JEPPIAAR ENGINEERING COLLEGE

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

QUESTION BANK

CS8391
OBJECT ORIENTED PROGRAMMING

II YEAR – III SEM
2017 -2021 BATCH

Vision of Institution: To build Jeppiaar Engineering College as an Institution of Academic Excellence in Technical education and Management education and to become a World Class University.

Mission of Institution

M1	To excel in teaching and **learning, research and innovation** by promoting the principles of scientific analysis and creative thinking
M2	To participate in the production, **development and dissemination of knowledge** and interact with **national and international communities**
M3	To equip students with values, ethics and life skills needed to enrich their lives and enable them to meaningfully contribute to the progress of society
M4	To prepare students for higher studies and lifelong learning, enrich them with the practical and entrepreneurial skills necessary to excel as future professionals and contribute to Nation's economy

Vision of Department: To emerge as a globally prominent department, developing ethical computer professionals, innovators and entrepreneurs with academic excellence through quality education and research.

Mission of Department

M1	To create **computer professionals** with an ability to identify and **formulate the engineering problems** and also to provide **innovative solutions** through **effective teaching learning process.**
M2	To **strengthen the core-competence** in computer science and engineering and to create an ability to **interact** effectively with industries.
M3	To produce engineers with good professional skills, **ethical values** and life skills for the **betterment of the society.**

M4	To encourage students towards **continuous and higher level learning** on technological advancements and provide a platform for **employment and self-employment.**

PROGRAM OUTCOMES (POs)

PO1	**Engineering Knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of computer science engineering problems.
PO2	**Problem analysis:** Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
PO3	**Design/development of solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.
PO4	**Conduct investigations of complex problems:** Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
PO5	**Modern tool usage:** Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.
PO6	**The engineer and society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
PO7	**Environment and sustainability:** Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.
PO8	**Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
PO9	**Individual and team work:** Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.

PO10	**Communication:** Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
PO11	**Project management and finance:** Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
PO12	**Life-long learning:** Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

PROGRAMME EDUCATIONAL OBJECTIVES (PEOs)

PEO 01: To address the real time complex engineering problems using innovative approach with strong core computing skills.

PEO 02: To apply core-analytical knowledge and appropriate techniques and provide solutions to real time challenges of national and global society.

PEO 03: Apply ethical knowledge for professional excellence and leadership for the betterment of the society**.**

PEO 04: Develop life-long learning skills needed for better employment and entrepreneurship.

PROGRAMME SPECIFIC OUTCOME (PSOs)

PSO1 – An ability to understand the core concepts of computer science and engineering and to enrich problem solving skills to analyze, design and implement software and hardware based systems of varying complexity.

PSO2 - To interpret real-time problems with analytical skills and to arrive at cost effective and optimal solution using advanced tools and techniques.

PSO3 - An understanding of social awareness and professional ethics with practical proficiency in the broad area of programming concepts by lifelong learning to inculcate employment and entrepreneurship skills.

OOPS -II

C202.1	
C202.2	
C202.3	
C202.4	
C202.5	

CS6301 PROGRAMMING AND DATA STRUCTURES II **L T P C**
3 0 0 3

UNIT I INTRODUCTION TO OOP AND JAVA FUNDAMENTALS 10

Object Oriented Programming - Abstraction – objects and classes - Encapsulation- Inheritance - Polymorphism- OOP in Java – Characteristics of Java – The Java Environment - Java Source File -Structure – Compilation. Fundamental Programming Structures in Java – Defining classes in Java – constructors, methods -access specifiers - static members -Comments, Data Types, Variables, Operators, Control Flow, Arrays , Packages - JavaDoc comments.

UNIT II INHERITANCE AND INTERFACES 9

Inheritance – Super classes- sub classes –Protected members – constructors in sub classes- the Object class – abstract classes and methods- final methods and classes – Interfaces – defining an interface, implementing interface, differences between classes and interfaces and extending interfaces - Object cloning -inner classes, ArrayLists - Strings

UNIT III EXCEPTION HANDLING AND I/O 9

Exceptions - exception hierarchy - throwing and catching exceptions – built-in exceptions, creating own exceptions, Stack Trace Elements. Input / Output Basics – Streams – Byte streams and Character streams – Reading and Writing Console – Reading and Writing Files

UNIT IV MULTITHREADING AND GENERIC PROGRAMMING 8

Differences between multi-threading and multitasking, thread life cycle, creating threads, synchronizing threads, Inter-thread communication, daemon threads, thread groups. Generic

Programming – Generic classes – generic methods – Bounded Types – Restrictions and Limitations.

UNIT V EVENT DRIVEN PROGRAMMING 9

Graphics programming - Frame – Components - working with 2D shapes - Using color, fonts, and images - Basics of event handling - event handlers - adapter classes - actions - mouse events - AWT event hierarchy - Introduction to Swing – layout management - Swing Components – Text Fields , Text Areas – Buttons- Check Boxes – Radio Buttons – Lists- choices- Scrollbars – Windows –Menus – Dialog Boxes.

TOTAL: 45 PERIODS

TEXT BOOKS

1. Herbert Schildt, "Java The complete reference", 8th Edition, McGraw Hill Education, 2011.

2. Cay S. Horstmann, Gary cornell, "Core Java Volume –I Fundamentals", 9th Edition, Prentice Hall, 2013.

REFERENCES

1. Paul Deitel, Harvey Deitel, "Java SE 8 for programmers", 3rd Edition, Pearson, 2015.

2. Steven Holzner, "Java 2 Black book", Dreamtech press, 2011.

3. Timothy Budd, "Understanding Object-oriented programming with Java", Updated Edition, Pearson Education, 2000.

UNIT I

INTRODUCTION TO OOP AND JAVA FUNDAMENTALS 9

Object Oriented Programming - Abstraction – objects and classes - Encapsulation- Inheritance - Polymorphism- OOP in Java – Characteristics of Java – The Java Environment - Java Source File -Structure – Compilation. Fundamental Programming Structures in Java – Defining classes in Java – constructors, methods -access specifiers - static members -Comments, Data Types, Variables, Operators, Control Flow, Arrays , Packages - JavaDoc comments.

Q.No	Questions	CO	Bloom's Level
1	**Define Objects and classes in java. NOV/DEC 2010 , MAY/JUNE 2014** Class is a template for a set of objects that share a common structure and a common behaviour. A class is a blueprint, or prototype, that defines the variables and the methods common to all objects of a certain kind. An object is a software bundle of variables and related methods.An instance of a class depicting the state and behavior at that particular time in		BTL1

	real world. Object is an instance of a class. It has state,behaviour and identity. It is also called as an instance of a class.		

2	**How does one import a single package? NOV/DEC 2010**
	It is easily achieved using an import statement in which the import keyword is followed by the fully qualified name of the member you wish to use. Consider the example given previously, but revised to use this method:
	`import java.util.Vector;`
	`class Test {`
	`  Vector vector;`
	`  Test() {`
	`  vector = new Vector();`
	`  }`
	`}`
			BTL1

3	**What is the default access to a member in a class? MAY/JUNE – 2011** Default is no access specifier. For classes, and interface declarations, the default is package private. This falls between protected and private, allowing only classes in the same package access. (protected is like this, but also allowing access to subclasses outside of the package. For interface members (fields and methods), the default access is public. But note that the interface declaration itself defaults to package private.		BTL1
4	**What is a package? MAY/JUNE – 2011 , NOV/DEC 2014** A package is a namespace that organizes a set of related classes and interfaces. Conceptually can think of packages as being similar to different folders on our computer. Because software written in the Java programming language can be composed of hundreds or *thousands* of individual classes, it makes sense to keep things organized by placing related classes and interfaces into packages.		BTL1
5	**Define class. NOV/DEC 2011** Class is a template for a set of objects that share a common structure and a common behaviour. A class is a blueprint, or prototype, that defines the variables and the methods common to all objects of a certain kind.		BTL1
6	**List any four Java Doc comments. NOV/DEC 2011** Javadoc is a program shipped with JDK that you can use to run over your source code to produce documentation of your classes in the same type of HTML files that Sun Microsystems has used for its Java API documentation. To use javadoc		BTL1

	on your source code, you have to tag your code with a certain type of comment formats. A simple example of Javadoc comments looks like this: /**Class MyButton implements java.io.Serailizable, extends java.awt.Button */ public class MyButton { /** Javadoc utility enables you to keep the code and the documentation in sync easily. The javadoc utility lets you put your comments right next to your code, inside your ".java" source files. All you need to do after completing your code is to run the Javadoc utility to create your HTML documentation automatically.		

