1. Write Java programs
   1. To print fibonacci series without using recursion and using recursion.(concept of loops, data types)
   2. To check prime numbers.
   3. To sort an array elements using bubble sort algorithm.
2. Create a class called account with the data members(Accno – integer, name String, Phone\_No: integer, balance\_amt:float), and following methods :
   1. getinput() to get input from the user
   2. Deposit() method which takes the amount to be deposited in to his/her account and do the calculation.
   3. Withdraw() method which gets the amount to be withdrawn from his/her account.
   4. Print the appropriate results.
3. Define a Stack class to implement the stack data structure. Include constructors to perform initialization, method push to push an element into the stack, method pop to remove an element from the stack and display method to display the elements of the stack.
4. Define a class Complex with data members as two real numbers, constructors for initialization these numbers, methods to add, subtract and multiply 2 complex numbers.
5. Write a java program to read 2 matrices and place the product in a third matrix . Use constructors and suitable methods.
6. Write a java program to work with strings.
7. Extract a portion of the string and print it. Variable m indicates the amount of characters to be extracted from the string starting from the nth position.
8. Read a text and count all the occurrences of a particular word.
9. Replace a substring in the given string.
10. Rearrange the string and rewrite in alphabetical order.
11. Compare two strings ignoring case.
12. Concatenate two strings.
13. Create a Personal class to hold the personal details of an person such as name, age, education, salary-(0basic, da, hra), years of experience, number of loans and loan amount. Write constructors to assign values to the data members. Include an
14. isEligible() method to indicate whether the person is eligible for loan,
15. taxPay() method to indicate the amount of tax to be paid,
16. isEligiblePromotion() to indicate whether the person is eligible for a promotion.
17. Display () method to display the details.

Enter the details of n employees and indicate their eligibility and the tax to be paid.

1. Create a Circle class with following members.

A data member that stores the radius of a circle.

A constructor function with an argument that initializes the radius

A function that computes and returns area of a circle

Create two derived classes Sector and Segment that inherit the Circle class. Both classes inherit radius and the function that returns the circle’s area from Circle.

In addition to the members inherited from Circle, Sector and Segment have some specific members as follows:

Sector

1. A data member that stores the control angle of a sector (in radians)

2. A constructor function with arguments that initialize radius and angle

3. A function that computes and returns the area of a sector

Segment

1. A data member that stores the length of a segment in a circle

2. A constructor function with arguments that initialize radius and length

3. A function that computes and returns the area of a segment

Create the main () function to instantiate an object of each class and then call appropriate member functions to compute and return the area of a circle, sector and segment.

Note : Area\_of\_circle = pi \* r 2

Area\_of\_Sector=(1/2) r 2 \*θ

Area\_of\_segment= r2 \*((r-h)/r) – (r-h) (2rh-h2)1/2 Where r is the radius of a circle, θ is the central angle of a sector in radians, h is the length of a segment and ((r-h)/r) is in radians.

1. Write a Java Program that does the following related to Inheritance:
2. Create an abstract class called Vehicle which contains the ‘year\_of\_manufacture’ data member and two abstract methods ‘getData()’ and ‘putData()’ with a constructor.
3. Create two derived classes “TwoWheeler” and “FourWheeler” and implement the abstract methods. Make “FourWheeler” as final class.
4. Create class ‘MyTwoWheeler’ which is a sub-class of “TwoWheeler” and demonstrate the use of super keyword to initialize data members of “MyTwoWheeler”.
5. Define an interface ‘Department’ with methods to readdata() and printdata(), print\_number\_designations(), number\_research\_consultancy\_projs(). Define a ‘Faculty’ class with members name, designation, age, years of experience, joining\_date and subjects\_handled.
   1. In package ISE define the ‘ISE\_department’ class that implements the ‘Department’ interface, accepts n faculty details and define all the methods. Raise a user defined exception ‘AgeException’ if the age of the faculty is > 58.
   2. In the default package define a ‘MainClass’ which uses the methods of the above classes and also displays those faculty details whose years of experience is greater than or equal to 20.
6. Write a Java Program that does the following related to Packages and Interfaces , Exception Handling:
7. Create an interface Student which gets the name and branch of a student.
8. Create a package called ‘StudentPackage’ which has a user-defined class RegisterStudent.
9. If a student registers above 30 credits for the semester, the method should throw a user-defined exception called ‘*CreditLimit*’ and display an appropriate message.
10. Create another package called ‘ResultPackage’ which displays the grade for the subject registered for particular semester and the SGPA . if SGPA is above 10 then throws an InvalidSGPA user-defined exception.
11. In the StudentPackage , collect the marks of all the subjects in 4 semesters and calculate SGPA and CGPA.
12. a. Write a java program to implement queues of Strings using an ArrayList class of the Collection framework.

b. Create a linked list of names (String type). Use an Iterator to traverse through the list and print those names whose length is < 5.

**Text Books:**

1. Herbert Schildt, “Java: The Complete Reference”, 9th Edition, McGraw Hill

**Course Outcomes**

Student will be able to:

* **CO1:** Design and develop programs using Java-specific programming constructs.

(PO2, PO 3) (PSO 1, PSO 2)

* **CO2**: Design solutions to variety of problems using Classes and Objects.

(PO 2, PO3, PO 4) (PSO 1, PSO 2)

* **CO3**: Design, reuse and develop solutions using the concepts of Inheritance, Packages

and Interfaces. (PO 2, PO3, PO 4) (PSO 1, PSO 2)

**CO4:** Identify exceptions in a program and handle them.

(PO 2, PO3, PO 4) (PSO 1, PSO 2)

* **CO5**: Use the collection framework to solve real world problems.

