S. No	Course Code	Name of the Course	L	T	P	C
7	19CSL302	Java Programming Lab	0	0	3	1.5

Pre-Requisites:

- Knowledge on Computer Hardware,
- Basic knowledge of problem solving.

Course objectives:

1. Problem solving using object-oriented paradigm.
2. Capacity of creating a good release strategy.
3. Capable of writing thread safe solutions for better throughput.
4. GUI Application building by handling events.
5. Capacity to write applications that collaborate users to communicate.

Course Outcomes:

CO-1: Implement object oriented programming concepts, and apply them in solving problems. **(Apply)**
CO-2: Experiment the implementation of packages and interfaces. **(Apply)**
CO-3: Experiment the concept of multithreading over single threaded programming. **(Analyze)**
CO-4: Use generic data structures of collection framework to manipulate data. **(Apply)**
CO-5: Test the GUI based network applications among multiple users through network programming. **(Analyze)**

LIST OF EXPERIMENTS

1. Create a class called Invoice that a hardware store might use to represent an invoice for an item sold at the store. An Invoice should include four pieces of information as instance variables-a part number (type String), a part description (type String),a quantity of the item being purchased (type int) and a price per item (double). Your class should have a constructor that initializes the four instance variables. Provide a set and a get method for each instance variable. In addition, provide a method named getInvoiceAmount() that calculates the invoice amount (i.e., multiplies the quantity by the price per item), then returns the amount as a double value. If the quantity is not positive, it should be set to 0. If the price per item is not positive, it should be set to 0.0. Write a test application named InvoiceTest that demonstrates class Invoice’s capabilities.[CO1]

2. Develop a Java application to generate Electricity bill. Create a class with the following members: Consumer no., consumer name, previous month reading, current month reading, and type of EB connection (i.e. domestic or commercial). Compute the bill amount using the following tariff.[CO1]

If the type of the EB connection is domestic, calculate the amount to be paid as follows:
First 100 units - Rs. 1per unit, 101-200units - Rs. 2.50per unit
201 -500 units - Rs. 4per unit, >501 units - Rs. 6per unit

If the type of the EB connection is commercial, calculate the amount to be paid as follows:
First 100 units - Rs. 2per unit, 101-200units - Rs. 4.50per unit
201 -500 units - Rs. 6per unit, >501 units - Rs. 7per unit

3. Create class SavingsAccount. Use a static variable annualInterestRate to store the annual interest rate for all account holders. Each object of the class contains a private instance variable savingsBalance indicating the amount the saver currently has ondeposit. Provide method calculateMonthlyInterest to calculate the monthly interest by multiplying the savingsBalance by annualInterestRate divided by 12 this interest should be added to savingsBalance. Provide a static method modifyInterestRate that sets the annualInterestRate to a new value.Write a program to test class SavingsAccount. Instantiate two savingsAccount objects, saver1 and saver2, with balances of $2000.00 and $3000.00, respectively. Set annualInterestRate to 4%, then calculate the monthly interest and print the new balances for both savers. Then set the annualInterestRate to 5%, calculate the next month’s interest and print the new balances for both savers.[CO1]

4. Create a class called Book to represent a book. A Book should include four pieces of information as instance variables-a book name, an ISBN number, an author name and a publisher. Your class should have a constructor that initializes the four instance variables. Provide a mutator method and accessor method (query method) for each instance variable. In addition, provide a method named getBookInfo() that returns the description of the book as a String (the description should include all the information about the book). You should use this keyword in member methods and constructor. Write a test application named BookTest to create an array of object for 30 elements for class Book to demonstrate the class Book's capabilities. [CO1]

5. To search for a word in a dictionary, you don’t review all the words; you just check one word in the middle and thus narrow down the set of remaining words to check. The divide and conquer principle may be helpful to search for a given word in a dictionary. Consider a sorted a list of numbers to apply the principle. Implement java code to search for a given word in a dictionary. (Binary Search) [CO1]

6. Suppose you have a big problem. To solve this, you recursively breaking down a problem into two or more sub-problems of the same or related type, until these become simple enough to be solved directly. The solutions to the sub-problems are then combined to give a solution to the original problem. Implement java code this with the help of a set of numbers. (Merge sort). [CO1]

7. Write a java program to make rolling a pair of dice 10,000 times and counts the number of times doubles of are rolled for each different pair of doubles. Hint: Math.random() [CO1]

8. Develop a java application to validate user information using regular expressions. [CO1]

9. Develop a java application with Employee class with Emp_name, Emp_id, Address, Mail_id, Mobile_no as members. Inherit the classes, Programmer, Assistant Professor, Associate Professor and Professor from employee class. Add Basic Pay(BP) as the member of all the inherited classes with 97% of BP as DA, 10% of BP as HRA, 12% of BP as PF, 0.1% of BP for staff club fund. Generate pay slips for the employees with their gross and net salary. [CO1]

10. Develop Rational number class in Java. Use JavaDoc comments for documentation. Your implementation should use efficient representation for a rational number, i.e. (500 / 1000) should be represented as (½).[CO1]

11. Write a Java Program to create an abstract class named Shape that contains two integers and an empty method named print Area(). Provide three classes named Rectangle, Triangle and Circle such that each one of the classes extends the class Shape. Each one of the classes contains only the method print Area () that prints the area of the given shape.[CO2]