7	**What is meant by private access specifier? <u>NOV/DEC – 2011</u>**
	
	Java offers four access specifiers, listed below in decreasing accessibility:
	
	• public
	• protected
	• default (no specifier)
	• private
	
	Private
	
	Private methods and fields can only be accessed within the same class to which the methods and fields belong. private methods and fields are not visible within subclasses and are not inherited by subclasses. So, theprivate access specifier is opposite to the public access specifier. It is mostly used for encapsulation: data are hidden within the class and accessor methods are provided. An example, in which the position of the upper-left corner of a square can be set or obtained by accessor methods, but individual coordinates are not accessible to the user.
	
	public class Square { // public class
	private double x, y // private (encapsulated) instance variables
	public setCorner(int x, int y) { // setting values of private fields
	this.x = x;
	this.y = y;
	}
			BTL1

	public getCorner() { // setting values of private fields
	return Point(x, y);
	}
	}		

8	**What is the need for javadoc multiline comments? NOV/DEC – 2011** Javadoc comments are any multi-line comments ("/** ... */") that are placed before class, field, or method declarations. They must begin with a slash and two stars, and they can include special tags to describe characteristics like method parameters or return values. The HTML files generated by Javadoc will describe each field and method of a class, using the Javadoc comments in the source code itself.		BTL1
9	**What do you mean by instance variables? MAY/JUNE – 2012** ● Instance variables are declared in a class, but outside a method, constructor or any block.When a space is allocated for an object in the heap a slot for each instance variable value is created. ● Instance variables are created when an object is created with the use of the key word 'new' and destroyed when the object is destroyed. ● Instance variables hold values that must be referenced by more than one method, constructor or block, or essential parts of an object.s state that must be present through out the class. ● Instance variables can be declared in class level before or after use. ● Access modifiers can be given for instance variables. ● Instance variables can be accessed directly by calling the variable name inside the class. However within static methods and different class (when instance variables are given accessibility) the should be called using the fully qualified name *.ObjectReference.VariableName*.		BTL5
10	**Mentions the purpose of finalize method. MAY/JUNE 2012** The **Object** class provides a callback method, ***finalize()***, that **may be** invoked on an object when it becomes garbage. **Java** provides a mechanism to run some code just before our object is deleted by the **Garbage Collector**. This code is located in a method named ***finalize()***. When **JVM** trying to delete the objects, then some objects refuse to delete if they held any resource. If our object opened up some resources, and we woluld like to close them before our object is deleted. So before **Garbage** operation we have to clean up the resources which the object held on, that clean up code we have to put in ***finalize()*** method. - Can override ***finalize()*** to do cleanup, such as freeing resources. Any code that we put into our Class's overridden ***finalize()*** method is not guarunteed to run, so don't provide essential code in that method.		BTL1

11	**What are the key characteristics of objects?** **MAY/JUNE 2013** Three key characteristics of objects are: ● **The Object's behaviour** – What can you do with this object , or what methods can you apply to it? ● **The Object's state** – How does the object react when you apply those methods? ● **The Object's identity** – How is the object distinguished from others that may have the same behaviour and state		BTL1
12	**Define finalize method.** **MAY/JUNE 2013** finalize( ) method is used just before an object is destroyed and can be called just prior to garbage collection.		BTL1
13	**Mention the features of JAVA.** **NOV/DEC 2013** **The features of JAVA are :** ➢ Compiled and Interpreted ➢ Platform-Independent and Portable. ➢ Robust and secure ➢ Distributed ➢ Familiar, Simple and Small ➢ Multithreaded and Interactive ➢ High Performance ➢ Dynamic and Extensible.		BTL1
14	**List the access specifiers used in java ? NOV/DEC 2014** Access Modifiers | Default | private | protected | public Accessible inside the class | yes | yes | yes | yes Accessible within the subclass inside the same package | yes | no | yes | yes Accessible outside the package | no | no | no | yes Accessible within the subclass outside the package | no | no | yes | yes		BTL1

15	**Define the characteristics of objects? APR/MAY 2015** An object has **identity** (each object is a distinct individual). An object has **state** (it has various properties, which might change). An object has **behavior** (it can do things and can have things done to it).		BTL1
16	**What is Abstract class? APR/MAY 2015** Abstract class is a class that has no instances. An abstract class is written with the expectation that its concrete subclasses will add to its structure and behaviour, typically by implementing its abstract operations.		BTL1
17	What is static and dynamic binding in java ? **MAY/JUNE 2014** **static binding** When type of the object is determined at compiled time(by the compiler), it is known as static binding. If there is any private, final or static method in a class, there is static binding. **Dynamic binding** When type of the object is determined at run-time, it is known as dynamic binding.		BTL1
18	**Name any three tags used in Java Doc Comment** Java supports three types of comments. The first two are the // and the /*. The third type is called a documentation comment. It begins with the character sequence /**. It ends with*/. In Java have javadoc tags Tag Meaning @author Identifies the author of a class @deprecated Specifies that a class or member is deprecated @param Documents a method's parameter @return Documents a method's return value		BTL1
19	**What is method overloading and method overriding?** When a method in a class having the same method name with different arguments is said to be method overloading. Method overriding : When a method in a class having the same method name with same arguments is said to be method overriding.		BTL1
20	**Define arithmetic operators in java?** Arithmetic operators perform the same basic operations you would expect if you used them in mathematics (with the exception of the percentage sign). They take two operands and return the result of the mathematical calculation. Java has five arithmetic operators: + to add two numbers together or concatenate two Strings.		BTL1

	- to subtract one number from another. * to multiply one number by another. / to divide one number by another. % to find the remainder from dividing one number by another.		
21	**Define bitwise operator.** Bitwise operators perform operations on the bits of their operands. The operands can only be byte, short, int, long or char data types. For example, if an operand is the number 48, the bitwise operator will perform its operation on the binary representation of 48 (i.e., 110000). There are three binary logical operators: & performs a logical AND operation. | performs a logical OR operation. ^ performs a logical XOR operation.		BTL1
22	**Define Precedence order.** When two operators share an operand the operator with the higher *precedence* goes first. For example, 1 + 2 * 3 is treated as 1 + (2 * 3), whereas 1 * 2 + 3 is treated as (1 * 2) + 3 since multiplication has a higher precedence than addition.		BTL1
23	**What are methods and how are they defined?** Methods are functions that operate on instances of classes in which they are defined. Objects can communicate with each other using methods and can call methods in other classes. Method definition has four parts.		BTL1
24	**What are different types of access modifiers (Access specifiers)?** Access specifiers are keywords that determine the type of access to the member of a **class.** These keywords are for allowing privileges to parts of a program such as functions and variables. These are: ***public*:** Any thing declared as public can be accessed from anywhere. ***Private*:** Any thing declared as private can't be seen outside of its class. ***Protected*:** Any thing declared as protected can be accessed by classes in the same package and subclasses in the other packages. ***Default modifier*:** Can be accessed only to classes in the same package.		BTL1
25	**What is an Object and how do you allocate memory to it?** Object is an instance of a class and it is a software unit that combines a structured set of data with a set of operations for inspecting and manipulating that data. When an object is created using new operator, memory is allocated to it.		BTL1
26	**list the usage of Java packages.** This is a way to organize files when a project consists of multiple modules. It also		BTL1

	helps resolve naming conflicts when different packages have classes with the same names. Packages access level also allows protecting data from being used by the non-authorized classes.		
27	**What gives java it's "write once and run anywhere" nature?** All Java programs are compiled into class files that contain bytecodes. These byte codes can be run in any platform and hence java is said to be platform independent.		BTL1
28	**What is static variable and static method?** Static variable is a class variable which value remains constant for the entire class. Static method is the one which can be called with the class itself and can hold only the static variables.		BTL1
29	**Differentiate between a Class and an Object?** The Object class is the highest-level class in the Java class hierarchy. The Class is used to represent the classes and interfaces that are loaded by a Java program. The Class class is used to obtain information about an object's design. A Class is only a definition or prototype of real life object. Whereas an object is an instance or living representation of real life object. Every object belongs to a class and every class contains one or more related objects		BTL2
30	**How is polymorphism achieved in java?** Inheritance, Overloading and Overriding are used to acheive Polymorphism in java.		BTL1

