[Abstract Classes](http://59.162.104.101:25000/PBLApp/tmodule.action?module=Abstraction%20/Packages%20/%20Exception%20Handling&stream=Java&tm=TM3" \l "T1)

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **Hands-on Assignment** | **Topics Covered** | **Status** |
| 1 | 1.1. Create a class called GeneralBank which acts as base class for all banks. This class has functionality getSavingInterestRate and getFixedInterestRate methods, which return the saving a/c rate of interest and fixed account rate of interest the specific bank gives. Since GeneralBank cannot say what percentage which bank would give, make it abstract.  1.2. Create 2 subclasses of GeneralBank called ICICIBank and KotMBank. Override the methods from base class. ICICI - Savings 4% Fixed 8.5% and KotMBank. - Savings 6% Fixed 9%  1.3. Create a main method to test the above classes. Try one by one and absorb your finding.  a) ICICIBank object reference instantiated with ICICIBank class.  b) KotMBank object reference instantiated with KotMBank class.  c) GeneralBank object reference instantiated with KotMBank class.  d) GeneralBank object reference instantiated with ICICIBank class. | Abstract Classes |  |
| 2 | Create an abstract class Compartment to represent a rail coach. Provide an abstract function notice in this class. Derive FirstClass, Ladies, General, Luggage classes from the compartment class. Override the notice function in each of them to print notice suitable to the type of the compartment.  Create a class TestCompartment . Write main function to do the following:  Declare an array of Compartment of size 10.  Create a compartment of a type as decided by a randomly generated integer in the range 1 to 4.  Check the polymorphic behavior of the notice method. | Abstract Classes |  |
| 3 | Create an abstract class Instrument which is having the abstract function play.  Create three more sub classes from Instrument which is Piano, Flute, Guitar. Override the play method inside all three classes printing a message  “Piano is playing tan tan tan tan ” for Piano class  “Flute is playing toot toot toot toot” for Flute class  “Guitar is playing tin tin tin ” for Guitar class  You must not allow the user to declare an object of Instrument class.  Create an array of 10 Instruments.  Assign different type of instrument to Instrument reference.  Check for the polymorphic behavior of play method.  Use the instanceof operator to print that which object stored at which index of instrument array. | Abstract Classes |  |

##### [Packages](http://59.162.104.101:25000/PBLApp/tmodule.action?module=Abstraction%20/Packages%20/%20Exception%20Handling&stream=Java&tm=TM3" \l "T2)

|  |
| --- |
|  |
| **No.** | **Hands-on Assignment** | **Topics Covered** | **Status** |
| 1 | Create a package called test package;  Define a class called foundation inside the test package;  Inside the class, you need to define 4 integer variables;  Var1 as private;  Var2 as default;  Var3 as protected;  Var4 as public;  Import this class and packages in another class.  Try to access all 4 variables of the foundation class and see what variables are accessible and what are not accessible. | Packages Access control Using package |  |
| 2 | Create a class called compartment which represents the ship compartments for watertight subdivision its height, width and breadth.  Take care it should not conflict with the compartment class you have created in Abstract class exercise 2.  To avoid conflict create this class in a new package called com.wipro.automobile.ship | Packages User defined packages |  |
| 3 | Create a package called com.automobile. Define an abstract class called Vehicle.  Vehicle class has the following abstract methods:  public String getModelName()  public String getRegistrationNumber()  public String getOwnerName()  Create TwoWheeler subpackage under Automobile package  Hero class extends Automobile.vehicle class  public int getSpeed()  – Returns the current speed of the vehicle.  public void radio()  – provides facility to control the radio device  Honda class extends com.automobile.vehicle class  public int getSpeed()  – Returns the current speed of the vehicle.  public void cdplayer()  – provides facility to control the cd player device which is available in the car.  Create a test class to test the methods available in all these child class. | Packages User defined packages |  |
| 4 | Add the following ideas to the previous hands on:  Create FourWheeler subpackage under Automobile package  Logan class extends com.automobile.vehicle class  public int speed()  – Returns the current speed of the vehicle.  public int gps()  – provides facility to control the gps device  Ford class extends com.automobile.vehicle class  public int speed()  – Returns the current speed of the vehicle.  public int tempControl()  – provides facility to control the air conditioning device which is available in the car | Packages User defined packages |  |