(PO 2, PO3, PO 4) (PSO 1, PSO 2)

**Lab Exercises:**

1. Write Java programs
   1. To print fibonacci series without using recursion and using recursion.(concept of loops, data types)
   2. To check prime numbers.
   3. To sort an array elements using bubble sort algorithm.

|  |
| --- |
| **Fibonnaci series without recursion program number-1a** |
| import java.util.Scanner; |
| class fib{ |
| public static void main(String[] args){ |
|  |
| Scanner sc = new Scanner(System.in); |
| System.out.println("-----------------------------------"); |
| System.out.print("Enter the number of terms you want:\t"); |
| int n = sc.nextInt(); |
|  |
| int a=0,b=1,c; |
| int i = 0; |
| System.out.print("The fibonacci series is: "); |
| while(i<n){ |
| System.out.print(a + " "); |
| c = a+b; |
| a = b; |
| b = c; |
| i++; |
| } |
| System.out.println("\n-----------------------------------"); |
| } |
| } |

|  |
| --- |
| **Fibonnaci series without recursion program number-1a** |
| import java.util.Scanner; |
| class fibrec{ |
|  |
| static int fib(int n){ |
| if(n == 0){ |
| return 0; |
| } |
| else if(n == 1){ |
| return 1; |
| } |
| else{ |
| return (fib(n-1)+fib(n-2)); |
| } |
| } |
|  |
| public static void main(String[] args){ |
| Scanner sc = new Scanner(System.in); |
| int n; |
| System.out.println("-----------------------------------"); |
| System.out.print("Enter the number of terms you want:\t"); |
| n = sc.nextInt(); |
|  |
| int i=0; |
| System.out.print("The fibonacci series is: \t"); |
| while(i<n){ |
| System.out.print(fib(i)+" "); |
| i++; |
| } |
| System.out.println("\n-----------------------------------"); |
| } |
| } |

|  |
| --- |
| **Prime numbers program number-1b** |
| import java.util.Scanner; |
| class prime{ |
|  |
| public static void main(String[] args){ |
|  |
| Scanner sc = new Scanner(System.in); |
| System.out.println("-----------------------------------"); |
| System.out.print("Enter the number you wanna check:\t"); |
| int n = sc.nextInt(); |
| int flag =0; |
| for(int i=2;i<=n/2;i++){ |
| if(n%i == 0){ |
| flag = 1; |
| break; |
| } |
| else{ |
| flag = 0; |
| } |
| } |
| if(flag == 1){ |
| System.out.println("The entered number is not a prime number"); |
| } |
| else{ |
| if(n != 1 || n != 0){ |
| System.out.println("The entered number is a prime number"); |
| } |
| else{ |
| System.out.println("The entered number is neither prime nor composite"); |
| } |
| } |
| System.out.println("-----------------------------------"); |
| } |
| } |

|  |
| --- |
| **Bubble sort program number-1c** |
| import java.util.Scanner; |
| class bubbleSort{ |
|  |
| public static void main(String[] args){ |
|  |
| Scanner sc = new Scanner(System.in); |
| //Accepting the number of elements in the array |
| System.out.println("Enter the number of elements in the array"); |
| int n = sc.nextInt(); |
|  |
| int[] array = new int[n]; |
| System.out.println("Enter the values"); |
| //Accepting values in the array |
| for(int i=0;i<n;i++){ |
| array[i] = sc.nextInt(); |
| } |
|  |
| //Sorting it using the bubble sort technique |
| for(int i=0;i<n;i++){ |
| for(int j=0;j<n-i-1;j++){ |
| if(array[j]>array[j+1]){ |
| int temp = array[j+1]; |
| array[j+1] = array[j]; |
| array[j] = temp; |
| } |
| } |
| } |
| //Printing the sorted array |
| System.out.print("The sorted array is:\t"); |
| for(int i=0;i<n;i++){ |
| System.out.print(array[i]+" "); |
| } |
|  |
| } |
| } |

1. Create a class called account with the data members(Accno – integer, name String, Phone\_No: integer, balance\_amt:float), and following methods :
   1. getinput() to get input from the user
   2. Deposit() method which takes the amount to be deposited in to his/her account and do the calculation.
   3. Withdraw() method which gets the amount to be withdrawn from his/her account.
   4. Print the appropriate results.

|  |
| --- |
| **Account Program Number-2** |
| import java.util.Scanner; |
| class account{ |
|  |
| Scanner sc = new Scanner(System.in); |
| long accNo; |
| long phoneNo; |
| String name; |
| double balanceAmt; |
| //Method to get the details of the user. |
| public void getInput(){ |
| System.out.print("Enter the account number:\t"); |
| accNo = sc.nextLong(); |
| sc.nextLine(); |
| System.out.print("Enter the account holders name:\t"); |
| name = sc.nextLine(); |
|  |
| System.out.print("Enter the account holders phone number:\t"); |
| phoneNo = sc.nextLong(); |
|  |
| balanceAmt =0; |
| } |
| //Method to deposit the amount in the bank |
| public void deposit(){ |
| System.out.println("Enter the amount you wanna deposit:\t"); |
| double deposit = sc.nextDouble(); |
|  |
| balanceAmt += deposit; |
| System.out.println("Amount Deposited Successfully"); |
| System.out.println("The account balance is:\t"+balanceAmt); |
| } |
| //Method to withdraw the amount from the bank |
| public void withdraw(){ |
| System.out.println("Enter the amount you wanna withdraw:\t"); |
| double withdraw = sc.nextDouble(); |
| if(balanceAmt-withdraw>=0){ |
| balanceAmt -= withdraw; |
| System.out.println("Amount withdrawn successful"); |
| } |
| else{ |
| System.out.println("Your account has insufficient funds"); |
| } |
| System.out.println("The account balance is:\t"+balanceAmt); |
| } |
| //Method to print the details |
| public void printDetails(){ |
| System.out.println("Account Number:\t"+accNo); |
| System.out.println("Account holders name:\t"+name); |
| System.out.println("Phone number:\t"+phoneNo); |
| System.out.println("Current balance in the account:\t"+balanceAmt); |
| } |
| public static void main(String[] args){ |
| Scanner sc = new Scanner(System.in); |
| account a1 = new account(); //Object reference for the account |
| while(true){ |
| System.out.println("1.Enter the user details"); |
| System.out.println("2.Deposit money in the account"); |
| System.out.println("3.Withdraw money from the account"); |
| System.out.println("4.Get the account details"); |
| System.out.println("5. Exit!"); |
| System.out.print("Enter your choice:\t"); |
| int ch = sc.nextInt(); |
| switch(ch){ |
| case 1: |
| a1.getInput(); |
| break; |
| case 2: |
| a1.deposit(); |
| break; |
| case 3: |
| a1.withdraw(); |
| break; |
| case 4: |
| a1.printDetails(); |
| break; |
| case 5: |
| System.exit(0); |
| break; |
| default: |
| System.out.println("Invalid choice!"); |
| } |
| System.out.println(""); |
| } |
| } |
| } |