PART-B

Q.No	Questions	CO	Bloom's level
1	**(i) What is a constructor? What is the use of new method? (4)** **NOV/DEC 2010** Refer page no: 144 **(ii) Give the syntax for static method and its initialization. (4)**		BTL5

	NOV/DEC 2010 Refer page no: 132 **(iii) Explain arrays in java. (8)** **NOV/DEC 2010** Refer page no:90		
2	**(i)What is class? How do you define a class in java? (4)** **NOV/DEC 2010** Refer page no:107 **(ii) Explain the following in Strings : (4)** **NOV/DEC 2010** **(1) Concatenation** **(2) Substrings.** Refer page no:53 **(iii) Explain any four methods available in string handling. (4)** **NOV/DEC 2010** Refer page no: 53 **(iv) What is a Package? How does a complier locate packages? (4)** **NOV/DEC 2010** Refer page no:153		BTL5
3	**i) Explain what is OOPS and explain the features of OOPS. (8) NOV/DEC 2011, MAY/JUNE 2014** Refer page no: 106 **(ii) Discuss about the usage of constructor with an example. Using Java. (8) NOV/DEC 2011, MAY/JUNE 2014** Refer page no: 144		BTL5
4	**(ii) Define package. Explain the types of package with its importance. (8) NOV/DEC 2011,** **What is package? How to add a class into a package? Give example. (8) MAY/JUNE 2013** Refer page no:153		BTL5
5	Explain briefly about object oriented programming concepts differ from structured programming concepts. (16) NOV/DEC 2011, How is OOP different from procedural programming language? (8) MAY/JUNE 2013 Refer page no: 106		BTL5

6	**Write a java program for push and pop operations in stack using arrays in classes and object. (16)** **NOV/DEC 2011** Refer page no:90		BTL6
7	**Write a java program to sort ten names in descending order. (16)** **MAY/JUNE 2012** Refer notes		BTL6
8	**i) Describe the concept of OOP. (5)** **NOV/DEC 2013** Refer page no: 106 **(ii) What is meant by overriding method? Give example. (5)** **MAY/JUNE 2014** Refer page no: 171 **(iii) Write a JAVA program to reverse the given number. (6)** Refer notes		BTL6
9	**What is meant by package? How it is created and implemented in JAVA. (8) NOV/DEC 2013** **(ii) Write a JAVA program to find the smallest number in the given list. (8)** Refer page no:153		BTL6
10	Write a Java program to sort the elements in increasing order ? **NOV/DEC 2014** **Refer notes**		BTL6
11	**i) Define class? (3) APR/ MAY 2015** **ii)Write short notes on Access specifiers (3) APR/ MAY 2015** **iii) string in JAVA** **iv) Explain the term static fields and methods and explain its types with examples (8) APR/ MAY 2015**		BTL5
12	**Define array. What is array sorting and explain with an example? APR/ MAY 2015**		BTL5

13	**State and explain documentation comments in java(8) APR/ MAY 2015**		BTL5

UNIT II

UNIT II INHERITANCE AND INTERFACES 9

Inheritance – Super classes- sub classes –Protected members – constructors in sub classes- the Object class – abstract classes and methods- final methods and classes – Interfaces – defining an interface, implementing interface, differences between classes and interfaces and extending interfaces - Object cloning -inner classes, ArrayLists - Strings

Q.No	Questions	CO	Bloom's Level

1	**What is meant by parameter passing constructors? Give example. NOV/DEC 2013**
	Parameter passing constructors are called Parameterized Constructors. Parameterized Constructor is a constructor with parameters.
	
	Ex: Class test // class name
	{
	int st.no;
	char st.name;
	int m1,m2,m3,total;
	float avg;
	
	test(no,name,t1,t2,t3) // parametized constructor
	st.no = no;
	st.name = name;
	m1 = t1;
	m2 = t2;
	m3 = t3;
	};
			BTL1

2	**What is Constructor? APR/MAY 2015** A constructor is a special method of a class or structure in object-oriented programming that initializes an object of that type. A constructor is an instance method that usually has the same name as the class, and can be used to set the values of the members of an object, either to default or to user-defined values.		BTL1

3	**What is Abstract class? APR/MAY 2015** Abstract class is a class that has no instances. An abstract class is written with the expectation that its concrete subclasses will add to its structure and behaviour, typically by implementing its abstract operations.		BTL1
4	**What is meant by Inheritance Hierarchy? Give an Example. NOV/DEC 2010** **Definitions:** A class that is derived from another class is called a *subclass* (also a *derived class*, *extended class*, or *child class*). The class from which the subclass is derived is called a *superclass* (also a *base class* or a *parent class*). The idea of inheritance is simple but powerful: When you want to create a new class and there is already a class that includes some of the code that you want, you can derive your new class from the existing class. In doing this, you can reuse the fields and methods of the existing class without having to write (and debug!) them.		BTL1
5	**In java what is the use of Interfaces? NOV/DEC 2010** In the Java programming language, an *interface* is a reference type, similar to a class, that can contain *only* constants, method signatures, and nested types. There are no method bodies. Interfaces cannot be instantiated—they can only be *implemented* by classes or *extended* by other interfaces. To use an interface, we have to write a class that *implements* the interface. When an instantiable class implements an interface, it provides a method body for each of the methods declared in the interface.		BTL2

6	**How to prevent inheritance? MAY/JUNE 2011**
	To stop a class being extended, the class declaration must explicitly say it cannot be inherited. This is achieved by using the "final" keyword:
	public **final** class Account {
	}
	This means that the Account class cannot be a superclass and the OverdraftAccount class can no longer be its subclass.
			BTL1

7	**Define Inheritance Hierarchy. NOV/DEC 2011** The Collection of all classes extending from a common superclass is called an inheritance hierarchy. The path from a particular class to its ancestors in the inheritance is its inheritance chain.		BTL1

8	**What is Interface?** **NOV/DEC 2011** An interface is basically a kind of class. Like classes, interfaces contain methods and variables but but with a major difference.The difference is that interfaces define only abstract methods and final fields. This means that interfaces do not specify any code to implement these methods and data fields contain only constants		BTL1
9	**What is object cloning? NOV/DEC 2011 , How to object clone? MAY/JUNE 2013, What is meant by object cloning? NOV/DEC 2013, MAY/JUNE 2014** It is the process of duplicating an object so that two identical objects will exist in the memory at the same time. In Java, Assign an object to another variable, only the memory address of the object is copied and hence any changes in the original object will be reflected in the new variable. *Cloning* means creating a copy of the object. The precise meaning of "copy" may depend on the class of the object. The general intent is that, for any object x, the expression:		BTL1
10	**Differentiate copying and cloning.** **MAY/JUNE 2011,** • clone - create something new based on something that exists. • copying - copy from something that exists to something else (that also already exists).		BTL2
11	**What are the conditions to be satisfied while declaring abstract classes? MAY/JUNE 2012, NOV/DEC 2014** An abstract class is a class that is incomplete, or to be considered incomplete.		BTL1
12	Define inner classes. Why would you want to do that? Inner class: classes defined in other classes, including those defined in methods are called inner classes. An inner class can have any accessibility including private.		BTL1
13	**Can we override a super class method which is throwing an unchecked exception with checked exception in the sub class?**		BTL1

	No. If a super class method is throwing an unchecked exception, then it can be overridden in the sub class with same exception or any other unchecked exceptions but can not be overridden with checked exceptions.		

