List of Experiments

The students are required to develop and execute the following programs in Java.

TW 1: Write a program to demonstrate the implementation of 2-dimension array.

Write a Java program to accept IA marks obtained by five students in three subjects. The program should accept marks obtained by each student and display the total marks and the average marks. The average marks is computed using a method as the average of best two marks obtained.

Additional programs to be implemented:

1.1) It is required to store and analyze data about 6 car manufacturer's sales data in all the 12 months of a year. Demonstrate how you would store the data in a two dimensional matrix and do the following

1. Write a function to Find for a given car manufacturer, the month in which, maximum no. of cars are sold.
2. Write a function to Find the average number of cars sold for each car manufacturer
3. Write a function to Find the total number of cars sold for each car manufacturer.
4. Write a function to find standard deviation for a given car manufacturer

Assume – row index 0 - 'Maruti Suzuki', 1 – 'Hundai' 2 – 'Tata Motors' 3-'KIA' 4 – 'BMW'   5 – 'Renault'

Col index 0 –'Jan', 1-'Feb'………………………………….11 –'Dec'

Demonstrate the working of the program with appropriate values for each car manufacturer and the months.

1.2) A joint family consisting of 10 households lives in the same compound. Due to mounting electricity bills, the head (Mr. X) of the joint family decides to analyze the consumption pattern (in terms of the billed amount) of each household for a year.   Mr. X needs access to the following information for his analysis:

1) The total expenditure on electricity consumption by each household in a year.
2) The maximum and minimum electricity consumption of each household in a year.
3) The amount by which each household exceeded the average consumption (+/-) of all households in the month of June.
4) The maximum, minimum and average electricity consumption of all households in a year.

Demonstrate how you would use a two dimensional matrix to help Mr. X.

1.3) Write a Java program to find the average score of five students in three papers.

Solution Approach:

Given, score of first student is 60, 55 and 70 while score of the second student is 80, 60 and 41. In the same way collect the score of another three students. We can store the score of the two students in a 2D array having 5 rows and 3 columns. The rows will represent the student and the columns will hold the score of the students.

1. Create a Score two dimensional array
2. Create a sum array
3. Create an average array
4. Compute sum
5. Compute average
6. Print the result

1.4) A company has 10 zonal sales offices in four zones namely, North, East, West and South. The company wants to organize the sales data of each of the office in each zone and find answers to queries such as,
1. Which office has performed the highest sales in each zone?
2. What is the average sales done by all the offices in each zone?
3. Which office among each zone is the poorly office?

You are required to answer the following:

1. How do you organize the above data?
2. How do you provide answers to the above queries?

Design a Java application for the same and demonstrate the correctness of the solution.

TW 2: Write a program to demonstrate the implementation of class and its member methods.

Design a class by name myTriangle to model a triangle geometrical object with three sides. Include functions to:

- Initialize the three sides of triangle.
- Determine the type of triangle represented by the three sides (Equilateral/ Isosceles/ Scalene triangle).
- Compute and return the area of the triangle.

Note:

When three sides are given we use the following formula:

s=(a+b+c)/2;

area=sqrt(s*(s-a)*(s-b)*(s-c));

Additional programs:

2.1) Design a class by name myCircle to model Circle geometrical object with its center and radius that enables:

1. Initializing the center, radius and
2. Compute area, perimeter, and diameter of the circle object/s.

TASK 1: Identify member variable/s and their types

TASK 2: Identify Constructor/s along with their arguments (if any) to initialize the member variables

TASK 3: Identify the methods along with their arguments and return types.

TASK 4: Identify member variable getters/setters (if needed)

2.2) Define a class to represent the student details such as name, roll number, marks obtained in three internal assessment tests.
a) Identify type and declare the instance variables
b) Identify and develop the constructors to initialize the instance variables
c) Write a method computeAverage() to compute the average as the average of best two marks
d) Write a method to display the student details

Write the corresponding Driver class to instantiate an array of student objects and demonstrate the working of the application by invoking appropriate methods.

2.3) Write a Java program to define a class Lamp. It can be in on or off state. You can turn on and turn off lamp (behavior). It makes use of class and its member methods.

Solution Approach:

Lamp class is created. The class has an instance variable isOn and three methods turnOn(), turnOff() and displayLightStatus().

Two objects l1 and l2 of Lamp class are created in the main() function.

Here, turnOn() method is called using l1 object: l1.turnOn();

This method sets isOn instance variable of l1 object to true.

And, turnOff() method is called using l2 object: l2.turnOff();

This method sets isOff instance variable of l2 object to false.

Finally, l1.displayLightStatus(); statement displays Light on? true because isOn variable holds true for l1 object.

And, l2.displayLightStatus(); statement displays Light on? false because isOn variable holds false for l2 object

3.   Write a program to demonstrate the implementation of parameterized:

     a. Methods.

     b. Constructor.

3.1) A certain small bank intends to automate few of its banking operations for its customers. Design a class by name mybankAccount to store the customer data having following details:

     1.accountNumber 2. acctType 3. Name 4. Address 5. accountBalance

The class must have both default and parameterized constructors. Write appropriate method to compute interest accrued on accountBalance based on accountType and time in years. Assume 5% for S/B account 6.5% for RD account and 7.65 for FD account. Further, add two methods withdrawAmount/depositAmount with amount as input to withdraw and deposit respectively. The withdrawAmount method must report in-sufficient balance if accountBalance falls below Rs. 1000.

TASK 1 : Build the class with appropriate member variables, constructors and methods.

TASK 2 : Instantiate three objects of above type and perform different operations on the same.

TASK 3 : Write a function to display all the three customer details in a tabular form with appropriate column headings.

Additional programs:

3.2) Define a class to represent a rectangle in which constructors and parameterized methods are used. It also has a method to compute area of rectangle.
  i.   First make a class rectangle in which we declare the parameterized constructor.
  ii. Then demonstrate the use of parameterized method.
  iii. Use a method to compute area of rectangle.
  iv. Create a class to demonstrate the call of the methods in previously created class rectangle holding constructors, parameterized methods and method to compute area of rectangle.