##### [Interfaces](http://59.162.104.101:25000/PBLApp/tmodule.action?module=Abstraction%20/Packages%20/%20Exception%20Handling&stream=Java&tm=TM3" \l "T4)

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **Hands-on Assignment** | **Topics Covered** | **Status** |
| 1 | A library needs to develop an online application for two types of users/roles, Adults and children. Both of these users should be able to register an account.  Any user who is less than 12 years of age will be registered as a child and they can borrow a “Kids” category book for 10 days, whereas an adult can borrow “Fiction” category books which need to be returned within 7 days.  Note: In future, more users/roles might be added to the library where similar rules will be enforced.  Develop Interfaces and classes for the categories mentioned above.  1. Create an interface LibraryUser with the following methods declared,  Method Name  registerAccount  requestBook  2. Create 2 classes “KidUsers” and “AdultUser” which implements the LibraryUser interface.  3. Both the classes should have two instance variables as specified below.  Instance variables Data type  age int  bookType String  4. The methods in the KidUser class should perform the following logic.  registerAccount function:  if age < 12, a message displaying “You have successfully registered under a Kids Account” should be displayed in the console.  If(age>12), a message displaying, “Sorry, Age must be less than 12 to register as a kid” should be displayed in the console.  requestBook function:  if bookType is “Kids”, a message displaying “Book Issued successfully, please return the book within 10 days” should be displayed in the console.  Else, a message displaying, “Oops, you are allowed to take only kids books” should be displayed in the console.  5. The methods in the AdultUser class should perform the following logic.  registerAccount function:  if age > 12, a message displaying “You have successfully registered under an Adult Account” should be displayed in the console.  If age<12, a message displaying, “Sorry, Age must be greater than 12 to register as an adult” should be displayed in the console.  requestBook function:  if bookType is “Fiction”, a message displaying “Book Issued successfully, please return the book within 7 days” should be displayed in the console.  Else, a message displaying, “Oops, you are allowed to take only adult Fiction books” should be displayed in the console.  6. Create a class “LibraryInterfaceDemo.java” with a main method which performs the below functions,  Test case #1:  Create an instance of KidUser class.  Set the age as specified in the below table and invoke the registerAccount method of the KidUser object  Age  10  18  Set the book Type as specified in the below table and invoke the requestBook method of the KidUser object,  BookType  “Kids”  “Fiction”  Test case #2:  Create an instance of AdultUser class.  Set the age as specified in the below table and invoke the registerAccount method of the AdultUser object  Age  5  23  Set the book Type as specified in the below table and invoke the requestBook method of the AdultUser object  BookType  “Kids”  “Fiction” | Interfaces |  |
| 2 | Write an interface called Playable, with a method  void play();  Let this interface be placed in a package called music.  Write a class called Veena which implements Playable interface. Let this class be placed in a package music.string  Write a class called Saxophone which implements Playable interface. Let this class be placed in a package music.wind  Write another class Test in a package called live. Then,  a. Create an instance of Veena and call play() method  b. Create an instance of Saxophone and call play() method  c. Place the above instances in a variable of type Playable and then call play() | Interfaces |  |

##### [Exception Handling](http://59.162.104.101:25000/PBLApp/tmodule.action?module=Abstraction%20/Packages%20/%20Exception%20Handling&stream=Java&tm=TM3" \l "T5)