14	**What is meant by an innerclass?**
	
	An inner class is a nested class whose instance exists within an instance of its enclosing class and has direct access to the instance members of its enclosing instance
	class <EnclosingClass>
	{
	class <InnerClass>
	{
	
	}
	}
			BTL1

15	**What is the use of 'Super' Keyword? Give an example.** Usage of 'super' keyword' The first calls the superclass constructor To access a member of the superclass that has been hidden by a member of a subclass		BTL1
16	**What is the difference between String and String Buffer?** a) String objects are constants and immutable whereas StringBuffer objects are not. b) String class supports constant strings whereas StringBuffer class supports growable and modifiable strings.		BTL1
17	**What is the difference between this () and super ()?** This () can be used to invoke a constructor of the same class whereas super() can be used to invoke a super class constructor.		BTL1
18	**What are interface and its use?** Interface is similar to a class which may contain method's signature only but not bodies and it is a formal set of method and constant declarations that must be defined by the class that implements it. Interfaces are useful for: a) Declaring methods that one or more classes are expected to implement b) Capturing similarities between unrelated classes without forcing a class relationship. c) Determining an object's programming interface without		BTL1

	revealing the actual body of the class.		
19	**Can you have an inner class inside a method and what variables can you access?** Yes, we can have an inner class inside a method and final variables can be accessed.		BTL1
20	**What is the difference between abstract class and interface?** a) All the methods declared inside an interface are abstract whereas abstract class must have at least one abstract method and others may be concrete or abstract. b) In abstract class, key word abstract must be used for the methods whereas interface we need not use that keyword for the methods. c) Abstract class must have subclasses whereas interface can't have subclasses.		BTL2
21	**What is the difference between a static and a non-static inner class?** A non-static inner class may have object instances that are associated with instances of the class's outer class. A static inner class does not have any object instances.		BTL2
22	**Define superclass and subclass?** Superclass is a class from which another class inherits. Subclass is a class that inherits from one or more classes.		BTL1
23	**What is reflection API? How are they implemented?** Reflection is the process of introspecting the features and state of a class at runtime and dynamically manipulate at run time. This is supported using Reflection API with builtin classes like Class, Method, Fields, and Constructors etc. Example: Using Java Reflection API we can get the class name, by using the getName method.		BTL3
24	**What is the useful of Interfaces?** a) Declaring methods that one or more classes are expected to implement b) Capturing similarities between unrelated classes without forcing a class relationship. c) Determining an object's programming interface without revealing the actual body of the class.		BTL1
25	**Define Protected members** **Protected** Access Modifier - **Protected**. **Variables**, methods, and constructors, which are declared **protected** in a superclass can be accessed only by the subclasses in other package or any class within the package of the **protected members'** class. The **protected** access modifier cannot be applied to class and interfaces.		BTL1
26	**What is object cloning in java** **clone**() is a method in the **Java** programming language for **object** duplication. In **java**, **objects** are manipulated through reference variables, and there is no operator for copying an **object**—the assignment operator duplicates the reference, not the **object**. The **clone**() method provides this missing functionality.		BTL1

27	**Write short notes on final classes and methods?**
	the final keyword in a method declaration to indicate that the method cannot be overridden by subclasses. The Object class does this—a number of its methods are final.
	class ChessAlgorithm {
	enum ChessPlayer { WHITE, BLACK }
	...
	final ChessPlayer getFirstPlayer() {
	return ChessPlayer.WHITE;
	}
	...
	}
			BTL1

28	Difference between interface and extending interface? Any **interface** that implements a particular **interface** should **implement** all methods defined **in the interface**, or should be declared as an abstract class. In short, Implements keyword **is** used **for** a class to **implement** a certain **interface**, while **Extends** keyword **is** used **for** a subclass to **extend** from a super class.		BTL2
29	List out the types of inheritance in java • Single Inheritance. • Multiple Inheritance (Through Interface) • Multilevel Inheritance. • Hierarchical Inheritance. • Hybrid Inheritance (Through Interface)		BTL1
30	List out the types of Constructor? There are two type of constructor in Java: No-argument constructor: A constructor that has no parameter is known as default constructor. If we **don**'t define a constructor in a class, then compiler creates default constructor(with no arguments) for the class		BTL1

PART-B

Q.No	Questions	CO	Bloom's level
1	**Explain with an example the following features of Constructors:** **(16) MAY/JUNE 2011** **(i) Overloaded constructors** **(ii) A Call to another constructor with this operator** **(iii) An object initialization block** **(iv) A static initialization block** Refer page no: 144		BTL5
2	**How Strings are handled in java? Explain with code, the creation of Substring, Concatenation and testing for equality.(16)** **MAY/JUNE 2011**		BTL5

	Refer page no: 53		
3	**(i) Explain any four string operations in Java. With suitable examples. (8) NOV/DEC 2011** Refer notes **Explain string handling classes in Java with examples. (16)** **MAY/JUNE 2012** Refer page no: 53		BTL5
4	**Write down the techniques to design good classes. (8)** **MAY/JUNE 2013** Refer notes **ii) Explain about java building string functions with an example. (8)** Refer page no: 53		BTL5
5	What is meant by constructor? Describe the types of constructors supported by Java with example? **NOV/DEC 2014** Refer page no: 144		BTL6
6	**(i) Explain Inheritance and class hierarchy. (8)** **NOV/DEC 2010** Refer page no:171 & 179 **(ii) Define Polymorphism. (2)** **NOV/DEC 2010** Refer page no:179 **(iii)Write briefly on Abstract classes with an example. (6)** **NOV/DEC 2010** Refer page no:186		BTL5
7	**(i) Explain the following with examples :** **a. The clone able interface (4)** **NOV/DEC 2010** **b. The property interface. (4)** **NOV/DEC 2010** Refer page no:242 **(ii) What is a static Inner class? Explain with example.(8)** **NOV/DEC 2010** Refer page no:258		BTL5
8	**(i) Explain interfaces with example. (8)** **MAY/JUNE 2011** Refer page no:242 **(ii) Compare interfaces and abstract classes. (8)** **MAY/JUNE 2011** Refer page no:186 & 242		BTL4

9	**(i) Explain interfaces with example. (8)** **MAY/JUNE 2011** Refer page no:242 **(ii) Compare interfaces and abstract classes. (8) MAY/JUNE 2011** Refer page no:186 & 242		BTL4
10	**(i) What is meant by object cloning? Explain it with an example. (8)** **NOV/DEC 2011, NOV/DEC 2014** Refer page no:249 **(ii) Discuss in detail about inner class, with its usefullness. (8) NOV/DEC 2011** Refer page no:258		BTL6

UNIT III

UNIT III EXCEPTION HANDLING AND I/O 9

Exceptions - exception hierarchy - throwing and catching exceptions – built-in exceptions, creating own exceptions, Stack Trace Elements. Input / Output Basics – Streams – Byte streams and Character streams – Reading and Writing Console – Reading and Writing Files

Q.No	Questions	CO	Bloom's Level
1	**What is the use of final keyword?** **MAY/JUNE 2013** The ***final*** keyword is used in several different contexts to define an entity which cannot later be changed. the keyword final is a simple but powerful tool that allows us to write code that is more readable, enables the compiler to catch some logic errors, and prevents accidental misuse of classes and member functions.		BTL1

2	**How will create throw exception in exception handling?** **NOV/DEC 2011**
	Throwing Exceptions
	
	If a method needs to be able to throw an exception, it has to declare the exception(s) thrown in the method signature, and then include a throw-statement in the method. Here is an example:
	public void divide(int numberToDivide, int numberToDivideBy)
	throws BadNumberException{
	if(numberToDivideBy == 0){
	throw new BadNumberException("Cannot divide by 0");
	}
	return numberToDivide / numberToDivideBy;
	}
			BTL1

3	**What is an exception? MAY/JUNE – 2012** Exceptions are such anomalous conditions (or typically an event) which changes the normal flow of execution of a program. Exceptions are used for signaling erroneous (exceptional) conditions which occur during the run time processing.		BTL1

