we can add renability additional feature to an existing class without modifying it. This is possible by deriving a new class from the existing one. The new clas swill hae the combined feature of both the classes.

Interfaces are used for multiple inheritance in java.