3.Define a Stack class to implement the stack data structure. Include constructors to perform initialization, method push to push an element into the stack, method pop to remove an element from the stack and display method to display the elements of the stack.

|  |
| --- |
| **Stack program number-3** |
| import java.util.Scanner; |
| class stack{ |
| int top; |
| int size; |
| int stack[]; |
|  |
| stack(int size,int top){ |
| this.size = size; |
| this.top = top; |
| this.stack = new int[size]; |
| } |
| public void push(int ele){ |
| if(top == size-1){ |
| System.out.println("Stack Overflow"); |
| } |
| else{ |
| stack[++top] = ele; |
| System.out.println("Element successfully inserted"); |
| } |
| } |
|  |
| public int pop(){ |
| if(top == -1){ |
| return -1; |
| } |
| else{ |
| return stack[top--]; |
| } |
| } |
|  |
| public void display(){ |
| if(top == -1){ |
| System.out.println("The stack is empty"); |
| } |
| else{ |
| System.out.println("The elemnets of the stack are:\t"); |
| for(int i=top;i>=0;i--){ |
| System.out.print(stack[i]+" "); |
| } |
| } |
| } |
|  |
| public static void main(String[] args){ |
| Scanner sc = new Scanner(System.in); |
| System.out.print("Enter the size of the stack:\t"); |
| int size = sc.nextInt(); |
|  |
| stack s1 = new stack(size,-1); |
|  |
| System.out.println("1. Push"); |
| System.out.println("2. Pop"); |
| System.out.println("3. Display"); |
| System.out.println("4. Exit!"); |
|  |
| while(true){ |
| System.out.print("\nEnter your choice:\t"); |
| int ch = sc.nextInt(); |
|  |
| switch(ch){ |
| case 1: |
| System.out.print("Enter the element you want to insert:\t"); |
| int ele1 = sc.nextInt(); |
| s1.push(ele1); |
| break; |
| case 2: |
| int ele2 = s1.pop(); |
| if(ele2 != -1){ |
| System.out.println("The elemnet popped out from the stck is:"+ele2); |
| } |
| else{ |
| System.out.println("Stack underflow!"); |
| } |
| break; |
| case 3: |
| s1.display(); |
| break; |
| case 4: |
| System.exit(0); |
| break; |
| default: |
| System.out.println("Invalid choice!"); |
| } |
| } |
| } |
| } |

4.Define a class Complex with data members as two real numbers, constructors for initialization these numbers, methods to add, subtract and multiply 2 complex numbers.

|  |
| --- |
| **Complex number program number-4** |
| import java.util.Scanner; |
| class complex{ |
|  |
| Scanner sc = new Scanner(System.in); |
| int a; |
| int b; |
| complex(){ |
| this.a = 0; |
| this.b =0; |
| } |
|  |
| public void getInput(){ |
| System.out.println("Enter the real part of the complex number"); |
| a = sc.nextInt(); |
| System.out.println("Enter the imaginary part of the complex number"); |
| b = sc.nextInt(); |
| } |
| public complex sum\_c(complex c1,complex c2){ |
| complex c3 = new complex(); |
| c3.a = c1.a + c2.a; |
| c3.b = c1.b + c2.b; |
| return c3; |
| } |
|  |
| public complex diff\_c(complex c1,complex c2){ |
| complex c3 = new complex(); |
| c3.a = c1.a - c2.a; |
| c3.b = c1.b - c2.b; |
| return c3; |
| } |
|  |
| public complex mul\_c(complex c1, complex c2){ |
| complex c3 = new complex(); |
| c3.a = (c1.a \* c2.a) - (c1.b \* c2.b); |
| c3.b = (c1.a \* c2.b) + (c2.a \* c1.b); |
| return c3; |
| } |
|  |
| public static void main(String[] args){ |
|  |
| Scanner sc = new Scanner(System.in); |
| System.out.println("1. Addition of two complex numbers"); |
| System.out.println("2. Subtraction of two complex numbers"); |
| System.out.println("3. Multiplication of two complex numbers"); |
| System.out.println("4. Exit!"); |
|  |
| complex c1 = new complex(); |
| complex c2 = new complex(); |
| complex compute = new complex(); |
|  |
| while(true){ |
| System.out.println("Enter your choice"); |
| int ch = sc.nextInt(); |
|  |
| switch(ch){ |
| case 1: |
| System.out.println("Enter the first complex number"); |
| c1.getInput(); |
| System.out.println("Enter the second complex number"); |
| c2.getInput(); |
| compute = compute.sum\_c(c1,c2); |
| System.out.println("The sum of "+c1.a + " +i " + c1.b + " and "+c2.a + " +i " + c2.b +" is "+compute.a + " +i (" + compute.b+")"); |
| break; |
| case 2: |
| System.out.println("Enter the first complex number"); |
| c1.getInput(); |
| System.out.println("Enter the second complex number"); |
| c2.getInput(); |
| compute = compute.diff\_c(c1,c2); |
| System.out.println("The sum of "+c1.a + " +i " + c1.b + " and "+c2.a + " +i " + c2.b +" is "+compute.a + " +i (" + compute.b +")" ); |
| break; |
| case 3: |
| System.out.println("Enter the first complex number"); |
| c1.getInput(); |
| System.out.println("Enter the second complex number"); |
| c2.getInput(); |
| compute = compute.mul\_c(c1,c2); |
| System.out.println("The sum of "+c1.a + " +i " + c1.b + " and "+c2.a + " +i " + c2.b +" is "+compute.a + " +i (" + compute.b+")" ); |
| break; |
| case 4: |
| System.exit(0); |
| break; |
| default: |
| System.out.println("Invalid choice"); |
| break; |
| } |
| } |
| } |
| } |

5.Write a java program to read 2 matrices and place the product in a third matrix . Use constructors and suitable methods.