	Exceptions may occur in any programming language Occurrence of any kind of exception in java applications may result in an abrupt termination of the JVM or simply the JVM crashes which leaves the user unaware of the causes of such anomalous conditions. However Java provides mechanisms to handle such situations through its superb exception handling mechanism. The Java programming language uses Exception classes to handle such erroneous conditions and exceptional events.		
4	Mention the different ways to generate an Exception? There are two different ways to generate an Exception. 1. Exceptions can be generated by the Java run-time system. Exceptions thrown by Java relate to fundamental errors that violate the rules of the Java language or the constraints of the Java execution environment. 2. Exceptions can be manually generated by the code. Manually generated exceptions are typically used to report some error condition to the caller of a method.		BTL1

5	**Can we keep other statements in between try, catch and finally blocks?**
	
	No. We shouldn't write any other statements in between try, catch and finally blocks. They form a one unit.
	
	try{ // Statements to be monitored for exceptions
	}
	//You can't keep statements here
	catch(Exception ex)
	{
	//Cathcing the exceptions here
	}
	//You can't keep statements here
	finally
	{
	// This block is always executed
	}
			BTL1

6	**What is Re-throwing an exception in java?** Exceptions raised in the try block are handled in the catch block. If it is unable to handle that exception, it can re-throw that exception using throw keyword. It is called re-throwing an exception.		BTL1
7	**what are checked and unchecked exceptions in java?** Checked exceptions are the exceptions which are known to compiler. These exceptions are checked at compile time only. Hence the name checked		BTL1

	exceptions. These exceptions are also called compile time exceptions. Because, these exceptions will be known during compile time. Unchecked exceptions are those exceptions which are not at all known to compiler. These exceptions occur only at run time. These exceptions are also called as run time exceptions. All sub classes of java.lang.RunTimeException and java.lang.Error are unchecked exceptions.		
8	**What are run time exceptions in java. Give example?** The exceptions which occur at run time are called as run time exceptions. These exceptions are unknown to compiler. All sub classes of java.lang.RunTimeException and java.lang.Error are run time exceptions. These exceptions are unchecked type of exceptions. For example, NumberFormatException, NullPointerException, ClassCastException, ArrayIndexOutOfBoundException, StackOverflowError etc.		BTL1
9	**There are three statements in a try block – statement1, statement2 and statement3. After that there is a catch block to catch the exceptions occurred in the try block. Assume that exception has occurred in statement2. Does statement3 get executed or not?** No. Once a try block throws an exception, remaining statements will not be executed. control comes directly to catch block.		BTL1
10	**Can we write only try block without catch and finally blocks?** No, It shows compilation error. The try block must be followed by either catch or finally block. You can remove either catch block or finally block but not both		BTL1
11	**What are Byte Stream in Java?** The byte stream classes provide a rich environment for handling byte-oriented I/O. List of Byte Stream classes ByteArrayInputStream ByteArrayOutputStream FilteredByteStreams BufferedByteStreams		BTL1
12	**How to read strings of text from a file ?** Open the file for reading by creating an object of the class FileReader and an object of the class BufferedReader associated to the FileReader object we have just created; 2. Read the lines of text from the file by using the readLine() method of the BufferedReader		BTL1

	object; 3.close the file when we are finished reading from it.		
13	**Write a note on char Array Reader** The CharArrayReader allows the usage of a character array as an InputStream. The usage of CharArrayReader class is similar to ByteArrayInputStream. The constructor is given below: public CharArrayReader(char c[ ])		BTL1
14	**What are Character Stream in Java?** The Character Stream classes provide a rich environment for handling character- oriented I/O. List of Character Stream classes FileReader FileWriter CharArrayReader CharArrayWriter		BTL1
15	**What are the different states of a thread? (N/D -10) [2 Marks]** The different states of a thread are New Runnable Waiting TimedWaiting Terminated		BTL2
16	**What is meant by checked and unchecked exception? [2 Marks]** All predefined exceptions in java are either checked exception or an unchecked exception checked exception must be caught using try() and catch() block.		BTL4

17	**How is custom exception created? [2 Marks]**
	By extending the exception class or one of its subclasses
	Example
	Class MyException extends Exception
	{
	Public MyException(){super();}
	Public MyException(String s){super(s);}
	}
			BTL1

18	**Mention the stream I/O operations.** Stream I/O operations involve three steps: 1. *Open* an input/output stream associated with a physical device (e.g., file, network, console/keyboard), by constructing an appropriate I/O stream instance. 2. *Read* from the opened input stream until "end-of-stream" encountered, or *write* to the opened output stream (and optionally flush the buffered output). 3. *Close* the input/output stream.		BTL1

19	**What is stream?** A *stream* is a sequential and contiguous one-way flow of data.Java does not differentiate between the various types of data sources or sinks (e.g., file or network) in stream I/O.		BTL1
20	**Mention the different ways to generate an Exception?** There are two different ways to generate an Exception. 1. Exceptions can be generated by the Java run-time system. Exceptions thrown by Java relate to fundamental errors that violate the rules of the Java language or the constraints of the Java execution environment. 2. Exceptions can be manually generated by the code. Manually generated exceptions are typically used to report some error condition to the caller of a method.		BTL1
21	**What is OutOfMemoryError in java?** OutOfMemoryError is the sub class of java.lang.Error which occurs when JVM runs out of memory.		BTL1
22	**Give some examples to checked exceptions?** ClassNotFoundException, SQLException, IOException		BTL1
23	**Give some examples to unchecked exceptions?** NullPointerException, ArrayIndexOutOfBoundsException, NumberFormatException		BTL1
24	**What are FileInputStream and FileOutputStream?** These two are general purpose classes used by the programmer very often to copy file to file. These classes work well with files containing less data of a few thousand bytes as by performance these are very poor. For larger data, it is preferred to use BufferedInputStream (or BufferedReader) and BufferedOutputStream (or BufferedWriter)		BTL1
25	**What is File class?** It is a non-stream (not used for file operations) class used to know the properties of a file like when it was created (or modified), has read and write permissions, size etc.		BTL1
26	**What is a IO stream?** It is a stream of data that flows from source to destination. Good example is file copying. Two streams are involved – input stream and output stream. An input stream reads from the file and stores the data in the process (generally in a temporary variable). The output stream reads from the process and writes to the destination file.		BTL1
27	Define stack trace elements the **stack trace** is encapsulated into an array of a **java** class called **java**.lang.**StackTraceElement** . The **stack trace element** array returned by Throwable.getStackTrace() method. Each **element** represents a single **stack** frame.		BTL1

28	**What is the difference between error and exception in java?** Errors are mainly caused by the environment in which an application is running. For example, OutOfMemoryError happens when JVM runs out of memory. Where as exceptions are mainly caused by the application itself. For example, NullPointerException occurs when an application tries to access null object.		BTL1

29	Give example for built in exception ?
	
	Arithmetic exception : It is thrown when an exceptional condition has occurred in an arithmetic operation.
	