3.3) Write a Java program to represent a Complex number. Include member functions to:

  1. Initialize a complex number to a default value of zero (default constructor)
  2. Initialize a complex number to a user defined value (parameterized constructor)
  3. Add two complex numbers and return the result. (Parameterized method)
  4. Subtract two complex numbers and return the result. (Parameterized method)
  5. Display a complex number. (non-parameterized method)

3.4) Create a IndMoney class with two integer instance variables rupees and paise. Add a constructor with two parameters for initializing a IndMoney object. The constructor should check the paise value is between 0 and 99 and, if not, transfer some of the paise to the rupees variable to make it between 0 and 99. Add a plus method to the class that takes a IndMoney object as parameter. It creates and returns a new IndMoney object representing the sum of the object whose plus() method is being invoked and the parameter. It does not modify the values of the two existing objects. It should also ensure that the value of the paise instance variable of the new object is between 0 and 99. For example, if x is an IndMoney object with 12 rupees and 80 paise, and if y is an IndMoney object with 8 rupees and 90 paise, then x.plus(y) will return a new IndMoney object with 21 rupees and 70 paise. Also, create a IndMoneyDemo driver class that tests the IndMoney class.

4.   Write a program to demonstrate the implementation of inheritance.

4.1) A company has two types of employees – FullTime and Partime. The company records for each employee his/her name, age, address, salary and gender. Given the basic salary of the FullTime employee the components of his/her gross salary are: Dearness allowance – 75% of basic salary, HRA – 7.5% of basic salary, IT – 10% of basic. The salary of a Partime employee is dependent on the qualification, experience, number of working hours and the rate per hour, as below:

	Qualification		
Experience	BE	MTech	Ph.D
1-5 years	300 Rs.	500 Rs.	800 Rs.
6-10 years	400 Rs.	700 Rs.	1200 Rs.
>10 years	500 Rs.	1000 Rs.	1500 Rs.

Model this as a problem of hierarchical inheritance by:

1) Identifying the super class with its data members and member functions.

2) Identify the sub-class/sub-classes and their associated data members and member functions.

Test the program by creating objects of the classes that are so identified.

Additional definitions:

4.2)   Design a base class called Employee who work for a Hospital and this class would have name, dob, address, salary and designation as the attributes. Add a constructor to initialize all these data members. This class would have reportForDuty method to display reporting time and date with a "Welcome" message to the employee. Devise two derived classes Doctor and Nurse to have expertise and yearsofExperience as data members respectively. Devise a method performDuty in each of these derived classes to print appropriate message depending on expertise of doctor and years of experience. For instance if the experitise of the Doctor is Surgeon and yearsofExperience >10 then print "Perform Heart Operation" else perform "Perform minor Surgery". If his expertise is orthopedic and experience is more than 5 years "Perform surgery and Plastering" else "Perform Plastering". If the nurse has more than 3 years experience in performDuty method print "Check BP, Sugar" and "Administer medicine" else print "Look after the patient". Create a Hospital Class that has main method, and instantiate objects of Doctor and Nurse and Perform reportForDuty and performDuty and record the output. Add a static method generateReport(Employee e) that accepts object of employee type and prints in a tabular form, Name, dob, birthday salary and the designation.

Note :

To get current time and date create an object of Calendar class declared in java.util package by importing it… import java.util.Calendar. to create calendar object..

Calendar cal=Calendar.getInstance();

To print date and time System.out.println(cal.getTime());

4.3)   The class Cylinder inherits all the instance variables (radius and color) and methods (getRadius(), getArea(), among others) from its superclass Circle. It further defines a variable called height,   three   methods   getHeight(),   setHeight() and getVolume() and   its   own constructors. Implement the hierarchy as shown below:
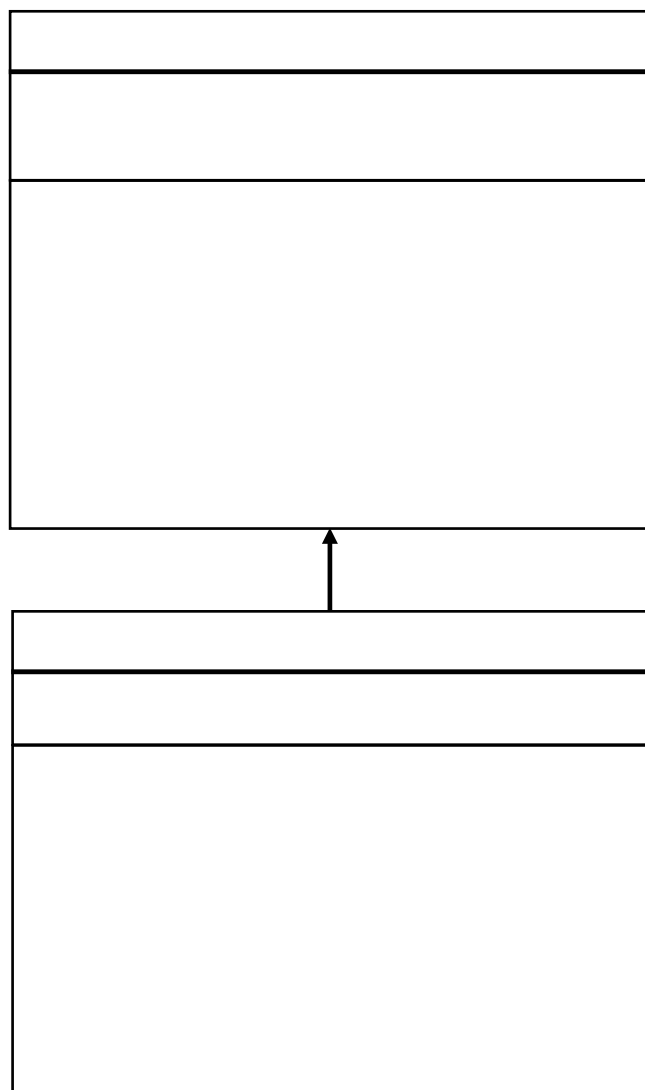
4.4) Implement a BANK class to demonstrate the inheritance in Java by implementing getRateOfInterestmember function for three different banks, as shown below.

