CHARUSAT
CHAROTAR UNIVERSITY OF SCIENCE AND TECHNOLOGY

cspit
Chandubhai S Patel Institute of Technology

Faculty of Technology and Engineering

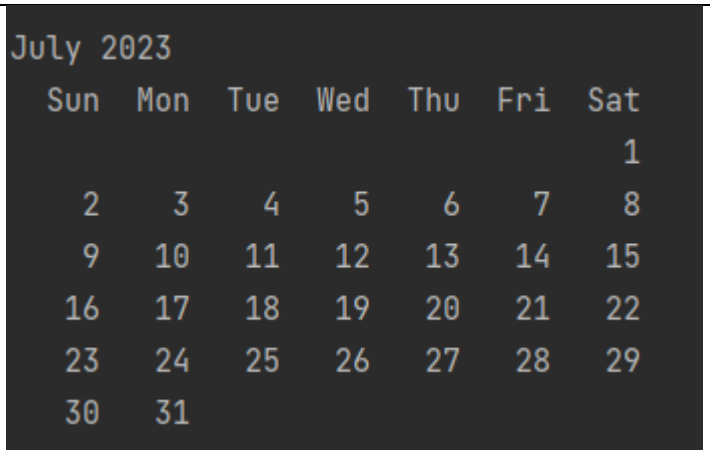
U & P U. Patel Department of Computer Engineering

Date: 28 / 06 / 2023

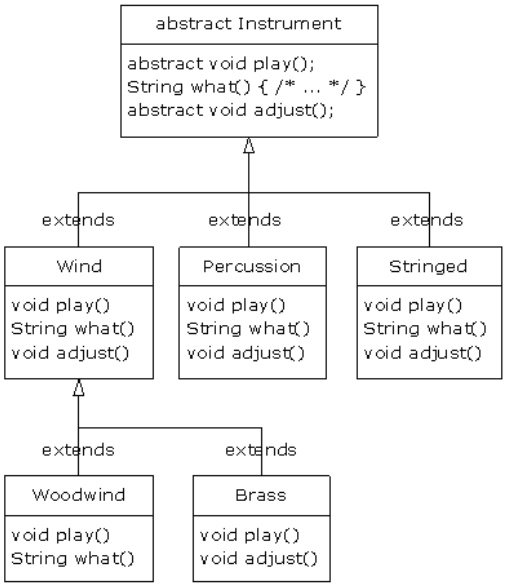
Practical List

Academic Year	:	2023-24	Semester	:	3
Course code	:	CE251	Course name	:	Java Programming

Sr.No	AIM	Hours	CO
	PART-I **Data Types, Variables, Arrays, Operators, Control Statements, String**		
1.	Introduction to Object Oriented Concepts, comparison of Java with other object oriented programming languages. Introduction to JDK, JRE, JVM, javadoc, command line argument.	2	1
2.	A typical mobile number in India is "+91-AA-BBB-CCCCC". Where the first two digits (AA)indicate a mobile system operator, the next three (BBB) denote the mobile switching code(MSC) while the remaining five digits (CCCCC) are unique to the subscriber. Write anapplication that takes a mobile number as an input from a user in above mentioned format and display code for mobile system operator, mobile switching code and last 5 digits which are unique to subscriber. Ex. For an input +91-94-999-65789, output should be :Mobile system operator code is 94 MSC is 999 Unique code is 65789	2	1
3.	Create a Java console application that displays a calendar of a given month and year. The program will take user input for the month and year they want to view the calendar and then create and display it. • Use an array to store the days in each month (including leap years), and a loop to output the calendar. • Use conditional statements to handle leap years and to align dates under the correct weekdays. **Input:** Enter the month (1-12): 7 Enter the year: 2023 **Output:**	2	1

