GUJARAT TECHNOLOGICAL UNIVERSITY

Program Name: Master of Computer Applications
Level: Post Graduate
Course / Subject Code : MC01094041
Course / Subject Name : Object Oriented Programming using JAVA

w. e. f. Academic Year:	2024-25
Semester:	1
Category of the Course:	Core Course

Prerequisite:	Knowledge of the C programming language and Object Oriented Concept
Rationale:	• To develop proficiency in creating console based applications using the Java Programming Language. • To interpret the concepts of object oriented Programming Language and easily use these concepts in Java program. • To develop application using java.util package (Java data structure). • To implement multi-threaded applications using the Java Programming Language. • To implement application including different file operations.

Course Outcome:

After completion of the course, student will able to:

No.	Course Outcomes	RBT Level*
CO1	Describe Java features, OOP concepts and Write console java program syntax and semantics using datatypes, variables, control and looping statements.	RM, UN, AP
CO2	Explain and Implement class, object, encapsulation and polymorphism in JAVA.	RM, UN, AP
CO3	Apply the concept of inheritance, package and exception handling for reusable, structured and error free programs.	RM, UN, AP
CO4	Use wrapper class and Collection interface to convert and manage list of data.	RM, UN, AP
CO5	Describe and Implement multithreaded programs, Generic class and IO programs.	

*RM: Remember, UN: Understand, AP: Apply, AN: Analyze, EL: Evaluate, CR: Create

Teaching and Examination Scheme:

Teaching Scheme (in Hours)			Total Credits L+T+ (PR/2)	Assessment Pattern and Marks				Total Marks
				Theory		Tutorial / Practical		
L	**T**	**PR**	**C**	ESE (E)	PA / CA (M)	PA/CA (I)	ESE (V)	
3	0	2	4	70	30	20	30	150

Course Content:

Unit No.	Content	No. of Hours	% of Weightage
1.	**Introduction to Java** **Java Fundamentals**: The Origins of Java, Java's Lineage: C and C++, How Java Impacted the Internet, Java's Magic: The Bytecode, Moving Beyond Applets, The Java Buzzwords, Object- Oriented Programming (Encapsulation, Polymorphism, Inheritance),Obtaining the Java Development Kit, A First Simple Program, Handling Syntax Errors A Second Simple Program, Another Data Type, Two Control Statements (The if Statement and The for Loop), Create Blocks of Code, Semicolons and Positioning, Indentation Practices, The Java Keywords, Identifiers in Java, The Java Class Libraries Try This 1-1: Converting Gallons to Liters, Try This 1-2: Improving the Gallons-to-Liters Converter **Introducing Data Types and Operators:** Why Data Types Are Important, Java's Primitive Types(Integers, Floating-Point Types, Characters), The Boolean Type, Literals (Hexadecimal, Octal, and Binary Literals, Character Escape Sequences, String Literals), A Closer Look at Variables(Initializing a Variable, Dynamic Initialization), The Scope and Lifetime of Variables, Operators, Arithmetic Operators (Increment and Decrement), Relational and Logical Operators, Short Circuit Logical Operators, The Assignment Operator, Shorthand Assignments, Type Conversion in Assignments, Casting Incompatible Types, Operator Precedence, Expressions (Type Conversion in Expressions, Spacing and Parentheses) Try This 2-1: How Far Away Is the Lightning? Try This 2-2: Display a Truth Table for the Logical Operators **Program Control Statements:** Input Characters from theKeyboard, The if Statement, Nested ifs, The if-else-if Ladder, The switch Statement, Nested switch Statements, The for Loop, Some Variations on the for Loop, Missing Pieces (The Infinite Loop), Loops with No Body, Declaring Loop Control Variables Inside the for Loop, The Enhanced for Loop, The while Loop, The do-while Loop, Usebreak to Exit a Loop, Use break as a Form of goto, Use continue, Nested Loops, Try This 3-1: Start Building a Java Help System,	7	15%