Bank
getRateOfInterest() : float

extends

SBI		ICICI		AXIS
getRateOfInterest() : float		getRateOfInterest() : float		getRateOfInterest() : float
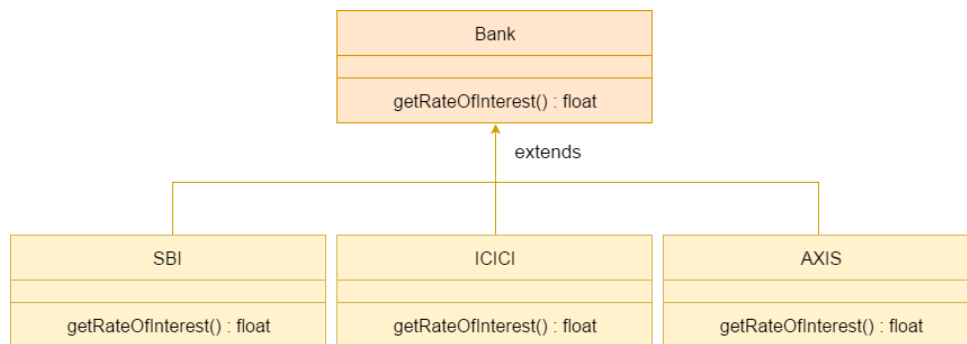
5a) Write a program to demonstrate the implementation of method overloading.

5.1) Create a Stack class having an integer array say elem and top_of_stack as instance variables. Define three overloaded methods having the following signatures:
   a. initStack(int size) to create an array of specified size and initialize the top_of_stack
   b. initStack(Stack another) to intialize the Stack object with state of the Stack object "another"
   c. initStack(int [] a) to initialize contents of a[] to the instance variable elem.

Write following methods:
       a. push(): Pushes the element onto the stack,
       b. pop(): Returns the element on the top of the stack, removing it in the process, and
       c. peek(): Returns the element on the top of the stack, but does not remove it.

Also write methods that check whether stack is full and stack is empty and return boolean value true or false appropriately.

Additional problem definitions:

5.2)   Implement a linear search function by using method overloading concept for an array of integers, double and character elements.

5.3)   Design a Vehicle class to include as data members, the vehicle's initial velocity (u), final velocity (v) and acceleration (a). The class must a parameterized constructor to initialize these data members**.**

   Design a class LawsofMotion and write two overloaded methods, **computeDistanceTravelled** and **computeAcceleration** to return the distance travelled by the vehicle and acceleration attained respectively, given the different parameter values. The class must have appropriate data members for the following methods to work.

   computeDistanceTravelled(float t)- computes and returns the distance travelled as: $u*t+0.5f*a*t*t$

   computeDistanceTravelled() - computes and returns the distance travelled as (($v*v$ $-u*u)/(2*a)$);

   computeAcceleration(float mass, float force) - computes and returns the acceleration as $a=mass/force$;

   computeAcceleration(float mass, float v, float u,float t) - computes and returns the acceleration as $(m*v-m*u)/t$;

Demonstrate the working by instantiating objects of the above class and verify the working of the overloaded methods.

5.4) Implement a sort function that illustrates overloading methods. Sort method sorts the array in the default ordering, sorts the array into the specified order, sorts array elements ranging from fromIndex to toIndex in the specified order.

5 b) Overriding.

5 b.1)Implement the following class hierarchy. In the Cuboid class, override the method computeArea() and computePerimeter() of Rectangle class to compute the surface area and perimeter of a rectangle cuboid. Add a method computeVolume() in Cuboid class to compute volume of the cuboid. Assuming length, width and height as l, w and h respectively,

- formula to find the surface area = 2(lw) + 2(hl) + 2(hw)
- formula to find the perimeter = 2l + 2w
- formula to find the volume = l x w x h
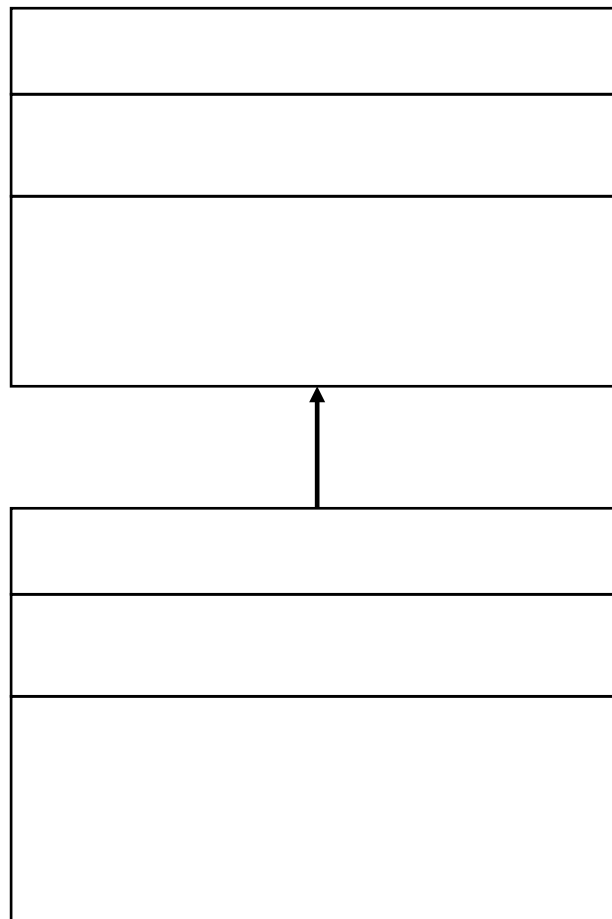
5 b. 2) Design a base class ArrayStack that uses array to hold the elements and has 3 methods namely, push, pop and display. Derive a class LinkedStack that overrides these 3 methods and uses linked list to implement stack. Demonstrate the working of both the classes by performing push, pop and display operations on the objects of the above to classes.