			
4.	Java program that converts a string entered by the user to Morse code or vice versa. It will require the implementation of data structures, including arrays, loops, and conditional statements. • Create two arrays - one to contain the strings of letters to be converted, and one to contain the Morse codes. • In the program's main method, prompt the user for input to choose between the string or Morse. • For Morse code conversion, read in a string from the user; use conditional statements, looping, and array methods to convert the string to Morse-code. • For string conversion, read in a Morse-coded string from the user; use arrays, conditional statements, and looping to convert Morse code to a string. **Output:** Enter 1 for String to Morse code conversion Enter 2 for Morse code to String conversion Enter 3 To Exit 1 Enter a string: charusat Morse code: -.-. .... .- .-. ..- ... .- - Enter 1 for String to Morse code conversion Enter 2 for Morse code to String conversion Enter 3 To Exit 2 Enter Morse code: -.-. .... .- .-. ..- ... .- - String: CHARUSAT Enter 1 for String to Morse code conversion Enter 2 for Morse code to String conversion Enter 3 To Exit 3 Thank you for using Morse Code Converter!	2	1
5.	Create a Java program that simulates a guessing game, where the computer picks a random number between 1 and 100 and the user has to guess it. We can use the Scanner class to get user input and a loop to allow multiple guesses. • Prompt the user to guess the number and keep track of the number of attempts they make. • Use if-else statements to give feedback like too low or too high compared to the	2	1

	number. • Use a loop to allow the user to guess again until they guess the correct number.		
	PART-II **Object-Oriented Programming: Classes, Methods, Inheritance**		
1.	Design a class Microsoft Product consider below attributes and methods of the class. Attributes: – productNo : float – productName: String – activationKey : String – priceofProduct: float Methods: + getProductName() : String + getActivationkey() : String + getProductNo() : float + getPriceofProduct() : float + setActivationKey(acticationKey:String) : void + display() : void Store at least 5 different product data in object array and print data as per the search based on productName and productNo. Use Scanner class to take input from user end.	2	1,2
2.	Design a class named Account that contains: • A private int data field named id for the account (default 0). • A private double data field named balance for the account (default 500₹). • A private double data field named annualInterestRate that stores the current interest rate (default 7%). Assume all accounts have the same interest rate. • A private Date data field named dateCreated that stores the date when the account was created. • A no-arg constructor that creates a default account. • A constructor that creates an account with the specified id and initial balance. • The accessor and mutator methods for id, balance, and annualInterestRate. • The accessor method for dateCreated. • A method named getMonthlyInterestRate() that returns the monthly interest rate. • A method named getMonthlyInterest() that returns the monthly interest. • A method named withdraw that withdraws a specified amount from the account. • A method named deposit that deposits a specified amount to the account.	2	1,2
3.	Create a Java program the object class Point. An instance of a Point class depicts a point in two dimensional area where the smallest and biggest values for axis x and y are zero and 100 respectively. Coordinate values are integers. Class needs declarations to two private int type instance variables, one for each axis. Name the variables as you see fit. In addition to two instance variables, the class also requires a constructor and two instance methods according to the following descriptions. Constructor has two parameters, one for each axis. Values of these parameters are assigned to be the values of corresponding instance variables. Constructor must make sure that the coordinate value stays inside valid scope. If the value is lower than zero, value is zero and similarly if the value is higher than 100, value is to be 100. A toString() method is declared for the class. Method returns a character string where the coordinate values are enclosed with parenthesis separated with commas. (e.g. "(86,34)"). Value of the x coordinate is presented first. Another method declared for the class is move Method receives two parameters which are used to change the coordinate value. Parameters present the difference in the original coordinate value, not the new value directly. Method must make sure that neither of the coordinate values are smaller than zero nor higher than 100. If the change	2	1,2