	`// Java program to demonstrate`
	`// ArithmeticException`
	`class ArithmeticException_Demo {`
	`public static void main(String args[])`
	`{`
	`try {`
	`int a = 30, b = 0;`
	`int c = a / b; // cannot divide by zero`
	`System.out.println("Result = " + c);`
	`}`
	`catch (ArithmeticException e) {`
	`System.out.println("Can't divide a number by 0");`
	`}`
	`}`
	`}`
			BTL1

30	**Give any two methods available in Stack trace Element** **<u>NOV/DEC 2010</u>.** public final class **StackTraceElement** extends Object implements Serializable An element in a stack trace, as returned by Throwable.getStackTrace(). Each element represents a single stack frame. All stack frames except for the one at the top of the stack represent a method invocation. The frame at the top of the stack represents the execution point at which the stack trace was generated. Typically, this is the point at which the throwable corresponding to the stack trace was created.		BTL1

PART-B

Q.No	Questions	CO	Bloom's level
1	**(ii) Explain the exception hierarchy.** **(8) NOV/DEC 2010** Refer page no:553 **(i) Explain the concept of throwing and catching exception in java.** **(8) NOV/DEC 2010** Refer page no:557 - 559		BTL5
2	**(i) Explain the concept of throwing and catching exception in java.** **(8) NOV/DEC 2010** Refer page no:557 - 559 **(ii) State the situations where assertions are not used.** **(4) NOV/DEC 2010, MAY/JUNE 2014** Refer page no:571 **(iii) Give an overview of logging.** **(4) NOV/DEC 2010** Refer page no:575		BTL5
3	**Explain briefly about user defined exceptions and stack trace elements in exception handling mechanisms. (16)**		BTL5
4	**Discuss on Exception handling in detail. (16)** **MAY/JUNE 2012, MAY/JUNE 2014** Refer page no:559		BTL6
5	**What is meant by exceptions? Why it is needed?Describe the exception hierarchy. Write note on Stack Trace Elements. Give example.** **(16) NOV/DEC 2013**		BTL6
6	**I)Define exception and explain its different types with example (8) APR/MAY 2015** Refer page no:553 **What is final keyword? Explain with an example? (8) APR/MAY 2015**		BTL5

	Refer page no:559		

UNIT IV

UNIT IV MULTITHREADING AND GENERIC PROGRAMMING 8

Differences between multi-threading and multitasking, thread life cycle, creating threads, synchronizing threads, Inter-thread communication, daemon threads, thread groups. Generic Programming – Generic classes – generic methods – Bounded Types – Restrictions and Limitations.

Q.No	Questions	CO	Bloom's Level
1	**What are the different states in thread?** **NOV/DEC 2010** ● **New:** A new thread begins its life cycle in the new state. It remains in this state until the program starts the thread. It is also referred to as a born thread. ● **Runnable:** After a newly born thread is started, the thread becomes runnable. A thread in this state is considered to be executing its task. ● **Waiting:** Sometimes a thread transitions to the waiting state while the		BTL1

	thread waits for another thread to perform a task.A thread transitions back to the runnable state only when another thread signals the waiting thread to continue executing. ● **Timed waiting:** A runnable thread can enter the timed waiting state for a specified interval of time. A thread in this state transitions back to the runnable state when that time interval expires or when the event it is waiting for occurs. ● **Terminated:** A runnable thread enters the terminated state when it completes its task or otherwise terminates.		

2	**What do you mean Synchronization? <u>NOV/DEC 2010 , MAY/JUNE 2012</u>**
	When two or more threads need access to a shared resource, they need some way to ensure that the resource will be used by only one thread at a time.
	
	The process by which this synchronization is achieved is called *thread synchronization*. The synchronized keyword in Java creates a block of code referred to as a critical section.
	
	Every Java object with a critical section of code gets a lock associated with the object. To
	
	enter a critical section, a thread needs to obtain the corresponding
	object's lock. This is the general form of the synchronized statement:
	
	synchronized(object) {
	
	// statements to be synchronized
	
	}
			BTL1

3	**Mention the two mechanisms for protecting a code block from concurrent access. <u>MAY/JUNE 2011</u>** Java 5 introduces general purpose synchronizer classes, including **semaphores, mutexes, barriers, latches, and exchangers,** which facilitate coordination between threads. These classes are a apart of the java.util.concurrent package		BTL1
4	**What do you mean by Threads in Java? <u>NOV/DEC 2011</u>** A *thread* is a program's path of execution. Most programs written today run as a single thread, causing problems when multiple events or actions need to occur at the		BTL1

	same time. Let's say, for example, a program is not capable of drawing pictures while reading keystrokes. The program must give its full attention to the keyboard input lacking the ability to handle more than one event at a time. The ideal solution to this problem is the seamless execution of two or more sections of a program at the same time.		
5	**What is multithreading. NOV/DEC 2011 , MAY/JUNE – 2012, MAY/JUNE 2013, NOV/DEC 2014** Multithreading enables us to write very efficient programs that make maximum use of the CPU, because idle time can be kept to a minimum. The Java Virtual Machine allows an application to have multiple threads of execution running concurrently Multithreaded applications deliver their potent power by running many threads concurrently within a single program. From a logical point of view, multithreading means multiple lines of a single program can be executed at the same time,		BTL1
6	**What is meant by notify methods in multithreading? NOV/DEC 2011** Used for Inter thread Communication: To avoid polling, Java includes an elegant interprocess communication mechanism via the following methods: - wait( ): This method tells the calling thread to give up the monitor and go to sleep until some other thread enters the same monitor and calls notify( ). - notify( ): This method wakes up the first thread that called wait( ) on the same object. - notifyAll( ): This method wakes up all the threads that called wait( ) on the same object.c The highest priority thread will run first.		BTL1
7	**Name any four thread constructors.** **MAY/JUNE 2013** 1) public Thread() 2) public Thread(String name) 3) Thread t1=new Thread(); 4) Thread t2=newThread("MYTHREAD");		BTL1
8	**What is the need for threads?** **MAY/JUNE 2013** Need of threads: thread priorities, daemon threads, thread groups, and handlers for uncaught exceptions.		BTL1

9	**How will interrupt threads in multiple windows?** **NOV/DEC 2011** An *interrupt* is an indication to a thread that it should stop what it is doing and do something else. It's up to the programmer to decide exactly how a thread responds to an interrupt, but it is very common for the thread to terminate. This is the usage emphasized in this lesson. A thread sends an interrupt by invoking interrupt on the Thread object for the thread to be interrupted. For the interrupt mechanism to work correctly, the interrupted thread must support its own interruption.		BTL1
10	**Write the life cycle of thread.** **NOV/DEC 2013** 1. Newborn state. 2. Runnable state. 3. Running state. 4. Blocked state. 5. Dead state.		BTL1
11	**What is meant by event-driven programming?** **NOV/DEC 2013, MAY/JUNE 2014 NOV/DEC 2014** **Event-driven programming** is a programming paradigm in which the flow of the program is determined by events such as user actions (mouse clicks, key presses), sensor outputs, or messages from other programs/threads. Event-driven programming is the dominant paradigm used in graphical user interfaces and other applications (e.g. JavaScript web applications) that are centered around performing certain actions in response to user input		BTL1
12	**What are the properties of a thread ? NOV/DEC 2014** runnable: the object whose run() method is run on the thread name: the name of the thread (used mainly for logging or other diagnostics) id: the thread's id (a unique, positive long generated by the system when the thread was created) threadGroup: the group to which this thread belongs daemon: the thread's daemon status. A daemon thread is one that performs services for other threads, or periodically runs some task, and is not expected to run to completion. contextClassLoader: the classloader used by the thread priority: a small integer between Thread.MIN_PRIORITY and Thread.MAX_PRIORITY inclusive		BTL1

	state: the current state of the thread interrupted: the thread's interruption status		
13	**List out the motivation needed in generic programming. (N/D- 11) [2 Marks]** The motivation needed in generic programming are : 🎬 To motivate generic methods, that contains three overloaded print Array methods 🎬 This methods print the string representations of the elements of an integer array, double array and character array 🎬 Choose to use arrays of type Integer, double and character to setup our generic methods		BTL1
14	**How is throw exception created in exception handling? (N/D- 11) [2 Marks]** Throw method is used to throw an exception manually. It has to declare the exceptions thrown in the method signature and then include a throw statement in the method throw new ExceptionName ("Value");		BTL1
15	**List out the advantages of multithreading.** The advantages are as follows · Can be created faster · Requires less overheads · Inter-process communication is faster · Context switching is faster · Maximum use of CPU time		BTL1
16	How Generics works in Java ? What is type erasure ? This is one of better interview question in Generics. Generics is implemented using Type erasure, compiler erases all type related information during compile time and no type related information is available during runtime. for example List<String> is represented by only List at runtime. This was done to ensure binary compatibility with the libraries which were developed prior to Java 5. you don't have access to Type argument at runtime and Generic type is translated to Raw type by compiler during runtime.		BTL1
17	How to write a generic method which accepts generic argument and return Generic Type? writing generic method is not difficult, instead of using raw type you need to use Generic Type like T, E or K,V which are well known placeholders for		BTL1