5b.3) Design a base class where Hospital provides no. of patients admitted in it. Number of patients varies with the different hospitals, For example Health India hospital has 1657 patients, IVY hospital has 2965 patients and Apollo Hospital has 1631 patients. Hospital parent class which has one method getNumberOfPatients() and sub class HealthIndia, IVY and Apolo class which extends parent class & override its method.

6.

Write a program to demonstrate Run-time Polymorphism.

6.1) Design an abstract class Car to have carName, chassiNum, modelName as member variables and add two abstract methods, startCar and operateSteering . Inherit MarutiCar and BmwCar from Car class and override the two abstract methods in their own unique way. Design a driver class to have driver name, gender and age as data members and add a method driveCar with abstract class reference variable as argument and invoke the two basic operations namely, startCar and operateStearing and demonstrate run-time polymorphism.

Additional definitions:

6.2) Implement the following inheritance hierarchy.
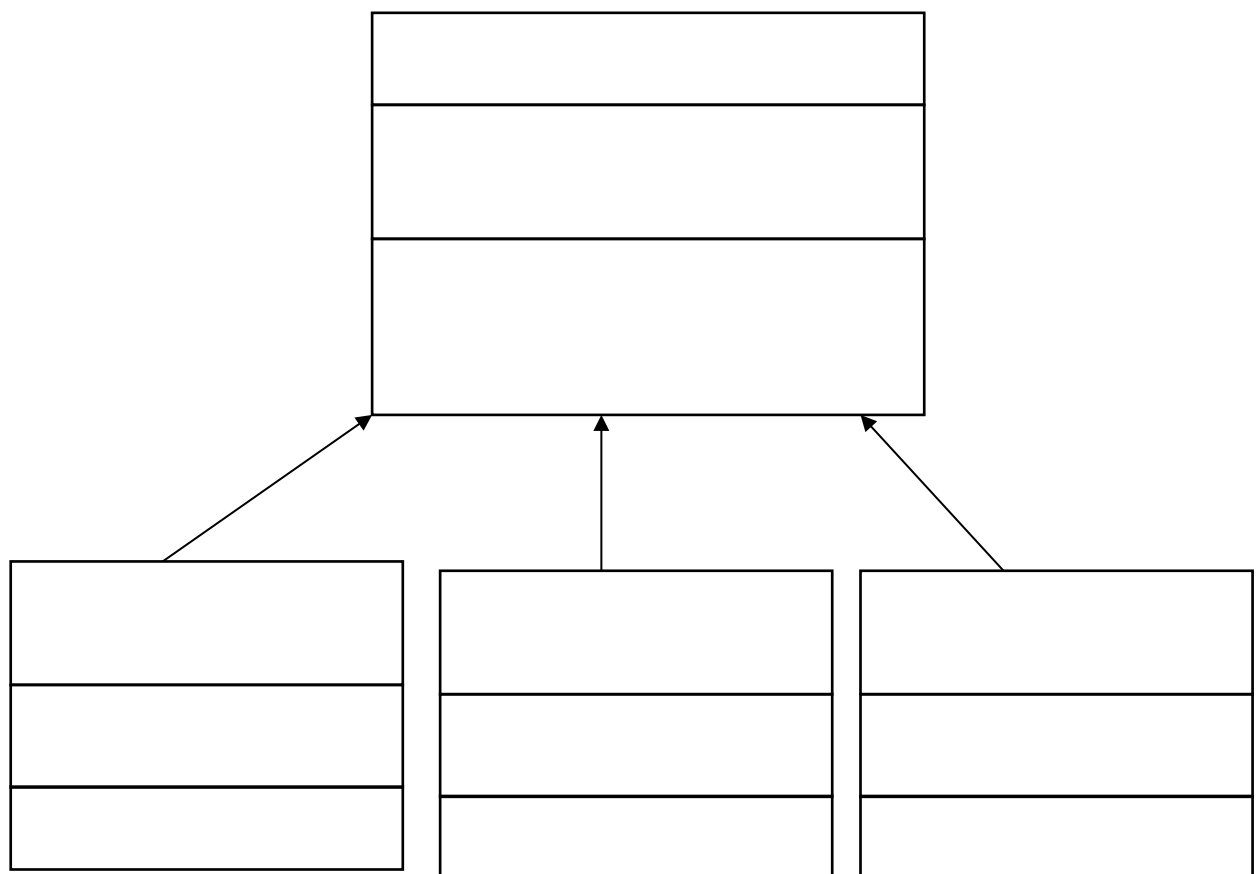
6.3) Write a Java program that defines an abstract class called Account and accepts the following customer account information:

1) Customer Name
2) Account Number
3) Balance

and provides below operations on customer account:

1) Deposit

2) Withdraw
3) Display Balance
4) Display full account details

There are two types of accounts – Savings and Current. The Current account has an overdraft facility limited to Rs. 75,000 per account. The following constraints hold on these accounts:

Savings Account:

1) The total number of deposits for a Savings account cannot exceed three per day.
2) The amount deposited into a savings account cannot exceed Rs.5000 in each transaction.
3) The maximum withdrawal amount is Rs.25,000 per transaction.
4) The minimum balance to be maintained is Rs. 10,000.

Current account:

1) The amount withdrawn cannot exceed the overdraft limit once the account balance is zero.
2) Maximum number of withdrawals is two.
3) No limit on the number of deposits.
4) Each deposit cannot exceed Rs. 25,000.

Test the program by creating objects of the Savings and Current Accounts.

6.4) Write a Java program that defines an abstract class called Employee and accepts the following Employee information: Employee name, address and number. It computes pay and mails a check to employee.

7.

Write a program to demonstrate the implementation of interfaces.

7.1) Write a Java application to implement the following UML diagram.