|  |
| --- |
| **Matrix program number-5** |
| import java.util.Scanner; |
| class matrix{ |
| int rows; |
| int columns; |
| int array[][]; |
| Scanner sc = new Scanner(System.in); |
| matrix(int rows,int columns){ |
| this.rows = rows; |
| this.columns = columns; |
| this.array = new int[rows][columns]; |
| } |
| public void getInput(){ |
| for(int i=0;i<rows;i++){ |
| for(int j=0;j<columns;j++){ |
| array[i][j] = sc.nextInt(); |
| } |
| } |
| } |
|  |
| public int matrixMul(matrix m1,matrix m2){ |
| int flag =0; |
| if(m1.columns != m2.rows){ |
| flag =1; |
| return flag; |
| } |
| else{ |
| for(int i=0;i<m1.rows;i++){ |
| for(int j=0;j<m2.columns;j++){ |
| for(int k=0;k<m1.columns;k++){ |
| array[i][j] += m1.array[i][k] \* m2.array[k][j]; |
| } |
| } |
| } |
| return flag; |
| } |
| } |
|  |
| public void printMatrix(){ |
| for(int i=0;i<rows;i++){ |
| for(int j=0;j<columns;j++){ |
| System.out.print(array[i][j]+" "); |
| } |
| System.out.println(); |
| } |
| } |
|  |
| public static void main(String[] args){ |
| Scanner sc = new Scanner(System.in); |
|  |
| //For matrix 1. |
| System.out.println("For matrix 1"); |
| System.out.print("Enter the no of rows:\t"); |
| int rows1 = sc.nextInt(); |
| System.out.print("Enter the no of columns:\t"); |
| int columns1 = sc.nextInt(); |
|  |
| matrix m1 = new matrix(rows1,columns1); |
| System.out.println("Enter the elements of the matrix"); |
| m1.getInput(); |
|  |
|  |
| //For matrix 2. |
| System.out.println("For matrix 2"); |
| System.out.print("Enter the no of rows:\t"); |
| int rows2 = sc.nextInt(); |
| System.out.print("Enter the no of columns:\t"); |
| int columns2 = sc.nextInt(); |
|  |
| matrix m2 = new matrix(rows2,columns2); |
| System.out.println("Enter the elements of the matrix"); |
| m2.getInput(); |
| matrix m3 = new matrix(m1.rows,m2.columns); |
| int flag = m3.matrixMul(m1,m2); |
|  |
| //Printing the matrices. |
| System.out.println("The First matrix is"); |
| m1.printMatrix(); |
|  |
| System.out.println("The Second matrix is"); |
| m2.printMatrix(); |
|  |
| if(flag == 0){ |
| System.out.println("The multiplication of two matrices is"); |
| m3.printMatrix(); |
| } |
| else{ |
| System.out.println("The multiplication of the two matrices is not possible"); |
| } |
| } |
| } |

6.Write a java program to work with strings.

1. Extract a portion of the string and print it. Variable m indicates the amount of characters to be extracted from the string starting from the nth position.
2. Read a text and count all the occurrences of a particular word.
3. Replace a substring in the given string.
4. Rearrange the string and rewrite in alphabetical order.
5. Compare two strings ignoring case.
6. Concatenate two strings.

|  |
| --- |
| **String program number-6** |
| import java.util.Scanner; |
| import java.util.Arrays; |
| class strings{ |
| public static void main(String[] args){ |
|  |
| Scanner sc = new Scanner(System.in); |
| System.out.println("1. Extract a portion of a string"); |
| System.out.println("2. Count the occurences of a particular word in a string"); |
| System.out.println("3. Replace a substring in a given string"); |
| System.out.println("4. Rearrange the strings in alphabetical order"); |
| System.out.println("5. Compare two strings ignoring case"); |
| System.out.println("6. Concatenate two strings"); |
| System.out.println("7. Exit!"); |
|  |
| while(true){ |
| System.out.print("Enter your choice:\t"); |
| int ch = sc.nextInt(); |
| sc.nextLine(); |
|  |
| switch(ch){ |
| case 1: |
| System.out.print("Enter the String:\t"); |
| String s = sc.nextLine(); |
| System.out.print("\nEnter the position of the string from which you wanna start extracting:\t"); |
| int n = sc.nextInt(); |
| System.out.print("\nEnter the no of characters to be extracted:\t"); |
| int m = sc.nextInt(); |
| String k = s.substring(n-1,m+n-1); |
| System.out.println("\nThe extracted string is: "+k); |
| break; |
|  |
|  |
| case 2: |
| System.out.print("Enter the String:\t"); |
| s = sc.nextLine(); |
| System.out.print("\nEnter the word whose occurences you wanna find:\n"); |
| k = sc.next(); |
| int counter = 0; |
| String arr[] = s.split(" "); |
| for(int i=0;i<arr.length;i++){ |
| if(arr[i].equals(k)){ |
| counter++; |
| } |
| } |
| System.out.println("\nThe no of occurences of the word "+k+" are "+counter); |
| break; |
|  |
| case 3: |
| System.out.print("Enter the string:\t"); |
| s = sc.nextLine(); |
| System.out.print("\nEnter the substring you wanna replace:\t"); |
| k = sc.next(); |
| System.out.print("\nEnter the substring you wanna replace with:\t"); |
| String p = sc.next(); |
| String t = s.replace(k,p); |
| System.out.println("\nThe new string is: "+t); |
| break; |
|  |
| case 4: |
| System.out.print("Enter the string:\t"); |
| s = sc.nextLine(); |
| k = s.toLowerCase(); |
| char[] arr2 = k.toCharArray(); |
| Arrays.sort(arr2); |
| p = new String(arr2); |
| System.out.println("\nThe string arranged aphabetically is: "+p); |
| break; |
|  |
| case 5: |
| System.out.print("Enter the string 1:\t"); |
| s = sc.nextLine(); |
| System.out.print("\nEnter the string 2:\t"); |
| k = sc.nextLine(); |
| boolean cmp = s.equalsIgnoreCase(k); |
| if(cmp){ |
| System.out.println("\nThe two strings are equal"); |
| } |
| else{ |
| System.out.println("\nThe two strings are not equal"); |
| } |
| break; |
|  |
| case 6: |
| System.out.print("Enter the string 1:\t"); |
| s = sc.nextLine(); |
| System.out.print("Enter the string 2:\t"); |
| k = sc.nextLine(); |
| p = s+" "+k; |
| System.out.println("\nThe concatenated strings is: "+p); |
| break; |
| case 7: |
| System.exit(0); |
| break; |
| default: |
| System.out.println("Invalid choice"); |
| } |
| } |
| } |
| } |

7.Create a Personal class to hold the personal details of an person such as name, age, education, salary-(0basic, da, hra), years of experience, number of loans and loan amount. Write constructors to assign values to the data members. Include an

1. isEligible() method to indicate whether the person is eligible for loan,
2. taxPay() method to indicate the amount of tax to be paid,
3. isEligiblePromotion() to indicate whether the person is eligible for a promotion.
4. Display () method to display the details.