	Type, Element and Key, Value. Look on Java Collection framework for examples of generics methods. In simplest form a generic method would look like:
	
	`public V put(K key, V value) {`
	`  return cache.put(key, value);`
	`}`		

18	Can we use Generics with Array? This was probably most simple generics interview question in Java, if you know the fact that Array doesn't support Generics and that's why Joshua Bloch suggested in Effective Java to prefer List over Array because List can provide compile time type-safety over Array.		BTL1
19	**List out the Advantage of Java Generics** There are mainly 3 advantages of generics. They are as follows: **1) Type-safety :** We can hold only a single type of objects in generics. It doesn't allow to store other objects. **2) Type casting is not required:** There is no need to typecast the object		BTL2
20	**What is meant by multithreading? (M/J- 12) [2 Marks]** Multithreading extends the idea of multitasking into applications where you can subdivide specific operations within a single application into individual threads. Each of the threads can run in parallel. The OS divides processing time not only among different applications, but also among each thread within an application.		BTL1
21	**What are the different identifier states of a thread? [2 Marks]** The different identifiers of a thread are: R - Running or runnable thread S - Suspended threads CW - Thread waiting on a condition variable MW - Thread waiting on a monitor lock MS - Thread suspended waiting on a monitor lock		BTL1
22	**What method must be implemented by all threads? [2 Marks]** All tasks must implement the run() method, whether they are a subclass of thread or implement the runnable interface.		BTL1
23	**Define Multitasking** Multitasking is a process of executing multiple tasks simultaneously. We use multitasking to utilize the CPU. Multitasking can be achieved by two ways: • Process-based Multitasking(Multiprocessing) • Thread-based Multitasking(Multithreading)		BTL1
24	**Define Inter-thread communication?** **Inter-thread communication** or **Co-operation** is all about allowing synchronized threads to communicate with each other. Cooperation (Inter-thread communication) is a mechanism in which a thread is paused running in its critical section and another thread is allowed to enter		BTL1

	(or lock) in the same critical section to be executed.It is implemented by following methods of **Object class**: • wait() • notify() • notifyAll()		
25	**Difference between wait and sleep?** Let's see the important differences between wait and sleep methods. **wait()** — **sleep()** wait() method releases the lock — sleep() method doesn't release the is the method of Object class — is the method of Thread class is the non-static method — is the static method is the non-static method — is the static method should be notified by notify() or notifyAll() methods — after the specified amount of time completed.		BTL2
26	What about isInterrupted and interrupted method? The isInterrupted() method returns the interrupted flag either true or false. The s method returns the interrupted flag afterthat it sets the flag to false if it is true.		BTL1
27	What if we call run() method directly instead start() method? • Each thread starts in a separate call stack. • Invoking the run() method from main thread, the run() method goes onto stack rather than at the beginning of a new call stack.		BTL1
28	Define daemon thread? A **daemon thread** is a **thread** that does not prevent the JVM from exiting when the program finishes but the **thread** is still running. An example for a **daemon thread** is the garbage collection. You can use the setDaemon(boolean) method to change the **Thread daemon** properties before the **thread** starts.		BTL1
29	**List out the different Methods for Java Daemon thread by Thread class** The java.lang.Thread class provides two methods for java daemon thread.		BTL1

	No. Method **Description** 1) public void setDaemon(boolean status) is used to mark the current thread as daem thread. 2) public boolean isDaemon() is used to check that current is daemon.		

30	**How to perform single task by multiple threads?**
	If you have to perform single task by many threads, have only one run() method.
	For example:
	
	1. class TestMultitasking1 extends Thread{
	2. public void run(){
	3. System.out.println("task one");
	4. }
	5. public static void main(String args[]){
	6. TestMultitasking1 t1=new TestMultitasking1();
	7. TestMultitasking1 t2=new TestMultitasking1();
	8. TestMultitasking1 t3=new TestMultitasking1();
	9.
	10. t1.start();
	11. t2.start();
	12. t3.start();
	13. }
	14. }
			BTL1

PART-B

Q.No	Questions	CO	Bloom's level
1	**(i) What is a thread? Explain its states and methods. (8) NOV/DEC 2010** Refer page no:716 **(ii) Explain thread properties. (8) NOV/DEC 2010**		BTL5

	Refer page no:733		
2	**(i) Explain the methods of interrupting threads in java. (8)** **NOV/DEC 2010** **(ii) What is Event-driven programming? Explain. (8)** **NOV/DEC 2010, MAY/JUNE 2014** Refer page no:728		BTL5
3	**Explain the procedure for running a task in a separate thread and running multiple threads. (16)** **MAY/JUNE 2011** Refer page no:731		BTL5
4	**Explain the states of threads and how the thread is interrupted?** **MAY/JUNE 201, MAY/JUNE 2014, NOV/DEC 2014** Refer page no:728		BTL5
5	**(i) Explain how threads are created in Java. (8)** **NOV/DEC 2011** **(ii) Write about various threads states in Java. (8)** **NOV/DEC 2011** Refer page no:730		BTL5
6	**Explain briefly about thread synchronization in multithreading concepts. (16)** **NOV/DEC 2011**		BTL5
7	**Explain the following (i) States of a thread with a neat diagram. [10]** **MAY/JUNE 2012** **(ii) Thread priorities. (6)** **MAY/JUNE 2012** Refer page no:716 & 733		BTL5
8	**i) How to implement runnable interface for creating and starting threads?(8)** **MAY/JUNE 2013** **ii) Define threads. Describe in detail about thread life cycle. (8)**		BTL5

UNIT V

UNIT V EVENT DRIVEN PROGRAMMING 9

Graphics programming - Frame – Components - working with 2D shapes - Using color, fonts, and images - Basics of event handling - event handlers - adapter classes - actions - mouse events - AWT event hierarchy - Introduction to Swing – layout management - Swing

Components – Text Fields , Text Areas – Buttons- Check Boxes – Radio Buttons – Lists- choices- Scrollbars – Windows –Menus – Dialog Boxes.

Q.No	Questions	CO	Bloom's Level
1	**What is virtual machine in generic programming?** • the JVM has no instantiations of generic types • a generic type definition is compiled once only, and a corresponding *raw type* is produced – the name of the raw type is the same name but type variables removed		BTL1
2	**Define generic class** A generic class declaration looks like a non-generic class declaration, except that the class name is followed by a type parameter section. As with generic methods, the type parameter section of a generic class can have one or more type parameters separated by commas. These classes are known as parameterized classes or parameterized types because they accept one or more parameters.		BTL1
3	**Define the term virtual machine.** **NOV/DEC2013** The fundamental idea behind a virtual machine is to abstract the hardware of a single computer (the CPU, memory, disk drives, network interface cards, and so forth) into several different execution environments, thereby creating the illusion that each separate execution environment is running its own private computer.		BTL1
4	**What is meant by generic classeses? Give example.** **NOV/DEC 2013** A generic class is a class with one or more type variables. This class allows to focus on generics without being distracted by data storage details. **Ex:** public class Pair<T> { public Pair() { first = null; second = null; } public Pair (T first, T second)		BTL1

	{ this.first = first; this.second = second; }		
5	**What is the need for generic programming? MAY/JUNE 2013** Generic programming means to write code that can be reused for objects of many different types. The single class **ArrayList** collects objects of any class.		BTL1
6	**Write about simple generic class with an example.** **NOV/DEC 2011** Java **Generic** methods and generic classes enable programmers to specify, with a single method declaration, a set of related methods or, with a single class declaration, a set of related types, respectively. Generics also provide compile-time type safety that allows programmers to catch invalid types at compile time. Using Java Generic concept we might write a generic method for sorting an array of objects, then invoke the generic method with Integer arrays, Double arrays, String arrays and so on, to sort the array elements.		BTL1
7	**What are Wildcard Types? MAY/JUNE 2011** The **wildcard** ? in Java is a special actual parameter for the instantiation of generic (parameterized) types. It can be used for the instantiation, not in the definition of a generic unit		BTL1
8	**Why generic programming is required?** **MAY/JUNE 2011** Important goal in implementing generic programming in Java is to respect the spirit of the language and not make it conform to the way generic programming is implemented in other languages. Three notions essential to the practice of programming in Java are: Using OO techniques for system design Using compile time checks to catch obvious errors and allowing the runtime system to catch the rest Using reflection to create flexible system architectures		BTL1
9	**State the features of Swing** **NOV/DEC 2013** Swing is the next-generation GUI toolkit that Sun Microsystems has developed for the Java language. It is essentially a vast component framework built over parts of the older AWT component libraries used in Java 1.0 and 1.1.		BTL1