	makes either of the values too low or too high, value is set to be zero or 100 respectively.Following are some examples of the method functionality: - if the old value of the coordinate is 12 and the difference is 34, new value is 46 - if the old value of the coordinate is 53 and the difference is -60, new value is 0 - if the old value of the coordinate is 63 and the difference is 82, new value is 100 Point class can be tested with PointTest class which has only the main method.		
3.	Create **GasMeter** class that keeps track of amount of refuelled gas. Class needs an instance method that receives the refuelled substance as parameter and the refuelled amount in litres. Class also needs four class methods which can print how much each substance has been used and the total amount of refuelled gas. Example output: what do you want: 1=95, 2=98, 3=Diesel (type any other number to quit): 1 How much do you want to refuel: 6,5 what do you want: 1=95, 2=98, 3=Diesel (type any other number to quit): 2 How much do you want to refuel: 5,5 what do you want: 1=95, 2=98, 3=Diesel (type any other number to quit): 0 Total used fuel: 12.0 Total used 95 octane fuel: 6.5 Total used 98 octane fuel: 5.5 Total used diesel fuel: 0.0	2	1,2
4.	Create a Java program to demonstrate the concept of method overloading using String. • Take the input as String through Scanner Class. • If String without having space, then the character 'A' replace by the 'Z' also displays length of the string. • If String has space, replace the second half of the string thought "CHARUSAT". • If String Length Is more than 10 with space, then convert String in lowercase.	2	1,2
	PART-III : Package & Interface		
1	Implement below UML diagram as per the given class name and method declared inside that. Play() method display which instrument called like "wind instrument played". what() method return instrument name. adjust() method display " instrument tuned properly". Called based on the lowest child to understand concept of method overriding also. Then separate call of the class is needed.  	2	1,2
2	Create a Recyclable Interface with the default method with the message "Give proper input". Create three different classes (Fabric, Bottel & Paper), which implement a Recyclable interface. Class requires toString methods which return the name of the recyclable materials. The toString method of the Fabric class returns the text "Febric" etc.	2	1,2

	Expected output: What do you want to recycle? Choose a number. 1 - Clothes 2 - Bottles 3 – Newspapers 4 – Exit Choose a Number: 2 Fabric recycled... Chose a number: 5 Give proper input Choose another number: 4 Thanks for coming...		
3.	Create program by writing two classes (Bottle and SodaBottle). Bottle class has one double type attribute: volume, which tells the volume of the bottle. This class also has one method: returnVolume, which returns the bottle volume. SodaBottle is derived from Bottle class and it implements the Recyclable interface class. SodaBottle also includes the name of the soda as attribute. A toString method is needed in SodaBottle class. toString returns the name of the soda and the volume of the bottle. Check example print for more precise printing needs. Recycle method should print the text "Bottle returned for recycling". Create object of SodaBottel Class only in main class. **Expected output** Type in the name of the soda: Pepsi Type in the volume of the bottle: 1 Pepsi, 1.0 litres Bottle returned for recycling.	2	1,2
4.	Create a package with class "Harmonic" have return type method to calculate harmonic series, call that class through main class and print the relevant output. Number should be take through Scanner class. **Equation of Harmonic Series** $\sum_{n=1}^{\infty} \frac{1}{n} = 1 + \frac{1}{2} + \frac{1}{3} + \frac{1}{4} + \frac{1}{5} + \cdots$	2	1,2
PART-IV : Exception Handling			
1	Write a Java Program which should ask for two integers and then add them together and print the result. Your task is to write the code which asks for the numbers and uses exception handling to check if the given numbers are integers. If the user inputs something else than an integer, "You did not type an integer!" is printed on screen. Program also includes the variable inputCorrect which value needs to be set to false if the given numbers are not integers.	2	4
2.	Write a java program with two classes (SmallException and BigException) and two methods to the ownException class (printErrorReport and testValue). SmallException and BigException are individual exception classes deriving from Exception class. Both classes require a constructor that receives a String object as parameter. Parameter is used to relay an informative message with the exception. Parameter is relayed to the superclass constructor. printErrorReport method receives the exception as a parameter and prints the error report of the exception using getMessage method. testValue method receives the tested number as a parameter. If the number is lower than five, method throws the SmallException and parameter is the message: Value is lower than 5. If the number is higher than 10, method throws the BigException and parameter is the message: Value is higher than 10.	2	4
PART-V : File Handling & Streams			
1.	Write a Java program which have readInfo method called in Person class. readInfo method is written to ClientsInFile class. The purpose of the program is to read the client information from clients.txt, make an object out of every client and finally print the	2	4