	Try This 3-2: Improve the Java Help System, Try This 3-3: Finish the Java Help System,		
2	**Class Fundamentals** **Introducing Classes, Objects, and Methods:** Class Fundamentals (The General Form of a Class, Defining a Class), How Objects Are Created, Reference Variables and Assignment, Methods (Adding a Method to the Vehicle Class), Returning from a Method, Returning a Value, Using Parameters (Adding a Parameterized Method to Vehicle), Constructors, Parameterized Constructors, Adding a Constructor to the Vehicle Class, The new Operator Revisited, Garbage Collection, The this Keyword, Try This 4-1: Creating a Help Class **A Closer Look at Methods and Classes:** Controlling Access to Class Members (Java's Access Modifiers), Pass Objects toMethods (How Arguments Are Passed), Returning Objects, Method Overloading, Overloading Constructors, Recursion, Understanding static (Static Blocks), Introducing Nested and Inner Classes, Varargs: Variable-Length Arguments (Varargs Basics, OverloadingVarargs Methods, Varargs and Ambiguity) Try This 6-1: Improving the Queue Class, Try This 6-2: Overloading the Queue Constructor, Try This 6-3: The Quicksort	**10**	**25%**
3	**Inheritance Basics** **Inheritance:** Inheritance Basics, Member Access and Inheritance, Constructors and Inheritance, Using super to Call Superclass Constructors, Using super to Access Superclass Members, Creatinga Multilevel Hierarchy, When Are Constructors Executed?, Superclass References and Subclass Objects, Method Overriding, Overridden Methods Support Polymorphism, Why Overridden Methods? (Applying Method Overriding to TwoDShape), Using Abstract Classes, Using final, (final Prevents Overriding, final Prevents Inheritance, Using final with Data Members), The Object Class Try This 7-1: Extending the Vehicle Class **Packages:** Packages (Defining a Package, Finding Packages and CLASSPATH, A Short Package Example), Packages and Member Access (A Package Access Example), Understanding Protected	**10**	**25%**

	Members, Importing Packages, Java's Class Library Is Containedin Packages **Interfaces:** Interfaces, Implementing Interfaces, Using Interface References, Variables in Interfaces, Interfaces Can Be Extended, Default Interface Methods (Default Method Fundamentals, A More Practical Example of a Default Method, Multiple Inheritance Issues), Use static Methods in an Interface, Private Interface Methods, Final Thoughts on Packages and Interfaces, Try This 8-1: Creating a Queue Interface **Exception Handling:** The Exception Hierarchy, Exception Handling Fundamentals (Using try and catch, A Simple Exception Example), The Consequences of an Uncaught Exception(Exceptions Enable You to Handle Errors Gracefully), Using Multiple catch Statements, Catching Subclass Exceptions, Try Blocks Can Be Nested, Throwing an Exception (Rethrowing an Exception), A Closer Look at Throwable, Using finally, Using throws, Three Additional Exception Features, Java's Built-in Exceptions, Creating Exception Subclasses Try This 9-1: Adding Exceptions to the Queue Class		
4	**Autoboxing, java.lang package** Autoboxing, Type Wrappers, Autoboxing Fundamentals, Autoboxing and Methods, Autoboxing/Unboxing Occurs in Expressions **Java.lang package (String, String Buffer, Comparable interface)** **The collection Framework:** Introduction, Collection framework (Collection interface, list interface, set interface, sorted set interface), The collection class, Array list and Link list classes (maintaining the capacity and the link list class), iterating elements of collection (the list iterator interface), hash set and tree setclasses	**8**	**15%**
5	**Multi-Threading, Generic, I/O** **Multithreaded Programming** Multithreading Fundamentals, The Thread Class and Runnable Interface, Creating a Thread, (One Improvement and Two Simple Variations), Creating Multiple Threads, Determining When a Thread Ends, Thread Priorities, Synchronization, Using Synchronized Methods, The synchronized Statement Try This 11-1: Extending Thread,	**10**	**20%**

