**JAVA**

**Packages**

**Lab Exercise No:**29

**Exercise Objective(s):***Package*

**Exercise:***Create a package called shapes. Create some classes in the package representing some*

*common geometric shapes like Square, Triangle, Circle and so on. Create a class called*

*TestShapes and create objects for all the shapes and print corresponding messages.*

*Execute the TestShapes class.*

**Solution:**

**package** shapes;

//class Square

**public** **class** Square

{

//displays square details

**void** display()

{

System.***out***.println("Square has four sides");

}

}

**package** shapes;

//class rectangle

**public** **class** Rectangle

{

//displays rectangle details

**void** display()

{

System.***out***.println("Rectangle has four sides");

}

}

**package** shapes;

//class Triangle

**public** **class** Triangle

{

//displays traingle details

**void** display()

{

System.***out***.println("Triangle has three sides");

}

}

**package** shapes;

**public** **class** Test

{

**public** **static** **void** main(String[] args)

{

//square class object

Square s=**new** Square();

//calling display method of square class

s.display();

//Triangle class object

Triangle ts=**new** Triangle();

//calling display method of Triangle class

ts.display();

//Rectangle class object

Rectangle rs=**new** Rectangle();

//calling display method of Rectangle class

rs.display();

}

}

**Lab Exercise No:**30

**Exercise Objective(s):***Jar*

**Exercise:**

1. *Create a new project in which create a package named org.animals. In that create various classes like Lion, Tiger, Deer, Monkey, Elephant and Giraffe. In each class create data members like color, weight,age etc. Create methods like isVegetarian, canClimb, sound etc*
2. *Create another project and in that create a package called zoo and create a class called VandalurZooand create objects for the animals that are existing in zoo and print the characteristic of each animal.*

**Solution:**

**package** org.animal;

//class lion with different characteristics of lion

**public** **class** Lion

{

**int** weight;

String color;

**int** age;

//display details

**public** **void** details()

{

System.***out***.println("----- Lion -----------");

System.***out***.println("Weight ==> 35");

System.***out***.println("Age ==> 9");

System.***out***.println("Color ==> Brown");

System.***out***.println("Is monkey vegeterian ==> "+isVegetarian());

System.***out***.println("Sounds ==> "+sound());

System.***out***.println("Can Climb ==> "+isVegetarian());

}

//retuns whether lion is vegetarian or not

**public** **boolean** isVegetarian()

{

**return** **false**;

}

//retuns the sound that lion makes

**public** String sound()

{

**return** "Growl";

}

//returns whether lion can climb or not

**public** **boolean** canClimb()

{

**return** **true**;

}

}

**package** org.animal;

//class monkey with different characteristics of monkey

**public** **class** Monkey

{

**int** weight;

String color;

**int** age;

//display details

**public** **void** details()

{

System.***out***.println("----- Monkey -----------");

System.***out***.println("Weight ==> 30");

System.***out***.println("Age ==> 12");

System.***out***.println("Color ==> Brown");

System.***out***.println("Is monkey vegeterian ==> "+isVegetarian());

System.***out***.println("Sounds ==> "+sound());

System.***out***.println("Can Climb ==> "+isVegetarian());

}

//retuns whether monkey is vegetarian or not

**public** **boolean** isVegetarian()

{

**return** **true**;

}

//retuns the sound that monkey makes

**public** String sound()

{

**return** "Gibber";

}

//returns whether monkey can climb or not

**public** **boolean** canClimb()

{

**return** **true**;

}

}

**package** org.animal;

//class tiger with different characteristics of tiger

**public** **class** Tiger

{

**int** weight;

String color;

**int** age;

//display details

**public** **void** details()

{

System.***out***.println("---------Tiger -----------");

System.***out***.println("Weight ==> 55");

System.***out***.println("Age ==> 20");

System.***out***.println("Color ==> Yellow");

System.***out***.println("Is monkey vegeterian ==> "+isVegetarian());

System.***out***.println("Sounds ==> "+sound());

System.***out***.println("Can Climb ==> "+isVegetarian());

}

//retuns whether tiger is vegetarian or not

**public** **boolean** isVegetarian()

{

**return** **false**;

}

//retuns the sound that tiger makes

**public** String sound()

{

**return** "Roar";

}

//returns whether tiger can climb or not

**public** **boolean** canClimb()

{

**return** **true**;

}

}

**package** zoo;

**import** org.animals.Lion;

**import** org.animals.Monkey;

**import** org.animals.Tiger;

**public** **class** VandalurZooand

{

**public** **static** **void** main(String[] args)

{

Tiger t=**new** Tiger();

t.details();

Lion l=**new** Lion();

l.details();

Monkey m=**new** Monkey();

m.details();

}

}

**Lab Exercise No:**31

**Exercise Objective(s):***System class*

**Exercise:***Create a class which displays the following about the JVM.*

1. *Version of Java*
2. *Vendor for Java*
3. *Class Path*
4. *Installed home directory*
5. *OS name on which it is installed with version*