10	**What are the steps needed to show a Frame?** **NOV/DEC 2010** *First Step: JFrame* *Second Step: WindowListener* *Third Step: Adding a Panel* *Fourth Step: Fonts in Panels* *Fifth Step: Basic Graphics* *6th Step: Basic Event Handling* *7th Step: Window Events* *8th Step: Event Classes and Listener Interfaces* *9th Step: Focus Event* *10th Step: KeyBoard Events and Sketch Demo* *11th Step: Mouse Events and Mouse Demo* *12th Step: Action Interface*		BTL1
11	**What is adapter class?** **MAY/JUNE 2013** An adapter class provides the default implementation of all methods in an event listener interface. Adapter classes are very useful to process only few of the events that are handled by a particular event listener interface. Define a new class by extending one of the adapter classes and implement only those events relevant.		BTL1
12	**Define frame.** **MAY/JUNE 2013** In graphics and desktop publishing applications, a rectangular area in which text or graphics can appear. In video and animation, a single image in a sequence of images. In HTML, refers to dividing the browser display area into separate sections, each of which is really a different Web page.		BTL1
13	**Mention any four event names of a button component.** **MAY/JUNE 2012** **getAccessibleContext()** Gets the AccessibleContext associated with this JButton. **getUIClassID()** Returns a string that specifies the name of the L&F class that renders this component. **isDefaultButton()** Gets the value of the defaultButton property, which if true means that this button is the current default button for its JRootPane.		BTL1

	isDefaultCapable() Gets the value of the defaultCapable property. **paramString**() Returns a string representation of this JButton.		
14	**How do you manage the color and font of a graphics in applet?** **MAY/JUNE 2012** To draw an object in a particular color, you must create an instance of the Color class to represent that color. The Color class defines a set of standard color objects, stored in class variables, to quickly get a color object for some of the more popular colors. For example, Color.red returns a Color object representing red (RGB values of 255, 0, and 0), Color.white returns a white color (RGB values of 255, 255, and 255)		BTL1
15	**What is meant by window adapter classes? NOV/DEC 2011,NOV/DEC 2014** The adapter which receives window events. The methods in this class are empty; this class is provided as a convenience for easily creating listeners by extending this class and overriding only the methods of interest. public abstract class **WindowAdapter** extends Object implements WindowListener		BTL1
16	**Differentiate GridBagLayout from GridLayout NOV/DEC 2011.** *GridLayout* class lays all components in a rectangular grid like structure of container. The container is divided into an equal sized rectangles and each component is placed inside a rectangle. The *GridBagLayout* class is a flexible layout manager that aligns components vertically and horizontally, without requiring that the components be of the same size. Each *GridBagLayout* object maintains a dynamic, rectangular grid of cells, with each component occupying one or more cells, called its display area.		BTL2
17	**How are frames created in Java? NOV/DEC 2011** A Frame is a top-level window with a title and a border. The size of the frame includes any area designated for the border. The dimensions of the border area may be obtained using the getInsets method. Since the border area is included in the overall size of the frame, the border effectively obscures a portion of the frame, constraining the area available for rendering and/or displaying subcomponents to the rectangle which has an upper-left corner location		BTL1

	of (insets.left, insets.top), and has a size of width - (insets.left + insets.right) by height - (insets.top + insets.bottom). A frame, implemented as an instance of the JFrame class, is a window that has decorations such as a border, a title, and supports button components that close or iconify the window. Applications with a GUI usually include at least one frame		
18	**Give the value for the following predefined actions. NOV/DEC 2011** **(a) SMALL-ICON -** Place to store small icon, The icon for the action used in the tool bar or on a button. You can set this property when creating the action using the AbstractAction(name, icon) constructor. **(b) MNEMONIC-KEY -** The mnemonic abbreviations, for display in menu items.		BTL1
19	**List the AWT controls?** Label Button Checkbox TextComponent Choice List Scrollbar		BTL4
20	**What method is used to distinguish between single, double and triple mouse clicks? MAY/JUNE – 2011** public void mouseClicked(MouseEvent e)		BTL1
21	**What is a JPanel object?** **MAY/JUNE 2011** JPanel is a generic lightweight container. For examples and task-oriented documentation for JPanel, **All Implemented Interfaces:** ImageObserver, MenuContainer, Serializable, Accessible **Direct Known Subclasses:** AbstractColorChooserPanel, JSpinner.DefaultEditor		BTL1
22	**What is the function of** **NOV/DEC 2010** a. **Set Layout** - The **SetLayout** function changes the layout of a device context (DC). b. **Flow Layout**		BTL1

23	**Write a note on push Button Control?** A push button is a component that contains a label and that generates as event when it is pressed. Push buttons are objects of type Button. Button defines these two constructors. Button( ) Button(String str) The first version creates an empty button. The second creates a button that contains str as a label.		BTL1
24	**Distinguish between component and container** Component is an abstract class that encapsulates all of the attributes of a visual component. All user interface elements that are displayed on the screen and that interact with the user are subclasses of Component. The container class is a subclass of Component. It has additional methods that allow other Component objects to be nested within it. Other Container objects can be stored inside of a container.		BTL2
25	**Write a note on Borderlayout?** The BorderLayout class implements a common layout style for top-level windows. It has four narrow, fixed-width components at the edges and one large area in the center. The four sides are referred to as north, south, east, and west. The middles area is called the center. Here are the constructors defined by BorderLayout BorderLayout( ) BorderLayout(int horz, int vert)		BTL1
26	What is the difference between the 'Font' and 'FontMetrics' class? The Font Class is used to render 'glyphs' - the characters you see on the screen. FontMetrics encapsulates information about a specific font on a specific Graphics object. (width of the characters, ascent, descent)		BTL2
27	How would you create a button with rounded edges? There's 2 ways. The first thing is to know that a JButton's edges are drawn by a Border. so you can override the Button's paintComponent(Graphics) method and draw a circle or rounded rectangle (whatever), and turn off the border. Or you can create a custom border that draws a circle or rounded rectangle around any component and set the button's border to it.		BTL1

28	Difference between Swing and Awt? AWT are heavy-weight componenets. Swings are light-weight components. Hence swing works faster than AWT.		BTL2
29	What is an event and what are the models available for event handling? An event is an event object that describes a state of change in a source. In other words, event occurs when an action is generated, like pressing button, clicking mouse, selecting a list, etc. There are two types of models for handling events and they are: a) event-inheritance model and b) event-delegation model		BTL1
30	What is the difference between scrollbar and scrollpane? A Scrollbar is a Component, but not a Container whereas Scrollpane is a Conatiner and handles its own events and perform its own scrolling.		BTL1

PART-B

Q.No	Questions	CO	Bloom's level
1	**Explain about the concepts of creating and positioning of frame (8) APR/MAY 2015**		BTL5
2	**Define Event handling write a program to handle a button event(8) APR/MAY 2015**		BTL5
3	**Explain the classes under 2D shapes. Working with 2D Shapes**		BTL5
4	**Explain event handling with examples. EVENT HANDLING**		BTL5
5	**n action event with an example. Actions**		BTL5
6	**are the swing components? Explain. Swing Component**		BTL5
7	**Describe the AWT event hierarchy. The AWT Event Hierarchy**		BTL6

8	**Give the methods available in graphics for COLOR and FONTS Color and fonts Using Color**		BTL5