Object Oriented Programming with Java(21IS4C03)

Credits: 04 **L:T:P 3:0:2**

CIE: 50 Marks **SEE: 100Marks**

SEE Hours: 03 **Max. Marks: 100**

1. Write a Java Program
   - to display any message
   - to Calculate the factorial of a number
   - to print prime numbers using CLA(Command Line Arguments)
   - Calculate the sum of first 'n' odd integer numbers. Note: Read the "n" value from keyboard.
2. Write a Java Program
   - to find the sum of all the elements in an array
   - to print all the elements in a 2D array. Get the input from key board
3. Write a program to create a room class, the attributes of this class is roomno, roomtypeand roomarea. In this class the member functions are setdata and displaydata.
4. A class called circle contains:
   - Two private instance variables: radius (of the type double) and color (of the type String), with default value of 1.0 and "red", respectively.
   - Two overloaded constructors - a default constructor with no argument, and a constructor which takes a double argument for radius.
   - Two public methods: getRadius() and getArea(), which return the radius and area of this instance, respectively.

More Basic OOP Concepts

- Constructor: Modify the class Circle to include a third constructor for constructing a Circle instance with two arguments - a double for radius and a String for color.
- Modify the test program Testcircle to construct an instance of Circle using this constructor.
- Getter: Add a getter for variable color for retrieving the color of this instance.
- public vs. private: In TestCircle, can you access the instance variable radius directly (e.g., System.out.println(c1.radius)); or assign a new value to radius (e.g., c1.radius=5.0)? Try it out and explain the error messages.
- Setter: Is there a need to change the values of radius and color of a Circle instance after it is constructed? If so, add two public methods called setters for changing the radius and color of a Circle instance.

- Keyword "this": Instead of using variable names such as r (for radius) and c (for color) in the methods' arguments, it is better to use variable names radius (for radius) and color (for color) and use the special keyword "this" to resolve the conflict between instance variables and methods' arguments.

5. Write a program to create a class named shape. In this class we have three sub classes circle, triangle and square each class has two member function named draw () and erase (). Create these using polymorphism concepts.
6. Write a program to create interface named test. In this interface the member function is square. Implement this interface in arithmetic class. Create one new class called ToTestInt in this class use the object of arithmetic class
7. Write a program to catch the ArithmeticException in the following, if any. On the occurrence of such an exception, your program should print "Exception caught: Division by zero." If there is no such exception, it will print the result of division operation on two integer values.
8. Create an Exception subclass UnderAge, which prints "Under Age" along with the age value when an object of UnderAge class is printed in the catch statement. Write a class exceptionDemo in which the method test() throws UnderAge exception if the variable age passed to it as argument is less than 18. Write main() method also to show working of the program.
9. An interface "Number" is defined in the following program. You have to declare a class "A", which will implement the interface "Number". Note that the method "findCube(n)" will return the cube of the number n.
10. Write a Java program for creating multiple threads.
    (a) Usingthread class
    (b) Using Runnable interface
11. Write a Java program that implements a multi-thread application that has three threads. First thread generates 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.
12. Write a program in java to create a String object. Initialize this object with your name. Find the length of your name using appropriate String method. Find whether character "a" is in your name or not, if yes find the number of time "a" it appear in your name. Print locations of occurrences of "a" .Try same for different String objects.
13. **Displaying a Sentence with Its Words Reversed**. Write an application that inputs a line of text, tokenizes the line with String method split and outputs the tokens in reverse order. Use space characters as delimiters.

14. **Displaying Strings in Uppercase and Lowercase.** Write an application that inputs a line of text and outputs the text twice—once in all uppercase letters and once in all lowercase letters.
15. **(Searching Strings)** Write an application that inputs a line of text and a search character and uses String method indexOf to determine the number of occurrences of the character in the text.
16. **Creating Three-Letter Strings from a Five-Letter Word.** Write an application that reads a five-letter word from the user and produces every possible three-letter string that can be derived from the letters of that word. For example, the three-letter words produced from the word "bathe" include "ate," "bat," "bet," "tab," "hat," "the" and "tea."
17. **Printing Dates in Various Formats**. Dates are printed in several common formats. Two of the more common formats are 04/25/1955 and April 25, 1955 Write an application that reads a date in the first format and prints it in the second format.
18. **Replacing Substrings and Splitting Strings** Sometimes it's useful to replace parts of a string or to split a string into pieces. For this purpose, class String provides methods replaceAll, replaceFirst and split.
19. Write program in Java for String handling which perform followings i) Checks the capacity of StringBuffer objects ii) Reverse the contents of a string given on console and convert the resultant string in upper case. iii) Read a string from console and append it to the resultant string of ii.