GUJARAT TECHNOLOGICAL UNIVERSITY

Program Name: Master of Computer Applications
Level: Post Graduate
Course / Subject Code : MC01094041
Course / Subject Name : Object Oriented Programming using JAVA

	Try This 11-2: Using the Main Thread, **Generic:** Generics Fundamentals, A Simple Generics Example, A Generic Class with Two Type Parameters, The General Form of a Generic Class), **Using I/O:** Java's I/O Is Built upon Streams, Byte Streams and Character Streams, The Byte Stream Classes, The Character Stream Classes, The Predefined Streams, Using the Byte Streams (Reading Console Input, Writing Console Output), Reading and Writing Files Using Byte Streams (Inputting from a File, Writing to a File), Automatically Closing a File, Reading and Writing Binary Data		
	Total	**45**	**100**

Suggested Specification Table with Marks (Theory):

Distribution of Theory Marks (in %)					
R Level	**U Level**	**A Level**	**N Level**	**E Level**	**C Level**
10	**20**	**70**	**-**	**-**	**-**

Where R: Remember; U: Understanding; A: Application, N: Analyze and E: Evaluate C: Create (as per Revised Bloom's Taxonomy)

Textbook:

1. Herbert Schildt, Java™: A Beginner's Guide, 7th Edition
2. Hari Mohan Pandey, JAVA Programming, 978-8131733110, Pearson

Reference Books:

1. Pravin Jain, "The class of Java" Pearson Education, (2010).
2. Paul J. Deitel, HarveyM. Deitel, Java SE8 for Programmers, ISBN: 9789332539068, Pearson
3. Cay S Horstmann, Gary Cornell, "Core Java 2, Volume 1 – Fundamentals",Pearson Education (8th edition – 2008).
4. Ivor Horton's "Beginning Java 2" JDK 5 Edition, Wiley Computer Publishing,(2007).
5. Ken Arnold, James Gosling, David Holmes, "The Java Programming Language",Addison-Wesley Pearson Education (4th Edition – 2005).
6. Raj Kumar Buyya, S. Thamarai Selvi, & Xing Chen Chu, "Object-OrientedProgramming with Java: Essentials & Applications", Tata McGraw Hill
7. Cay Horstmann, "Big Java", Wiley Computer publishing (2nd edition – 2006).
8. Sharan Zakhour, Scott Hommel, Jacob Royal, Isaac Rabinovitch, Tom Risser, Mark Hoeber "The Java Tutorial", Addison-Wesley Pearson Education(4th Edition),

9. James Gosling, Bill Joy, Guy Steele, Gilad Bracha, "The Java Langauge

Suggested Course Practical List:

1	Install the JDK (Download the JDK and install it.) • Set path of the jdk/bin directory. • Create the java program • Compile and run the java program Write a simple "Hello World" java program, compilation, debugging,executing using java compiler and interpreter.
2	Write a program to pass Starting and Ending limit and print all prime numbers and Fibonacci numbers between this ranges.
3	Write a java program to check whethernumber is palindrome or not. Input: 528 Output: It is not palindromenumber Input: 545 Output: It is not palindrome number
4	Write a java program to print value ofx^n. Input: x=5 Input: n=3 Output: 125
5	Write a java program to check Armstrong number. Input: 153 Output:Armstrong number Input: 22 Output: not Armstrong number
6	Write a program in Java to find minimum of three numbers using conditional operator.
7	Write a java program which should display maximum number of given 4 numbers.
8	Write a program in Java to multiply two matrix. Declare a class Matrix where 2D array is declared as instance variable and array shouldbe initialized, within class.
9	Write a java program to create a class "Matrix" that would contain integer values havingvaried Numbers of columns for each row. Print row-wise sum of the integer values for eachrow.
10	Write a Java application which takes several command line arguments, which aresupposed to be names of students and prints output as given below: (Suppose we enter 3 names then output should be asfollows).. Number of arguments = 3 1.First Student Name is = Arun 2.Second Student Name is = Hiren 3.Third Student Name is = Hitesh
11	Write a Java application to count and display frequency of letters and digits from theString given by user as command-line argument.
12	Create a class "Student" that would contain enrollment No, name, and gender andmarks as instance variables and count as static variable which stores the count of the objects; constructors and display(). Implement constructors to initialize instance variables. Also demonstrate constructor chaining. Create objects of class "Student" and displays all values of objects.