Enter the details of n employees and indicate their eligibility and the tax to be paid.

|  |
| --- |
| import java.util.Scanner; |
| class personal{ |
| String name; |
| int age; |
| String education; |
| double basic; |
| double hra; |
| double da; |
| int years\_of\_exp; |
| int no\_of\_loans; |
| double loan\_amt[]; |
|  |
| personal(String name,int age,String education,double basic,double hra,double da,int years\_of\_exp,int no\_of\_loans,double[] loan\_amt){ |
| this.name = name; |
| this.age = age; |
| this.education = education; |
| this.basic = basic; |
| this.hra = hra; |
| this.da = da; |
| this.years\_of\_exp = years\_of\_exp; |
| this.no\_of\_loans = no\_of\_loans; |
| this.loan\_amt = new double[no\_of\_loans]; |
| this.loan\_amt = loan\_amt; |
| } |
| void isEligible(personal p){ |
| if((p.basic+p.hra+p.da)>500000 && p.no\_of\_loans <=2){ |
| System.out.println("The person is eligible for loan"); |
| } |
| else{ |
| System.out.println("The person is not eleigible for loan"); |
| } |
| } |
|  |
| void taxPay(personal p){ |
| double sal = p.basic + p.hra + p.da; |
| if(sal<=250000){ |
| System.out.println("No tax is to be paid"); |
| } |
| else if(sal<=500000){ |
| System.out.println("The tax to be paid is: "+0.2\*sal); |
| } |
| else if(sal<=1000000){ |
| System.out.println("The tax to be paid is: "+0.3\*sal); |
| } |
| else{ |
| System.out.println("The tax to be paid is: "+0.4\*sal); |
| } |
| } |
|  |
| void isEligiblePromotion(personal p){ |
| if(p.years\_of\_exp >=10 && p.age>=43){ |
| System.out.println("The person is eligible for promotion"); |
| } |
| else{ |
| System.out.println("The person is not eleigible for promotion"); |
| } |
| } |
|  |
| void display(personal p){ |
| System.out.println("Name of the person is: "+p.name); |
| System.out.println("Age of the person is: "+p.age); |
| System.out.println("Education of the person is: "+p.education); |
| System.out.println("The total salary of the person is: "+(p.basic+p.hra+p.da)); |
| System.out.println("The years of experience of the person is: "+p.years\_of\_exp); |
| System.out.println("The no of loans in the name of the person is: "+p.no\_of\_loans); |
| System.out.print("The lloan amount for each loan is:\t "); |
| for(int i=0;i<p.no\_of\_loans;i++){ |
| System.out.print(p.loan\_amt[i]+" "); |
| } |
|  |
| System.out.println(); |
| p.isEligible(p); |
| p.taxPay(p); |
| p.isEligiblePromotion(p); |
| } |
|  |
| public static void main(String[] args){ |
| Scanner sc = new Scanner(System.in); |
| System.out.println("Enter the no of employees whose detail you wanna enter"); |
| int n = sc.nextInt(); |
| sc.nextLine(); |
| personal[] p = new personal[n]; |
|  |
| for(int i=0;i<n;i++){ |
|  |
| System.out.println("Enter the name"); |
| String name = sc.nextLine(); |
| System.out.println("Enter the age"); |
| int age = sc.nextInt(); |
| sc.nextLine();//Have to give after int |
| System.out.println("Enter the education"); |
| String education = sc.nextLine(); |
| System.out.println("Enter the basic salary"); |
| double basic = sc.nextDouble(); |
| System.out.println("Enter the hra"); |
| double hra = sc.nextDouble(); |
| System.out.println("Enter the da"); |
| double da = sc.nextDouble(); |
| System.out.println("Enter the years of experience"); |
| int years\_of\_exp = sc.nextInt(); |
| System.out.println("Enter the no of loans"); |
| int no\_of\_loans = sc.nextInt(); |
| double[] loan\_amt = new double[no\_of\_loans]; |
| for(int j=0;j<no\_of\_loans;j++){ |
| System.out.println("Enter the loan amount "+(j+1)); |
| loan\_amt[j] = sc.nextDouble(); |
| } |
| sc.nextLine(); |
| p[i] = new personal(name,age,education,basic,hra,da,years\_of\_exp,no\_of\_loans,loan\_amt); |
| System.out.println("The details of the employee are: "); |
| System.out.println("---------------------------------------"); |
| p[i].display(p[i]); |
| System.out.println("---------------------------------------"); |
| } |
| } |
| } |

8.Create a Circle class with following members.

A data member that stores the radius of a circle.

A constructor function with an argument that initializes the radius

A function that computes and returns area of a circle

Create two derived classes Sector and Segment that inherit the Circle class. Both classes inherit radius and the function that returns the circle’s area from Circle.

In addition to the members inherited from Circle, Sector and Segment have some specific members as follows:

Sector

1. A data member that stores the control angle of a sector (in radians)

2. A constructor function with arguments that initialize radius and angle

3. A function that computes and returns the area of a sector

Segment

1. A data member that stores the length of a segment in a circle

2. A constructor function with arguments that initialize radius and length

3. A function that computes and returns the area of a segment

Create the main () function to instantiate an object of each class and then call appropriate member functions to compute and return the area of a circle, sector and segment.