**Solution:**

**package** hsbc.training.day4;

**import** java.util.Properties;

//class JavaDetails is used to find details of java Properties using Property class

**public** **class** JavaDetails

{

**public** **static** **void** main(String[] args)

{

System.***out***.println("Java Version ==> "+System.*getProperty*("java.version"));

System.***out***.println("Java Vendor ==> "+System.*getProperty*("java.vendor"));

System.***out***.println("Class Path ==> "+System.*getProperty*("java.class.path"));

System.***out***.println("Installed Home Directory ==> "+System.*getProperty*("java.home"));

System.***out***.println("Installed Home Directory ==> "+System.*getProperty*("os.name"));

}

}

**Lab Exercise No:**32

**Exercise Objective(s):***Scanner class*

**Exercise:***Create a class called Student. Get the details like name, degree, age, total marks and*

*percentage from the user and display the same.*

**Solution:**

**package** hsbc.training.day4;

**import** java.util.Scanner;

**public** **class** Student

{

String name;

**int** rollNo;

**float** cgpa;

**public** **static** **void** main(String[] args)

{

Student s=**new** Student();

s.accept();

s.display();

}

//accepts the student details

**public** **void** accept()

{

Scanner s=**new** Scanner(System.***in***);

System.***out***.println("Enter the name of the student ==>");

name=s.next();

System.***out***.println("Enter the roll no ==> ");

rollNo=s.nextInt();

System.***out***.println("Enter the CGPA scored ==>");

cgpa=s.nextFloat();

}

//display student details

**public** **void** display()

{

System.***out***.println("Student Name ==>"+name);

System.***out***.println("Student Roll No ==>"+rollNo);

System.***out***.println("Student CGPA ==>"+cgpa);

}

}

**Lab Exercise No:**33

**Exercise Objective(s):***Systemclass,usingstaticimport*

**Exercise:***Create a Package called house. Create 2 classes namely Hall and Kitchen.*

1. *In the Hall class print the message “This is the first room while entering the house” without using the class name System explicitly in the println statement.*
2. *In the Kitchen class create an array called appliances and initialize with values and print the same.*
3. *After printing copy that array into a different array.*
4. *Invoke garbage collector explicitly for the Kitchen class.*

**Solution:**

**package** house;

**import** **static** java.lang.System.***out***;

**public** **class** House

{

**public** **static** **void** main(String[] args)

{

Hall h = **new** Hall();

h.hall();

Kitchen k = **new** Kitchen();

k.copy();

}

}

**class** Hall

{

//printing message using System.out as static import

**public** **void** hall()

{

***out***.println("This is the first room while entering the house");

}

}

**class** Kitchen

{

String[] appliances = {"Microwave","Juicer","Induction","Refrigerator","Mixer"};

**public** **void** copy()

{

//copying appliances array

String[] copyAppliances = appliances;

//displayig the copied array

System.***out***.println("Kitchen Appliances ==>");

**for** (**int** i=0;i<copyAppliances.length;i++)

{

System.***out***.println(copyAppliances[i]);

}

//explicity invoking garbage collector;

System.*gc*();

}

}

**JAVA**

**Exception Handling**

**Lab Exercise No:**50

**Exercise Objective(s):***syntax*

**Exercise:***In the Lab Exercise 14, change the code such that the numbers are taken as input from the*

*user. Handle the appropriate exceptions.*

**Solution:**

**package** hsbc.training.day4;

**import** java.util.InputMismatchException;

**import** java.util.Scanner;

//Calculator class

**public** **class** Calculator

{

**public** **static** **void** main(String[] args)

{

//Creating object of Calculator class

Calculator C = **new** Calculator();

Scanner s = **new** Scanner(System.***in***);

**int** num1,num2;

//Accept two numbers

System.***out***.println("Enter two numbers ==>");

num1 = s.nextInt();

num2 = s.nextInt();

//calling addition function

C.add(num1, num2);

//calling difference function

C.diff(num1, num2);

//calling division function

C.div(num1, num2);

//calling multiplication function

C.mul(num1, num2);

}

//Addition function

**public** **void** add(**int** no1,**int** no2)

{

**try**

{

//adding two numbers and storing the result in res

**int** res = no1+no2;

System.***out***.println("Addition of "+no1+" and "+no2+" is "+res);

}

**catch**(InputMismatchException e)

{

System.***out***.println("Input should be a Integer No !");

}

}

//Multiplication function

**public** **void** mul(**int** no1,**int** no2)

{

**try**

{

//multiplying two numbers and storing the result in res

**int** res = no1\*no2;

System.***out***.println("Multiplication of "+no1+" and "+no2+" is "+res);

}

**catch**(InputMismatchException e)

{

System.***out***.println("Input should be a Integer No !");

}

}

//Difference function

**public** **void** diff(**int** no1,**int** no2)

{

**try**

{

//subtracting two numbers and storing the result in res

**int** res = no1-no2;

System.***out***.println("Difference of "+no1+" and "+no2+" is "+res);