|  |
| --- |
|  |
| **No.** | **Hands-on Assignment** | **Topics Covered** | **Status** |
| 1 | Handle exception in number  Problem statement:  Get the input String from user and parse it to integer, if it is not a number it will throw number format exception Catch it and print "Entered input is not a valid format for an integer." or else print the square of that number. (Refer Sample Input and Output).  Sample input and output 1:  Enter an integer: 12  The square value is 144  The work has been done successfully  Sample input and output 2:  Enter an integer: Java  Entered input is not a valid format for an integer. | Exception Handling |  |
| 2 | Write a program that takes as input the size of the array and the elements in the array. The program then asks the user to enter a particular index and prints the element at that index.  This program may generate Array Index Out Of Bounds Exception. Use exception handling mechanisms to handle this exception. In the catch block, print the class name of the exception thrown.  Sample Input and Output 1:  Enter the number of elements in the array  3  Enter the elements in the array  20  90  4  Enter the index of the array element you want to access  2  The array element at index 2 = 4  The array element successfully accessed  Sample Input and Output 2:  Enter the number of elements in the array  3  Enter the elements in the array  20  90  4  Enter the index of the array element you want to access  6  java.lang.ArrayIndexOutOfBoundsException | Exception Handling: Try-catch |  |
| 3 | Write a program that takes as input the size of the array and the elements in the array. The program then asks the user to enter a particular index and prints the element at that index. Index starts from zero.  This program may generate Array Index Out Of Bounds Exception or NumberFormatException . Use exception handling mechanisms to handle this exception.  Sample Input and Output 1:  Enter the number of elements in the array  2  Enter the elements in the array  50  80  Enter the index of the array element you want to access  1  The array element at index 1 = 80  The array element successfully accessed  Sample Input and Output 2:  Enter the number of elements in the array  2  Enter the elements in the array  50  80  Enter the index of the array element you want to access  9  java.lang.ArrayIndexOutOfBoundsException  Sample Input and Output 3:  Enter the number of elements in the array  2  Enter the elements in the array  30  j  java.lang.NumberFormatException | Exception Handling: Try-catch Use multiple catch block |  |
| 4 | Write a class MathOperation which accepts integers from command line. Create an array using these parameters. Loop through the array and obtain the sum and average of all the elements.  Display the result.  Check for various exceptions that may arise like ArithmeticException, NumberFormatException, and so on.  For example: The class would be invoked as follows:  C:>java MathOperation 1900, 4560, 0, 32500 | Exception handling: throws |  |
| 5 | Write a Program with a division method who receives two integer numbers and performs the division operation. The method should declare that it throws ArithmeticException. This exception should be handled in the main method. | throws |  |
| 6 | Write a Program to take care of Number Format Exception if user enters values other than integer for calculating average marks of 2 students. The name of the students and marks in 3 subjects are taken from the user while executing the program.  In the same Program write your own Exception classes to take care of Negative values and values out of range (i.e. other than in the range of 0-100) | Exception Handling: Throw & Used Defined Exception |  |
| 7 | A student portal provides user to register their profile. During registration the system needs to validate the user should be located in India. If not the system should throw an exception.  Step 1: Create a user defined exception class named “InvalidCountryException”.  Step 2: Overload the respective constructors.  Step 3: Create a main class “UserRegistration”, add the following method,  registerUser– The parameter are String username,String userCountry and add the following logic,  • if userCountry is not equal to “India” throw a InvalidCountryException with the message “User Outside India cannot be registered”  • if userCountry is equal to “India”, print the message “User registration done successfully”  Invoke the method registerUser from the main method with the data specified and see how the program behaves,  Name Country Expected Output  Mickey US InvalidCountryException should be thrown.  The message should be “User Outside India cannot be registered”  Mini India The message should be “User registration done successfully”  Sample Input and Output | Exception Handling: User Defined Exception & throw |  |
| 8 | Write a program to accept name and age of a person from the command prompt(passed as arguments when you execute the class) and ensure that the age entered is >=18 and < 60.  Display proper error messages.  The program must exit gracefully after displaying the error message in case the arguments passed are not proper. (Hint : Create a user defined exception class for handling errors.) | Exception handling: User Defined Exception & throw |  |
| 9 | Write a program that accepts 2 integers a and b as input and finds the quotient of a/b.  This program may generate an Arithmetic Exception. Use exception handling mechanisms to handle this exception. In the catch block, print the message as shown in the sample output.  Also illustrate the use of finally block. Print the message “Inside finally block”.  Sample Input and Output 1:  Enter the 2 numbers  5  2  The quotient of 5/2 = 2  Inside finally block  Sample Input and Output 2:  Enter the 2 numbers  5  DivideByZeroException caught  Inside finally block | Exception Handling: Finally block |  |