- PrimeTester class implements isPrime() method by iterating from 2 to n-1 for a given number n
- ImprPrimeTester class implements isPrime() method by iterating from 2 to n/2
- FasterPrimeTester class implements isPrime() method by iterating from 2 to $\sqrt{n}\sqrt{n}$
- FastestPrimeTester class implements isPrime() method using Fermat's Little theorem.
  - Fermat's Little Theorem:
  - If n is a prime number, then for every a, $1 < a < n-1$,
    - $a^{n-1} \% n = 1$
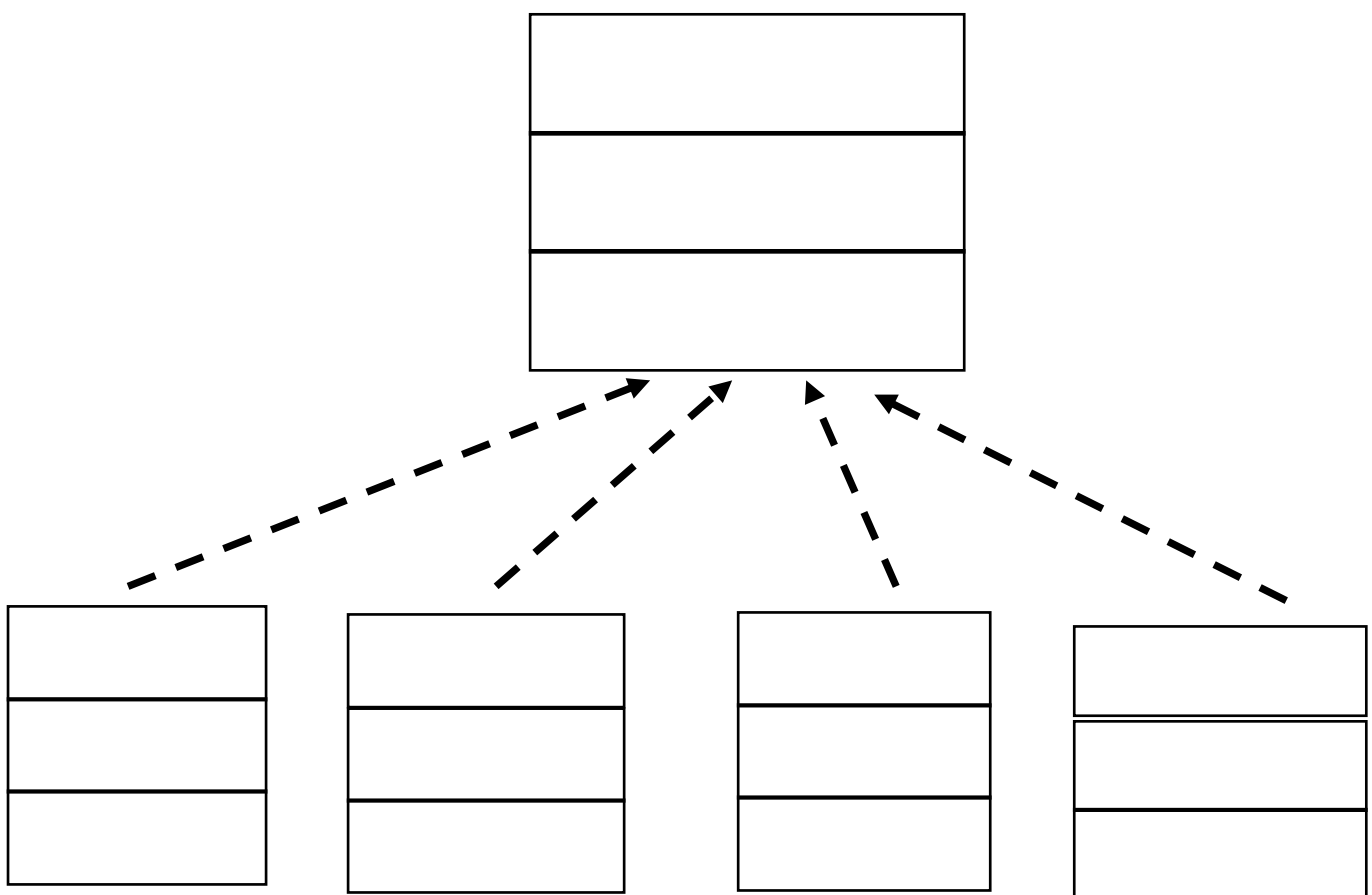
Additional problem definitions:

 7.2) Write a JAVA program which has

i. An Interface class for Stack Operations (viz., push(), pop(), peek(),display())
ii. A Class that implements the Stack Interface and creates a fixed length Stack.
iii. A Class that implements the Stack Interface and creates a Dynamic Length Stack.
iv. A Class that uses both the above Stacks through Interface reference and does the Stack operations that demonstrates the runtime binding.

**7.3)** Design an interface IAnimal that has walk, and sleep methods, an interface IBird that has fly, and peck methods, an interface IHuman that has eat and speak methods. Construct a Bird class that implements IAnimal and IBird interfaces and also construct Human class that implements IAnimal and IHuman interfaces. Demonstrate the working of these methods by invoking the methods using appropriate reference variables.
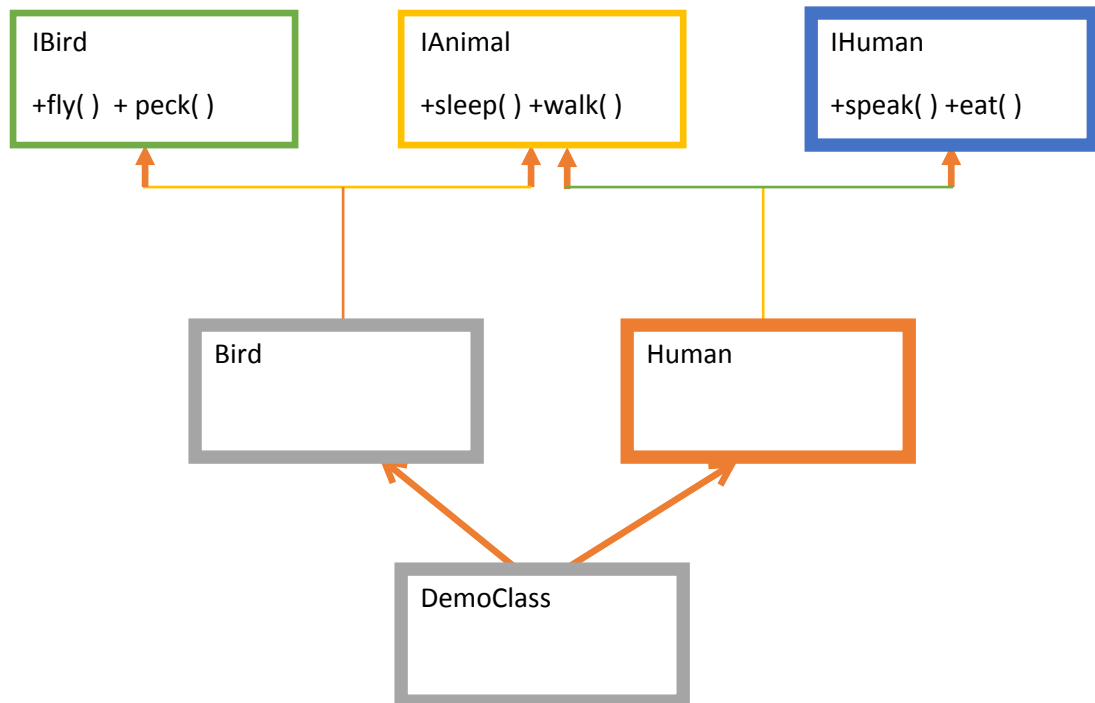
8. Write a program to demonstrate the implementation of **customized exception handling.**