Note : Area\_of\_circle = pi \* r 2

Area\_of\_Sector=(1/2) r 2 \*θ

Area\_of\_segment= r2 \*((r-h)/r) – (r-h) (2rh-h2)1/2 Where r is the radius of a circle, θ is the central angle of a sector in radians, h is the length of a segment and ((r-h)/r) is in radians.

|  |
| --- |
| **Circle program number-8** |
| import java.util.Scanner; |
| import java.lang.Math; |
| class Circle{ |
| double radius; |
| Circle(double r){ |
| this.radius = r; |
| } |
| double circle\_area(){ |
| return Math.PI\*radius\*radius; |
| } |
| } |
|  |
| class Sector extends Circle{ |
| double angle; //To be taken in radians |
| Sector(double r,double a){ |
| super(r); |
| this.angle = a; |
| } |
| double sector\_area(){ |
| return (0.5\*radius\*radius\*angle); |
| } |
| } |
| class Segment extends Circle{ |
| double length; |
| Segment(double r,double l){ |
| super(r); |
| this.length = l; |
| } |
| double segment\_area(){ |
|  |
| // r2\*((r-h)/r) – (r-h) (2rh-h2)1/2 |
| return (((radius\*radius)\*Math.acos((radius-length)/radius)) - ((radius-length)\*Math.sqrt(((2\*radius\*length)-(length\*length))))); |
| } |
| } |
| class inheritance{ |
| public static void main(String[] args){ |
| Scanner sc = new Scanner(System.in); |
| System.out.println("1. Area of circle"); |
| System.out.println("2. Area of sector"); |
| System.out.println("3. Area of segment"); |
| System.out.println("4. Exit!"); |
| while(true){ |
| System.out.println("Enter your choice"); |
| int ch = sc.nextInt(); |
|  |
| switch(ch){ |
| case 1: |
| System.out.println("Enter the radius of the circle"); |
| double r = sc.nextDouble(); |
| Circle c = new Circle(r); |
| System.out.println("The area of the circle with radius "+r+ " is "+c.circle\_area()); |
| break; |
|  |
| case 2: |
| System.out.println("Enter the radius of the circle"); |
| r = sc.nextDouble(); |
| System.out.println("Enter the sector angle in degrees"); |
| double d = sc.nextDouble(); |
| double p = (Math.PI/180)\*d; |
| Sector s = new Sector(r,p); |
| System.out.println("The area of the sector with radius "+r+ " and sector angle "+ d+"(degrees) is "+s.sector\_area()); |
| break; |
|  |
| case 3: |
| System.out.println("Enter the radius of the circle"); |
| r = sc.nextDouble(); |
| System.out.println("Enter the length of the segment"); |
| double l = sc.nextDouble(); |
| Segment se = new Segment(r,l); |
| System.out.println("The area of the segment with radius "+r+ " and segement length "+ l+" is "+se.segment\_area()); |
| break; |
|  |
| case 4: |
| System.exit(0); |
| break; |
| default: |
| System.out.println("Invalid choice"); |
| } |
| } |
| } |
| } |

9.Write a Java Program that does the following related to Inheritance:

1. Create an abstract class called Vehicle which contains the ‘year\_of\_manufacture’ data member and two abstract methods ‘getData()’ and ‘putData()’ with a constructor.
2. Create two derived classes “TwoWheeler” and “FourWheeler” and implement the abstract methods. Make “FourWheeler” as final class.
3. Create class ‘MyTwoWheeler’ which is a sub-class of “TwoWheeler” and demonstrate the use of super keyword to initialize data members of “MyTwoWheeler”.

|  |
| --- |
| **Vehicle Program Number-9** |
| a. /\*Write a Java Program that does the following related to Inheritance: |
| Create an abstract class called Vehicle which contains the |
| „year\_of\_manufacture‟ data member and two abstract methods „getData()‟ and |
| „putData()‟ with a constructor. |
| b. Create two derived classes “TwoWheeler” and “FourWheeler” and implement |
| the abstract methods. Make “FourWheeler” as final class. |
| c. |
| Create class „MyTwoWheeler‟ which is a sub-class of “TwoWheeler” and |
| demonstrate the use of super keyword to initialize data members of |
| “MyTwoWheeler”.\*/ |
| } |
| class TwoWheeler extends Vehicle { |
|  |
| @Override |
| public int getData() { |
|  |
| return year\_of\_manufacture; |
| } |
| @Override |
| public void putData(int year) { |
| year\_of\_manufacture = year; |
| } |
|  |
| TwoWheeler(int year) { |
| year\_of\_manufacture = year; |
| } |
| } |
|  |
| final class FourWheeler extends Vehicle { |
|  |
| @Override |
| public int getData() { |
|  |
| return year\_of\_manufacture; |
|  |
| } |
|  |
| @Override |
| public void putData(int year) { |
| year\_of\_manufacture = year; |
|  |
| } |
| FourWheeler(int year){ |
| year\_of\_manufacture=year; |
| } |
|  |
| } |
|  |
| class MyTwoWheeler extends TwoWheeler { |
| MyTwoWheeler(int year) { |
| super(year); |
| } |
|  |
| } |

|  |
| --- |
| package com.msrit.problem9; |
|  |
| public class VehicleDriver { |
|  |
| public static void main(String[] args) { |
| MyTwoWheeler myTwoWheeler=new MyTwoWheeler(2000); |
| System.out.println(myTwoWheeler.getData()); |
| } |
|  |
| } |

10.Define an interface ‘Department’ with methods to readdata() and printdata(), print\_number\_designations(), number\_research\_consultancy\_projs(). Define a ‘Faculty’ class with members name, designation, age, years of experience, joining\_date and subjects\_handled.

* 1. In package ISE define the ‘ISE\_department’ class that implements the ‘Department’ interface, accepts n faculty details and define all the methods. Raise a user defined exception ‘AgeException’ if the age of the faculty is > 58.
  2. In the default package define a ‘MainClass’ which uses the methods of the above classes and also displays those faculty details whose years of experience is greater than or equal to 20.