13	Write a program in Java to demonstrate use of **this** keyword. Check whether **this** canaccess the Static variables of the class or not. [Refer class student in Q12 to perform the task]
14	Create a class "Rectangle" that would contain length and width as an instance variable and count as a static variable. Define constructors **[constructor overloading (default, parameterized and copy)]** to initialize variables of objects. Define methods to find area and to display variables' value of objects which are created. [Note: define initializer block, static initializer block and the static variable and method. Also demonstrate the sequence of execution of initializer block and static initialize block]
15	Write a java program static block which will be executed before main ( ) method in a Class.
16	Write programs in Java to use Wrapper class of each primitive data types.
17	Write a class "circle" with radius as data member and count the number of instancescreated using default constructor only. [**Constructor Chaining**]
18	Create a class "Vehicle" with instance variable vehicle_type. Inherit the class in a class called "Car" with instance model_type, company name etc. display the information of the vehicle by defining the display() in both super and sub class **[Method Overriding]**
19	Create a class "Account" containing accountNo, and balance as an instance variable. Derive the Account class into two classes named "Savings" and "Current". The"Savings" class should contain instance variable named interestRate, and the "Current"class should contain instance variable called overdraftLimit. Define appropriatemethods for all the classes to enable functionalities to check balance, deposit, andwithdraw amount in Savings and Current account. [Ensure that the Account class cannot be instantiated.]
20	Write a program in Java in which a subclass constructor invokes the constructor of thesuper class and instantiate the values. [refer class Account and sub classes savingAccount and CurrentAccount in Q 19 for this task]
21	Write a program in Java to demonstrate the use of **'final'** keyword in the fielddeclaration. How it is accessed using the objects.
22	Write a java program to illustrate how to access a hidden variable. Class **A** declaresa static variable **x**. The class **B** extends **A** and declares an instance variable **x**. **display**( ) method in **B** displays both of these variables.
23	Describe **abstract** class called **Shape** which has three subclasses say **Triangle**, **Rectangle**, **and Circle**. Define one method **area ()** in the abstract class and override this **area ()** in these three subclasses to calculate for specific object i.e. area () of Triangle subclass should calculate area of triangle etc. Same for **Rectangle** and **Circle**
24	Write a java program to implement an interface called Exam with a method Pass (int mark) that returns a boolean. Write another interface called Classify with a method Division (int average) which returns a String. Write a class called Result which implements both Exam and Classify. The Pass method should return true if the mark is greater than or equal to 50 else false. The Division method must return "First" when the parameter average is 60 or more, "Second" when