	information of every client on screen. Every client has their individual row in the file. File has every client's name and ID. Person class has one String type attribute where the information of the person (name and ID) is stored in. A toString method is required for Person class as well. toString returns the information of the person. readInfo method receives an array as parameter. This array will be used to store the created people. Method should create an object from each client in the file and store it in the array. Method returns the number of persons in the file. **Expected Output:** David 121279-2251 Matt 190970-1691 Homer 230369-2512 Joe 220755-1361		
2.	Write a java program should read grades from grades.txt, increment grades by one, write the incremented grades to the file results.txt and finally print the incremented grades from grades.txt. If the grade is 10, it is not incremented. Each grade in grades.txt has an individual row and the number of grades may differ. Grades written to results.txt are also to be written to individual rows. Use BufferedReader class to read the grades.	2	4
3	Write a java program that uses java NIO File API to create the following directory structure as a sub directory in the current folder. |--Documents | --Work |--project1.txt |--project2.txt |--Personal |--weekendPlan.txt |--summerTrip.txt No need to add anything in the txt files	2	4
	PART-VI : Multithreading		
1.	Write a program to create thread which display "Hello World" message. A. by extending Thread class B. by using Runnable interface.	2	3
2.	Write a Java program to take the salary of five different employees in an array. Salary must be incremented by 5% through the thread. After every increment thread should be sleeping for around 2000 milliseconds.	2	3
3.	Write a Java program with three different Thread names "Dhoni", "Kohli", "Hardik". Give "Dhoni" the highest priority and "Hardik" the lowest priority and check the execution of the Thread from highest to lowest place every thread in the loop of 5 iterations. After every print, there is sleep of thread around 1000 millisecond. If the execution of the thread does not go in the given order then find the alternate way and create another program through join() method.	2	3
4.	Write a program to solve producer-consumer problem using thread Synchronization.	4	3
	PART-VII : Collection Framework and Generic		
1	Create a Java program that allows the user to manage their to-do list. The program will present the user with a menu of options to manage their to-do list, including adding new tasks, displaying a list of tasks, editing tasks, and deleting tasks from the list. We'll use an ArrayList for storing the tasks and implement methods for each of the menu options to give functionality to the program.	4	5,6
2.	Let's create a program that generates a randomly ordered deck of cards using Java. The program should create a deck of cards, shuffle the order of the cards, and then display the cards in random order to the user. We'll use ArrayList to store the deck of cards and methods such as shuffle() and random() to randomize the order of the cards.	4	5,6

	• Create a class called Card with fields suit and rank. • Create a class called Deck with an ArrayList object of Card type, called cards, and methods such as shuffle() and displayCards(). • Add 52 cards with 4 different suits, aces, numbers, and face cards. • Use the shuffle() method of the Collections class in Java to shuffle the deck of cards. • Finally, display the cards using the displayCards() method		
3.	Create a Java program that simulates a simple online bookstore. The program should allow the user to browse books, add books to their cart, and checkout. We'll use HashMaps to store the items and implement methods for browsing books, adding books to the cart, and checking out. This project will cover the concepts of HashMaps, loops, conditional statements, and methods in Java. • In the main method, create a HashMap of books, with each book mapped to a unique ID. • In another method, loop over the HashMap to print out the list of books. • Use the Scanner class to get user input to add a book to the cart. • Create an ArrayList to store the items in the cart. • Create a method for checking out and iterating over the cart item to calculate the total cost	4	5,6