|  |
| --- |
| package Q10; |
|  |
| public class Faculty{ |
| public String name,designation,joining\_date,subjects\_handled; |
| public int age,years\_of\_experience,no\_of\_designations,no\_of\_research\_projects; |
| } |
|  |
| import java.util.Scanner; |
| import Q10.ISE.ISE\_department; |
| import Q10.Faculty; |
|  |
| public class MainClass{ |
|  |
| public static void main(String[] args){ |
| Scanner sc = new Scanner(System.in); |
| int counter =0; |
| int flag1 =0; |
| int flag2 =0; |
| int flag3 =0; |
| int flag4 =0; |
|  |
| ISE\_department ise = new ISE\_department(); |
| Faculty[] f = new Faculty[20]; |
|  |
| while(true){ |
| System.out.println("1. Add a faculty"); |
| System.out.println("2. Get the details of a particular faculty"); |
| System.out.println("3. Get the name of experienced faculty"); |
| System.out.println("4. Know the no of designations a particular faculty had"); |
| System.out.println("5. Know the no fo research consultancy proects done by a faculty"); |
| System.out.println("6. Exit!"); |
|  |
| System.out.println("Enter your choice:\t"); |
| int ch = sc.nextInt(); |
|  |
| switch(ch){ |
| case 1: |
| f[counter] = new Faculty(); |
| ise.readData(f[counter]); |
| counter++; |
| break; |
| case 2: |
| System.out.println("Enter the name of the faculty whose details you want"); |
| sc.nextLine(); |
| String x = sc.nextLine(); |
|  |
| for(int i=0;i<counter;i++){ |
| if(x.equals(f[i].name)){ |
| flag1 = 1; |
| ise.printData(f[i]); |
| break; |
| } |
| } |
| if(flag1 == 0){ |
| System.out.println("Faculty not found!"); |
| } |
| break; |
| case 3: |
| System.out.println("The faculty with experience greater than 20 years are:"); |
| for(int i=0;i<counter;i++){ |
| if(f[i].years\_of\_experience >=20){ |
| System.out.println(f[i].name); |
| flag2 = 1; |
| } |
| } |
| if(flag2 == 0){ |
| System.out.println("No Such record found"); |
| } |
| break; |
| case 4: |
| System.out.println("Enter the name of the faculty whose detail you want"); |
| sc.nextLine(); |
| x = sc.nextLine(); |
|  |
| for(int i=0;i<counter;i++){ |
| if(x.equals(f[i].name)){ |
| flag3 = 1; |
| System.out.println("The no of designations the faculty had are: "+f[i].no\_of\_designations); |
| break; |
| } |
| } |
| if(flag3 == 0){ |
| System.out.println("Faculty not found!"); |
| } |
| break; |
| case 5: |
| System.out.println("Enter the name of the faculty whose detail you want"); |
| sc.nextLine(); |
| x = sc.nextLine(); |
|  |
| for(int i=0;i<counter;i++){ |
| if(x.equals(f[i].name)){ |
| flag4 = 1; |
| System.out.println("The no of reserach consultancy projects faculty did are: "+f[i].no\_of\_research\_projects); |
| break; |
| } |
| } |
| if(flag4 == 0){ |
| System.out.println("Faculty not found!"); |
| } |
| break; |
| case 6: |
| System.exit(0); |
| break; |
| default: |
| System.out.println("Invalid choice!"); |
| } |
| } |
|  |
| } |
| } |

|  |
| --- |
| package Q10; |
| import Q10.Faculty; |
| public interface department{ |
|  |
| void readData(Faculty f); |
| void printData(Faculty f); |
| void print\_number\_designations(Faculty f); |
| void number\_research\_consultancy\_projects(Faculty f); |
|  |
| } |

11.Write a Java Program that does the following related to Packages and Interfaces , Exception Handling:

1. Create an interface Student which gets the name and branch of a student.
2. Create a package called ‘StudentPackage’ which has a user-defined class RegisterStudent.
3. If a student registers above 30 credits for the semester, the method should throw a user-defined exception called ‘*CreditLimit*’ and display an appropriate message.
4. Create another package called ‘ResultPackage’ which displays the grade for the subject registered for particular semester and the SGPA . if SGPA is above 10 then throws an InvalidSGPA user-defined exception.
5. In the StudentPackage , collect the marks of all the subjects in 4 semesters and calculate SGPA and CGPA.

|  |
| --- |
| package Q11.ResultPackage; |
| import java.util.Scanner; |
| import Q11.StudentPackage.CGPA; |
|  |
| class InvalidSGPA extends Exception{ |
| InvalidSGPA(String s){ |
| super(s); |
| } |
| } |
| public class Result extends CGPA{ |
|  |
| public char grade[][] = new char[4][6]; |
| public int credit[] = new int[6]; |
| public String name =""; |
| public String branch = ""; |
| public int semester; |
| public String subjects[] = new String[6]; |
| public int totalCredits =0; |
| Scanner sc = new Scanner(System.in); |
| public void result(){ |
| for(int i=0;i<6;i++){ |
| System.out.println("Enter the marks in Subject["+(i+1)+"]"); |
| marks[semester-1][i] = sc.nextInt(); |
| } |
|  |
| for(int i=0;i<6;i++){ |
| if(marks[semester-1][i]>90){ |
| grade[semester-1][i] = 'O'; |
| sgpa[semester-1] += credit[i]\*10; |
| } |
|  |
| else if(marks[semester-1][i]>80){ |
| grade[semester-1][i] = 'S'; |
| sgpa[semester-1] += credit[i]\*9; |
| } |
|  |
| else if(marks[semester-1][i]>70){ |
| grade[semester-1][i] = 'A'; |
| sgpa[semester-1] += credit[i]\*8; |
| } |
|  |
| else if(marks[semester-1][i]>60){ |
| grade[semester-1][i] = 'B'; |
| sgpa[semester-1] += credit[i]\*7; |
| } |
|  |
| else if(marks[semester-1][i]>50){ |
| grade[semester-1][i] = 'C'; |
| sgpa[semester-1] += credit[i]\*6; |
| } |
|  |
| else if(marks[semester-1][i]>40){ |
| grade[semester-1][i] = 'D'; |
| sgpa[semester-1] += credit[i]\*5; |
| } |
| else{ |
| grade[semester-1][i] = 'F'; |
| sgpa[semester-1] += credit[i]\*5; |
| } |
| } |
| sgpa[semester-1] /= totalCredits; |
|  |
| try{ |
| if(sgpa[semester-1]>10){ |
| throw new InvalidSGPA("Invalid SGPA Exception"); |
| } |
| else{ |
| System.out.println("SGPA in semester"+semester+" = "+sgpa[semester-1]); |
| } |
| } |
| catch(InvalidSGPA ex){ |
| System.out.println(ex.getMessage()); |
| } |
| } |
| } |
| public class CGPA{ |
| public int marks[][] = new int[4][6]; |
| public float sgpa[] = new float[4]; |
| float cgpa =0; |
| public void calculateGpa(){ |
| for(int i=0;i<4;i++){ |
| cgpa += sgpa[i]; |
| } |
| cgpa /=4; |
| System.out.println("CGPA is:"+cgpa); |
| } |
| } |
| package Q11.StudentPackage; |
| import java.util.Scanner; |
| import Q11.ResultPackage.Result; |
|  |
| interface Student{ |
| public void getName(); |
| public void getBranch(); |
| } |
| class CreditLimit extends Exception{ |
| CreditLimit(String s){ |
| super(s); |
| } |
| } |
| public class RegisterStudent extends Result implements Student{ |
| Scanner sc = new Scanner(System.in); |
| public void getName(){ |
| System.out.println("Enter name:\t"); |
| name = sc.nextLine(); |
| } |
| public void getBranch(){ |
| System.out.println("Enter branch:\t"); |
| branch = sc.nextLine(); |
| } |
| public void register(int sem){ |
| semester = sem; |
| if(semester == 1){ |
| getName(); |
| getBranch(); |
| } |
| System.out.println("Enter the 6 subjectss registered"); |
| for(int i=0;i<6;i++){ |
| System.out.println("Enter subjects "+(i+1)); |
| subjects[i] = sc.next(); |
| } |
| System.out.println("Enter the respective credits"); |
| totalCredits =0; |
| for(int i=0;i<6;i++){ |
| System.out.println("Enter the credits of subjects "+(i+1)); |
| credit[i] = sc.nextInt(); |
| totalCredits += credit[i]; |
| } |
| try{ |
| if(totalCredits >30){ |
| throw new CreditLimit("Credit Limit Exception"); |
| } |
| } |
| catch(CreditLimit cl){ |
| System.out.println(cl.getMessage()); |
| } |
| } |
| } |