	average is 50 or more but below 60, "No division" when average is less than 50.
25	Assume that there are two packages, student and exam. A student package contains Student class and the exam package contains Result class. Write a program that generates mark sheet for students.
26	Define a class A in package apack. In class A, three variables are defined of access modifiers protected, private and public. Define class B in package bpack which extends A and write display method which accesses variables of class A. Define class C in package cpack which has one method display() in that create one object of class A and display its variables. Define class ProtectedDemo in package dpack in which write main () method. Create objects of class B and C and class display method for both these objects.
27	Write a java program to implement Generic class Number_1 for both data type int and float in java.
28	Write a java program to accept string to check whether it is in Upper or Lower case. After checking, case will be reversed.
29	Write a java program to use important methods of String class.
30	Write a program in Java to demonstrate use of final class, final variable and final method
31	Write a program in Java to develop user defined exception for 'Divide by Zero' error
32	Write a program in Java to demonstrate throw, throws, finally, multiple try block and multiple catch exception.
33	Write a small application in Java to develop Banking Application in which user deposits the amount Rs 1000.00 and then start withdrawing of Rs 400.00, Rs 300.00 and it throws exception "Not Sufficient Fund" when user withdraws Rs. 500 thereafter.
34	Write a program to write at least 10 objects of the Circle class in a File and to perform basic operations: adding, retrieving, updating, removing elements.
35	Write a program for Java Generics class for Sorting operations: 1. Sorting a list according to natural ordering of elements 2. Reversing sort order 3. Sorting a list whose elements of a custom type 4. Sorting a list using a Comparator. [desirable]
36	Write a program in Java to create, write, modify, read operations on a Text file.
37	Write a java program to illustrate use of standard input stream to read the user input.
38	Write a java program to checks the existence of a specified file.
39	Write a java program to create a file to the specified location.
40	Write a java program to demonstrate the way contents are read from a file.
42	Write a java program to count the availability of text lines in the particular file. A file is read before counting lines of a particular file.

GUJARAT TECHNOLOGICAL UNIVERSITY

Program Name: Master of Computer Applications
Level: Post Graduate
Course / Subject Code : MC01094041
Course / Subject Name : Object Oriented Programming using JAVA

43	Write a generic method to count the number of elements in a collection that have a specific property (for example, odd integers, prime numbers, palindromes).
44	Write a generic method to exchange the positions of two different elements in an array.
	Thread Programming
1	Write a program to implement the concept of threading by extending "Thread" Class.
2	Write a program to implement the concept of threading by implementing "Runnable" Interface.
3.	Write a program that executes two threads. One thread displays "Thread1" every 2,000 milliseconds, and the other displays "Thread2" every 4,000 milliseconds.
4.	Write a program that executes two threads. One thread will print the even numbers and another thread will print odd numbers from 1 to 50.
5.	Write java program that create and runs following threads: a) print "A" 20 times b) print "B" 30 times c) print "C" 15 times
6.	Write a program in Java to demonstrate use of synchronization of threads when multiple threads are trying to update common variable for "Account" class.
	The collection Framework
1.	Develop a program to create Linked List for "Student" class objects references. "Student" class has std_id, std_name, Array of marks, total_ marks. Calculate total_marks for all students of Linked List. Display Linked List and also display a particular student based on student name as a command line argument.
2.	Develop a program to create Array List for "Employee" class objects references. Employee class has emp_code, emp_name, basic_sal, gross_ sal. Calculate gross_sal for all employees of Array List. Display Array List and also insert an employee object reference in a particular position (input) in Array List. Gross_sal=basic_sal+20% of basic_sal (MA)+30% of basic_sal(HRA)
3.	Develop a program to create Hash Map for "Customer" class objects references. Customer class has Bill_no, cust_mobile_no, Array of item_name, Array of item_unit_price, Array of item_count, total_price. Calculate total_price for all customers of Hash Map. Display Hash Map and also search particular customer's bill based on customer mobile no. Key is mobile no.
4.	Sort "Student" Linked List (mentioned in Q:1) based on std_name using "Comparator" interface.

Active Learning Assignment:

Consider any small application e.g. Library Management System having few transactions like Issue, Return, Renew and Report. Develop Java object oriented application programs and Present the application developed

CO- PO Mapping:

Semester - 1	Course Name : Object Oriented Programming using JAVA							
	POs							
Course Outcomes	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8
CO1	3	3	3	-	-	-	-	-
CO2	3	3	3	-	-	-	-	-
CO3	3	3	3	-	-	-	-	-
CO4	3	3	3	-	-	-	-	-
CO5	3	3	3	-	-	-	-	-
	3	**3**	**3**	-	-	-	-	-

Legend: '3' for high, '2' for medium, '1' for low and '-' for no correlation of each CO with PO.

* * * * * * *