8.1) Assume that you have received a request from the transport authority for automating the task of issuing the permanent license for two wheelers. The mandatory condition to issue the license are: 1) the applicant must over 18 years of age and 2) holder of a valid learner's license and 3) no accident cases in the last one year.

Write a Java program that reads user details as required (use the Scanner class). Create user defined exceptions to check for the three conditions imposed by the transport authority. Based on the inputs entered by the user, decide and display whether or not a license has to be issued or an error message as defined by the user exception.

License
name: String age: int gender : String valid_LL : char/String no_of_cases : int
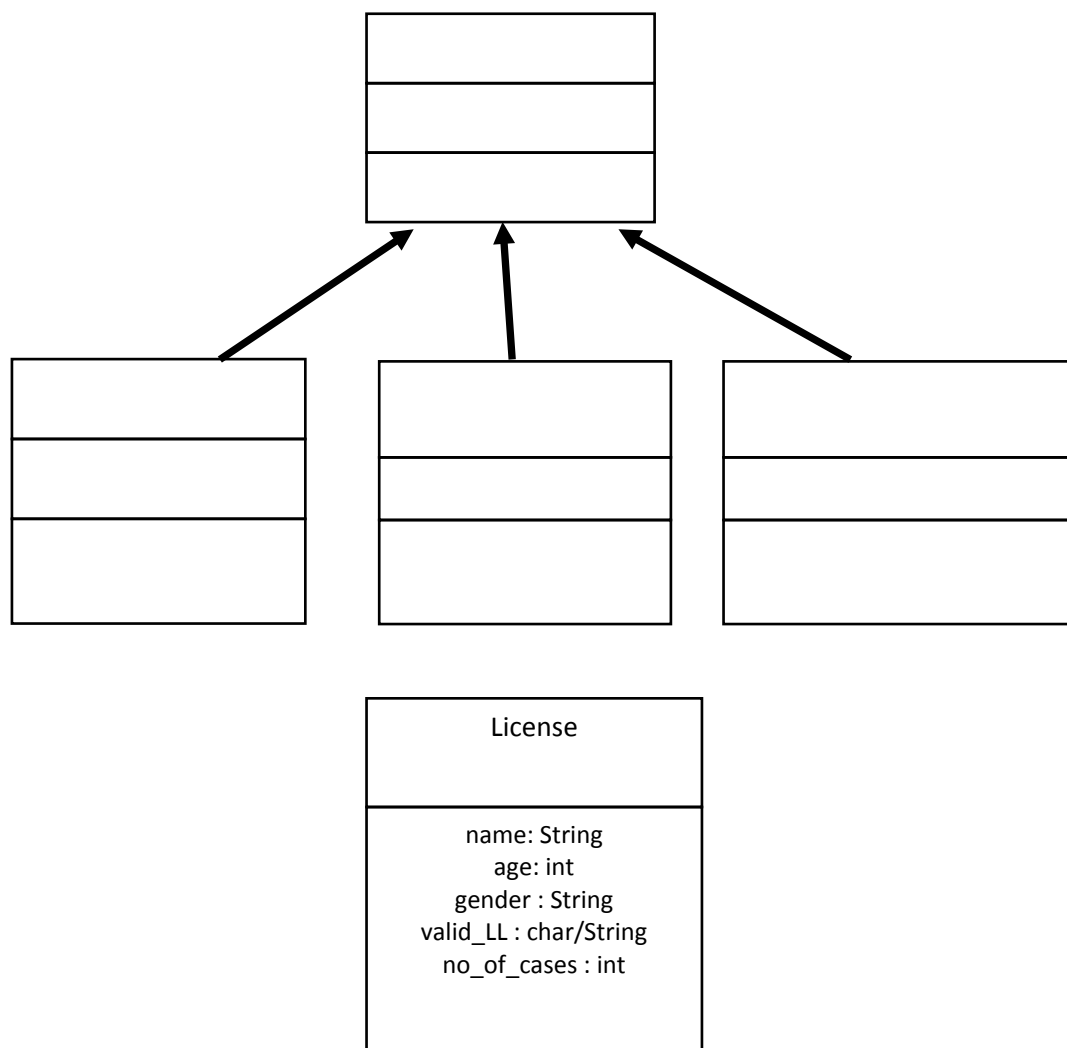
Additional problem definitions:

8.2) Design a class CustomerAccount that has acctNum, custName and balance as member variables and a constructor to initialize these. Implement withDraw and depositAmount methods that accepts amount as argument and it must throw an user defined exception called InsufficientBalalance/InvalidAmount exception when amount is greater than balance/ amount is negative respectively. Design two classes InsufficientBalance and InvalidAmount that extend the Exception class and override toString method. Demonstrate the working of the user defined exceptions by instantiating an object of customerAccount class and invoking withDraw and depoistAmount in try… catch.. finally block.

```
+---------------+
|   Exception   |
+---------------+
        ^
        |
   +----+----+
   |         |
```

```
+---------------------+        +---------------------+
| InvalidAmount       |        | InsufficientBalance |
|                     |        |                     |
| +getMessage( )      |        | +getMessage()       |
+---------------------+        +---------------------+
```

```
+--------------------------------------------------------+
| CustomerAccount                                        |
|                                                        |
| -accNum      -balance    -custName                     |
|                                                        |
| +withDraw(float amount ) throws InsufficientBalance    |
|                                                        |
| +depositAmount(float amount ) throws InvalidAmount     |
+--------------------------------------------------------+
```

```
DemoClass  { public static void main(String[] args){

CustomerAccount  newCustomer=new CustomerAccount( );

try{   newCustomer.withDraw(7000);

     newCustomer.depositAmount(-500);

}catch(InsufficientBalance e) { e.getMessage( );}

catch(InvalidAmount e) { e.getMessage( ); } finally{ println("Transaction failed…"); }

} }
```

8.3) Design a class to implement Stack data structure. The method push() throws a custom exception called OverflowException if the item cannot by pushed on to the stack. The method pop() throws a custom exception called UnderflowException if the stack is empty. Design two classes OverflowException and UnderflowException that extend the Exception class and override the getMessage() method. Develop a driver class to demonstrate the working of the custom exceptions by instantiating an object of Stack class and performing push() and pop() operations.
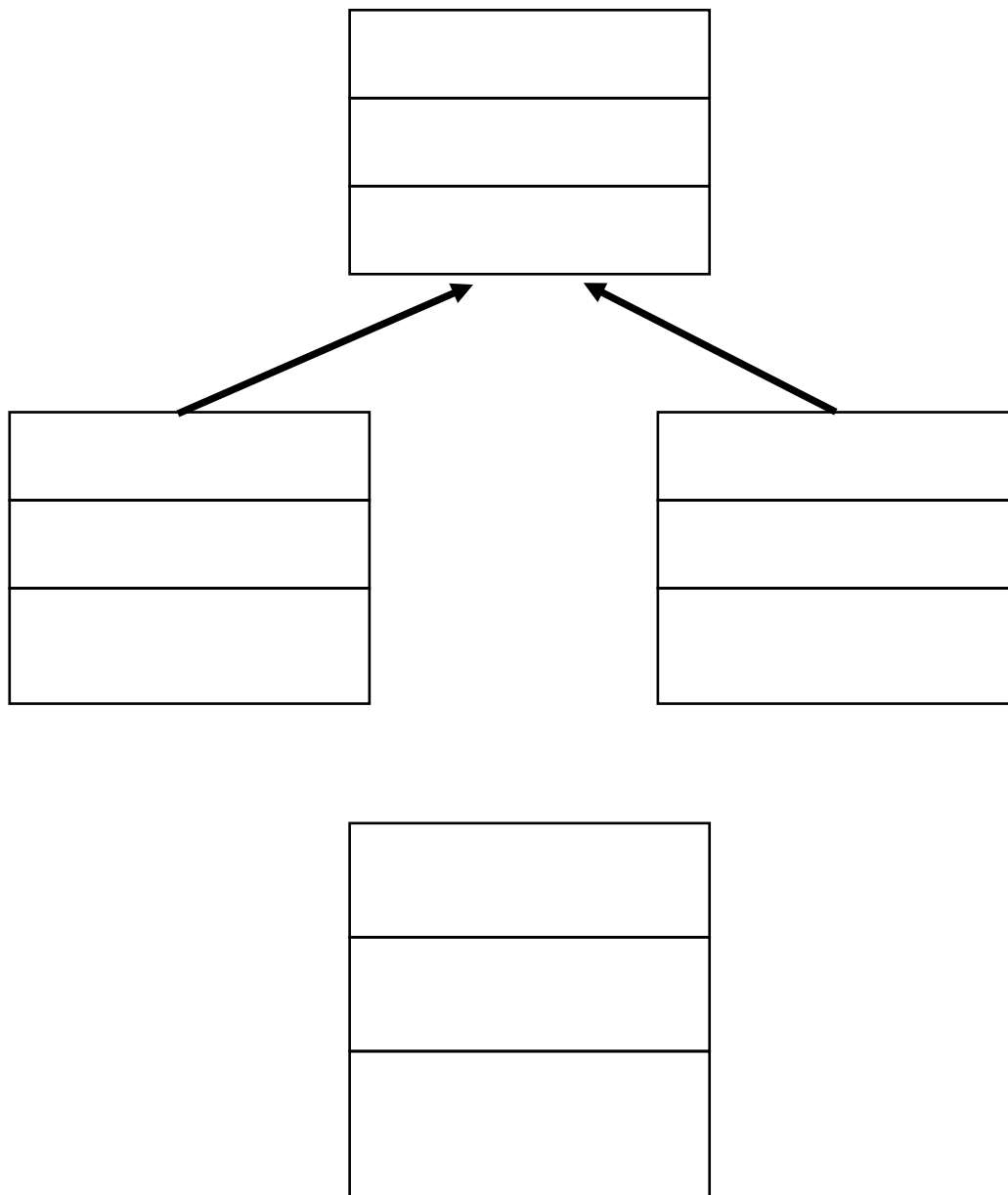
8.4).Write java program that takes the value of num variable and checks it is odd, then the throw keyword will raise the user defined exception and the catch block will get execute.

OddNumberException class is derived from the Exception class. To implement user defined exception throw an exception object explicitly.