12. Develop a java application to implement currency converter (Dollar to INR, EURO to INR, Yen to INR and vice versa), distance converter (meter to KM, miles to KM and vice versa) ,timeconverter (hours to minutes, seconds and vice versa)using packages. [CO2]
13. Write a Java Program to Handle Arithmetic Exceptions and InputMisMatchExceptions. [CO2]

14. Write a multi-threaded Java program to print all numbers below 100,000 that are both prime and Fibonacci number (some examples are 2, 3, 5, 13, etc.). Design a thread that generates prime numbers below 100,000 and writes them into a pipe. Design another thread that generates Fibonacci numbers and writes them to another pipe. The main thread should read both the pipes to identify numbers common to both. [CO3]

15. Write a java program that implements a multi-threaded application that has three threads. First thread generates a random integer every 1 second and if the value is even, second thread computes the square of the number and prints. If the value is odd, the third thread will print the value of cube of the number.[CO3]

16. Write a Java program that correctly implements the producer – consumer problem using the concept of inter-thread communication.[CO3]

17. Write a Java program that reads a file name from the user, displays information about whether the file exists, whether the file is readable, or writable, the type of file and the length of the file in bytes.[CO3]

18. Count the number of words in a given string. Implement a java code using HashMap. [CO4]

19. Implement a java code to write a method removeBadPairs that accepts an ArrayList of integers and removes any adjacent pair of integers in the list if the left element of the pair is larger than the right element of the pair. Every pair's left element is an even-numbered index in the list, and every pair's right element is an odd index in the list. For example, suppose a variable called list stores the following element values: [3, 7, 9, 2, 5, 5, 8, 5, 6, 3, 4, 7, 3, 1]

20. We can think of this list as a sequence of pairs: (3, 7), (9, 2), (5, 5), (8, 5), (6, 3), (4, 7), (3, 1). The pairs (9, 2), (8, 5), (6, 3), and (3, 1) are "bad" because the left element is larger than the right one, so these pairs should be removed. So the call of removeBadPairs(list); would change the list to store the following element values: [3, 7, 5, 5, 4, 7] .

21. If the list has an odd length, the last element is not part of a pair and is also considered "bad;" it should therefore be removed by your method. If an empty list is passed in, the list should still be empty at the end of the call. You may assume that the list passed is not null. You may not use any other arrays, lists, or other data structures to help you solve this problem.[CO4]

22. Store n student details (rollno, name, percentage, address, branch) in sorted order based a) rollno b)name c) percentage Implement a java code to this using TreeSet. [CO4]
23. Develop a Calculator using Swings with event handling.[CO5]
24. Write a Java program to implement JMenu to draw all basic shapes using Graphics. [CO4]
25. Demonstrate swing components like JTable and JTree.[CO5]
26. Write a Java program to implement JTabbedPane.[CO5]

27. Write a Java Program that implements a simple client/server application. The client sends data to a server. The server receives the data, uses it to produce a result and then sends the result back to the client. The client displays the result on the console. For ex: The data sent from the client is the radius of a circle and the result produced by the server is the area of the circle.[CO5]

CO-PO mapping Table with Justification

Contribution of Course Outcomes towards achievement of Program Outcomes & Strength of correlations (High: 3, Medium: 2, Low: 1)

	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO 11	PO12	PSO1	PSO2
CO1	2	-	2	-	3	1	-	2	2	2	-	-	2	2
CO2	2	-	2	-	3	1	-	2	2	2	-	-	2	2
CO3	2	2	2	-	3	1	-	2	2	2	-	-	2	2
CO4	2	-	2	-	3	1	-	2	3	2	-	2	2	2
CO5	2	2	2	-	3	1	-	2	3	2	-	2	2	2

Justification:

Mapping	Score	Justification
CO1-PO1, CO2-PO1 CO3-PO1, CO4-PO1 CO5-PO1	2	Problems based on science and engineering fundamentals are worked out.
CO3-PO2, CO5-PO2	2	Problem solving based on system architecture and network based applications are practiced.
CO1-PO3, CO2-PO3 CO3-PO3, CO4-PO3 CO5-PO3	2	Develop solutions for real time problems.
CO1-PO5, CO2-PO5 CO3-PO5, CO4-PO5 CO5-PO5	3	Solving problem using IDE's like Eclipse/Netbeans with plugins like JUnit, Code Coverage etc.,
CO1-PO6, CO2-PO6 CO3-PO6, CO4-PO6 CO5-PO6	2	Problem solving following the clean code practices.
CO1-PO8, CO2-PO8 CO3-PO8, CO4-PO8 CO5-PO8	2	Problem solving following Ethical Practices.
CO1-PO9, CO2-PO9 CO3-PO9	2	Collaborative working in problem solving.
CO4-PO9, CO5-PO9	3	Collaborative working in problem solving.
CO1-PO10, CO2-PO10, CO3-PO10 CO4-PO10, CO5-PO10	2	Interact with faculty and peers for investigating the problems to be solved and document the same.
CO4-PO9, CO5-PO9	2	Explore vendor specific API for latest libraries.
CO1-PSO1 CO2-PSO1 CO3-PSO1 CO4-PSO1 CO5-PSO1	2	Problem solving with ease through best software engineering.
CO1-PSO2 CO2-PSO2 CO3-PSO2 CO4-PSO2 CO5-PSO2	2	Language learned should become a skill to be successful in planning the career.

*****