**BITS, Pilani - Hyderabad Campus**

**Object-Oriented Programming (CS F213)**

**Extra Practice Problems**

**Topics: Interfaces and Packages**

1. Create an interface with a default method that prints "This is default interface" and abstract method to find the factors of a number. Create a class named even that implements interface. Create another class named odd that implements same interface. Now, create an object for each of the class. If the input number is even then call even class to find the factors of a number else call the odd class to find the factors of a number.
2. Create an interface named 'Shape' with an abstract method to display the number of sides in a shape. Then create two other classes named 'Rectangle', 'Circle' implementing the Shape interface, both having a method to print "This is rectangular shape" and "This is circular shape" respectively. There is no relation between two classes. If an object created by Rectangle class is called then the number of shapes is equal to 4 and if an object created by Circle class is called then the it should be print “No sides for circle”.
3. A class called circle is designed with Two private instance variables: radius (of the type double) and color (of the type String), with default value of 1.0 and "red", respectively. Two overloaded constructors - a default constructor with no argument, and a constructor which takes a double argument for radius in one package.

Import first package and create two public methods: getRadius() and getArea(), which return the radius (given in first package) and area of this instance, respectively.

Can you get radius (Private variable) value from one package to another?

1. Create a class called Employee. This class will implement an interface person which is having abstract method to compute raise\_salary. An employee record has an employee's name (inherited from the class Person), an annual salary represented as a single value of type double, a year the employee started work as a single value of type int.

IF employee joins 10 years before 2020 then raise salary by 10%

IF employee joins 15 years before 2020 then raise salary by 15%

Display the final salary, name and year.

1. Using the Account class as a base class, write two derived classes called Savings Account and Current Account in two different packages. Account class should have attributes like account balance, loan (yes or no Boolean variable). A Savings Account object, in addition to the attributes of an Account object, should have an interest variable and a method which adds interest (assume your own interest) to the account if the account is associated with loan. A Current Account object, in addition to the attributes of an Account object, should have an overdraft limit variable to display if the account balance is less than 1000. Compute the interest and display the initial amount, loan amount, amount after deducting loan amount and check whether final balance is less than 1000 or not.
2. Create an interface for opening a bank account with necessary details using Detail abstract method. Add two bank account. Create a class to delete account details only if the account number matches. Take the account number from user and match with already existing two account numbers. If matches display account removed else no account matched.

Is it mandatory to have either abstract or default keyword before a method in interface?

1. Write a program to extend the Vehicle interface. The interface has two abstract method to get and display the information about vehicles, including the number of passengers they can carry, their fuel capacity, and fuel consumption rate. We can use the abstract method in vehicle interface as a starting point from which more specialized classes are developed. For example, one type of vehicle is a truck. An important attribute of a truck is its cargo capacity. Thus, to create a Truck class, you can implement abstract method in Vehicle interface, adding a private instance variable that stores the carrying capacity.
2. Create class SavingsAccount. Use a static variable annualInterestRate to store the annual interest rate for all account holders. Each object of the class contains an instance variable savingsBalance indicating the amount the saver currently has on deposit in one package.

Provide method calculateMonthlyInterest in another package to calculate the monthly interest by multiplying the savingsBalance by annualInterestRate divided by 12 this interest should be added to savingsBalance in another package.

Write a program to test class SavingsAccount. Instantiate two savingsAccount objects, saver1 and saver2, with balances of $2000.00 and $3000.00, respectively. Set annualInterestRate to 4% in package 1. Then calculate the monthly interest and print the new balances for both savers using package 2.

Then set the annualInterestRate to 5%, calculate the next month’s interest and print the new balances for both savers.

1. Create an iterface called Date that includes three pieces of information in default method as Dateformat—a month (typeint), a day (typeint) and a year (typeint). Create checkdate which follows same date format given in default method. Check, month should be less than equal to 12, year should be four digit number and less than 2020, day should be less than 31. Create another class daycheck which inherits checkdate. In this check whether the given day is in month or not (jan 31 days, feb either 28 or 29, march 31… and so on). Finally print whether the date given is correct or not.
2. Create a class called Employee that includes three pieces of information as instance variables—a first name (typeString), a last name (typeString) and a monthly salary (double) in one package.

Inherit Employee in another package and check whether the monthly salary is positive or not, set it to 0.0 if the monthly salary is negative. Create another class which inherits employee to give each Employee a 10% raise.

Write a test application named EmployeeTest that demonstrates class Employee’s capabilities. Create two Employee objects and display each object’s yearly salary before and after increment.