|  |
| --- |
| package Q11; |
| import Q11.StudentPackage.RegisterStudent; |
| import Q11.StudentPackage.CGPA; |
| class MainClass{ |
| public static void main(String[] args){ |
| RegisterStudent s = new RegisterStudent(); |
| System.out.println("For Semester 1"); |
| s.register(1); |
| s.result(); |
| System.out.println("For Semester 2"); |
| s.register(2); |
| s.result(); |
| System.out.println("For Semester 3"); |
| s.register(3); |
| s.result(); |
| System.out.println("For Semester 4"); |
| s.register(4); |
| s.result(); |
| s.calculateGpa(); |
|  |
| } |
| } |

12.a. Write a java program to implement queues of Strings using an ArrayList class of the Collection framework.

b. Create a linked list of names (String type). Use an Iterator to traverse through the list and print those names whose length is < 5.

|  |
| --- |
| import java.util.\*; |
| class ArrayList1{ |
| public static void main(String[] args){ |
|  |
| Scanner sc = new Scanner(System.in); |
| ArrayList<String> Q = new ArrayList<String>(); |
|  |
| while(true){ |
| System.out.println("1. Enqueue"); |
| System.out.println("2. Dequeue"); |
| System.out.println("3. Peak Element"); |
| System.out.println("4. Display"); |
| System.out.println("5. Exit!"); |
|  |
| System.out.println("Enter your choice"); |
| int ch = sc.nextInt(); |
|  |
| switch(ch){ |
| case 1: |
| System.out.println("Enter the string you wanna Enqueue"); |
| String x = sc.next(); |
| Q.add(Q.size(),x); |
| System.out.println("Element enqueued successfully"); |
| break; |
|  |
| case 2: |
| if(Q.isEmpty()){ |
| System.out.println("The Queue is empty"); |
| } |
| else{ |
| System.out.println("The element dequeued is: "+Q.remove(0)); |
| } |
| break; |
| case 3: |
| if(Q.isEmpty()){ |
| System.out.println("The Queue is empty"); |
| } |
| else{ |
| System.out.println("The element at the peak is :"+Q.get(0)); |
| } |
| break; |
| case 4: |
| if(Q.isEmpty()){ |
| System.out.println("The Queue is empty"); |
| } |
| else{ |
| System.out.println("The elements of the queue are:"); |
| for(int i=0;i<Q.size();i++){ |
| System.out.println(Q.get(i)); |
| } |
| } |
| break; |
| case 5: |
| System.exit(0); |
| break; |
| default: |
| System.out.println("Invalid choice!"); |
| } |
| } |
| } |
| } |

|  |
| --- |
| import java.util.Scanner; |
| import java.util.LinkedList; |
| import java.util.Iterator; |
|  |
| class LinkedList1{ |
| public static void main(String[] args){ |
| Scanner sc = new Scanner(System.in); |
| LinkedList<String> ll = new LinkedList<String>(); |
|  |
| while(true){ |
| System.out.println("1. Insert an element into the linked list"); |
| System.out.println("2. Remove an element from the linked list"); |
| System.out.println("3. List all elements with length less than 5"); |
| System.out.println("4. Display the LinkedList"); |
| System.out.println("5. Exit!"); |
|  |
| System.out.print("Enter your choice:\t"); |
| int ch = sc.nextInt(); |
|  |
| switch(ch){ |
| case 1: |
| System.out.println("Enter the element to be inserted"); |
| String x = sc.next(); |
| ll.add(x); |
| System.out.println("Element inserted successfully"); |
| break; |
| case 2: |
| if(ll.peek() == null){ |
| System.out.println("The linked list is empty"); |
| } |
| else{ |
| System.out.println("Enter the index of the element you wanna remove"); |
| int p = sc.nextInt(); |
| System.out.println("The elemnet removed is: "+ll.remove(p)); |
| } |
| break; |
| case 3: |
| if(ll.peek() == null){ |
| System.out.println("The Linked list is empty"); |
| } |
| else{ |
| Iterator<String> itr = ll.iterator(); |
| String e1 = ""; |
| System.out.println("The strings with length less than 5 are: "); |
| while(itr.hasNext()){ |
| e1 = itr.next(); |
| if(e1.length()<5){ |
| System.out.println(e1); |
| } |
| } |
| } |
| break; |
| case 4: |
| if(ll.peek() == null){ |
| System.out.println("The linked list is empty"); |
| } |
| else{ |
| Iterator<String> itr = ll.iterator(); |
| System.out.println("The elements of the linked list are:"); |
| while(itr.hasNext()){ |
| System.out.println(itr.next()); |
| } |
| } |
| break; |
| case 5: |
| System.exit(0); |
| break; |
| default: |
| System.out.println("Invalid choice"); |
|  |
| } |
| } |
| } |
| } |