9. **Write a program to demonstrate the implementation of string handling.**

9.1) Read a string containing 3_4 words using Scanner class object. Split it into words and for each word check if it's palindrome by writing a function isPalindrome(String the myWord, int s, int e) which return true if its palindrome else return false. Where s is start index and e is end index of the input myWord. Print it in uppercase if it is palindrome else reverse the string and print it in lowercase. Use appropriate string functions to implement the above problem statement.
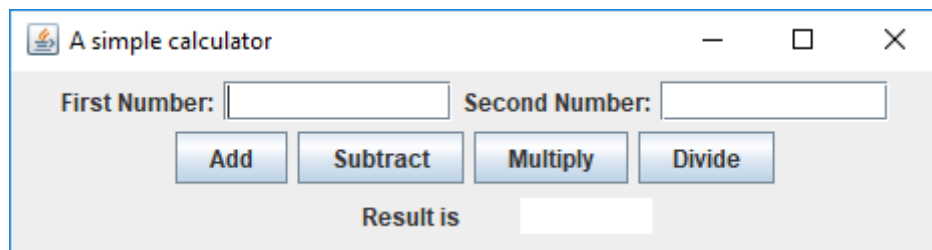
Additional problem definitions:

9.2) Two strings will be anagram to each other if and only if they contain the same number of characters (order of the characters doesn't matter). That is, If the two strings are anagram to each other, then one string can be rearranged to form the other string. For Example: creative and reactive are anagrams. Write a Java program to test whether two strings are anagrams or not. (listen and silent, stressed and desserts, dusty and study)

9.3) Write a Java program that creates a simple book database (use an array of N objects). Each book is represented with a ID, title, author (First Name & last name), Genre (category – technical, Sci Fi, Fiction, Comedy etc) and a Publisher name. Define methods to perform the following tasks:
1. Given a title, returns a status to indicate whether or not the book exists in database.
2. Given a string "str", lists the details of all the books whose title contains str.
3. Given a genre, lists publishers who have published books in that genre.
4. Given a character "c", lists all authors who name starts with "c".

10. Write a program to demonstrate the implementation of JAVA swings.

10.1) Design and develop a GUI application as shown below. Assume the two numbers to be integers. The application must check for invalid division condition and throw an appropriate exception.

Additional program definitions

10.2) Design and develop a GUI application to accepts student details as shown in the GUI below. Allow the user to add records one after the other. Once the user clicks on close, compute the average score and display the details using JTable component.
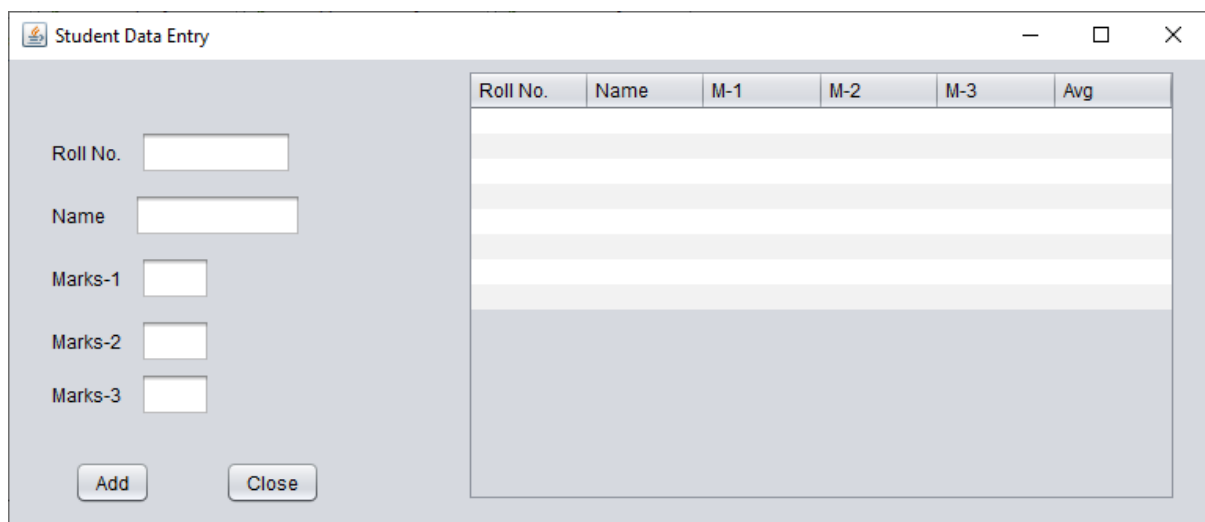
Rename close as "Summarize"

10.3) Design and implement a Home loan Emi calculator using appropriate Swing components. The GUI should like as under:

The formula to compute Home loan EMI for a given Principal amount PA and interest rate IR for a period of T years is

$$EMI = (PA + (PA * IR * T))/12*T ;$$

Hint : Use SlideBars for Amount and Loan Period   LoanType - ComboBox

Course Outcome (Cos)

At the end of the course, the student will be able to:	Bloom's Level
1. Use the NetBeans IDE to write and execute Java programs.	L3
2. Write Java application programs using OOP principles and proper program structuring.	L3
3. Identify classes, members of a class and relationships among them needed for a specific problem	L2
3 Write Java programs to demonstrate error handling techniques using exception handling.	L3
4 Write Java programs to demonstrate packages and interfaces and String	L3

handling.

| 5 | Use Swing concept to develop simple GUI applications. | **L3** |

Program Outcome of this course (POs)   **PO No.**

1. **Engineering knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.   **PO1**

2. **Design/development of solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.   **PO3**

3. **Modern tool usage**: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.   **PO5**

4. **Life-long learning**: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.   **PO12**

Assessment Methods

1. Regular Journal Evaluation and Attendance Monitoring.
2. Lab Internal Assessment.

Components	Conduct of the Lab	Journal Submission	Total Marks
Maximum Marks: 25	10	10	25
Submission and Certification of lab journal is compulsory to qualify for SEE. Minimum marks required to qualify for SEE: 13			

Scheme of Semester End Examination (SEE):

1. It will be considered for 50 marks of 3 hours duration. **It will be reduced to 25 marks for the calculation of SGPA and CGPA.**
2. **Minimum marks required in SEE to pass: 40%**.
   Initial write up   10 Marks
   Conduct of Experiments   20 Marks   50 Marks
   Viva-voce   20 Marks