}

**catch**(InputMismatchException e)

{

System.***out***.println("Input should be a Integer No !");

}

}

//Division function

**public** **void** div(**int** no1,**int** no2)

{

**try**

{

//dividing two numbers and storing the result in res

**int** res = no1/no2;

System.***out***.println("Division of "+no1+" and "+no2+" is "+res);

}

**catch**(InputMismatchException e)

{

System.***out***.println("Input should be a Integer No !");

}

**catch**(ArithmeticException e)

{

System.***out***.println(e);

}

}

}

**Lab Exercise No:**51

**Exercise Objective(s):***syntax*

**Exercise:***In the Lab Exercise 17, handle the scenarios if the String variable is not initialized.*

**Solution :**

**package** hsbc.training.day4;

**public** **class** StringExceptionHandling

{

**public** **static** **void** main(String args[])

{

**try**

{

String str=**null**;//generating the NullPointerException

//Print the character at the 12th index

System.***out***.println("The character at the 12th index is "+str.charAt(12));

//Checking whether the String contains the word “is”.

System.***out***.println("Whether the String contains the word “is” ?"+str.contains("is"));

//Adding the string “and killed it” to the existing string.

System.***out***.println("Adding the string “and killed it” to the existing string the string is "+str.concat(" and killed it"));

//Checking whether the String ends with the word “dogs”.

System.***out***.println("Whether the String ends with the word “dogs” ? "+str.endsWith("dogs"));

//Checking whether the String is equal to “The quick brown Fox jumps over the lazy Dog”.

System.***out***.println("Whether the given two strings are equal ? "+str.equals("The quick brown Fox jumps over the lazy Dog"));

//Checking whether the String is equal to “THE QUICK BROWN FOX JUMPS OVER THE LAZY DOG”.

System.***out***.println("Whether the given two strings are equal ? "+str.equals("THE QUICK BROWN FOX JUMPS OVER THE LAZY DOG"));

//Finding the index position of the character “a”.

System.***out***.println("Index position of the character “a” is "+str.indexOf('a'));

//Finding the last index position of the character “e”.

System.***out***.println("Last index position of the character “e” is "+str.lastIndexOf('e'));

//Finding the length of the String.

System.***out***.println("The length of the String is "+str.length());

//Checking whether the String matches to “The quick brown Fox jumps over the lazy Dog”.

System.***out***.println("Whether the given two strings matches ? "+str.equalsIgnoreCase("The quick brown Fox jumps over the lazy Dog"));

//Replacing the word “The” with the word “A”.

System.***out***.println("Replacing the word “The” with the word “A”."+str.replace("The", "A"));

//initializing variable with the starting index of animals

**int** startIndexOfFox=str.indexOf("fox");

**int** startIndexOfDog=str.indexOf("dog");

//checking for which animal comes first in the string

**if**(startIndexOfFox<startIndexOfDog)

{

//Spiting the above string into two such that two animal names do not come together.

System.***out***.println("First Part of Split is "+str.substring(0, startIndexOfFox+3));

System.***out***.println("Second Part of Split is "+str.substring(startIndexOfFox+3,str.length()));

}

**else**

{

//Spiting the above string into two such that two animal names do not come together.

System.***out***.println("First Part of Split is "+str.substring(0, startIndexOfDog+3));

System.***out***.println("Second Part of Split is "+str.substring(startIndexOfDog+3,str.length()));

}

//Printing the animal names alone separately from the given string.

System.***out***.println("Names of the animals are "+str.substring(startIndexOfFox, startIndexOfFox+3)+" , "+str.substring(startIndexOfDog, startIndexOfDog+3));

//Print the above string in completely lower case.

System.***out***.println("String in completely lower case : " +str.toLowerCase());

//Print the above string in completely upper case.

System.***out***.println("String to in completely upper case : " +str.toUpperCase());

}

**catch**(NullPointerException e) //handling the NullPointerException

{

System.***out***.println(e);

}

}

}

**Lab Exercise No:**52

**Exercise Objective(s):***syntax*

**Exercise:***Using Lab Exercise 17, catch and demonstratethe required exceptions.*

**Solution:**

**package** hsbc.training.day4;

**public** **class** StringExceptionHandling

{

**public** **static** **void** main(String args[])

{

**try**

{

String str="";//generating the StringIndexOutOfBoundsException

//Print the character at the 12th index

System.***out***.println("The character at the 12th index is "+str.charAt(12));

//Checking whether the String contains the word “is”.

System.***out***.println("Whether the String contains the word “is” ?"+str.contains("is"));

//Adding the string “and killed it” to the existing string.

System.***out***.println("Adding the string “and killed it” to the existing string the string is "+str.concat(" and killed it"));

//Checking whether the String ends with the word “dogs”.

System.***out***.println("Whether the String ends with the word “dogs” ? "+str.endsWith("dogs"));

//Checking whether the String is equal to “The quick brown Fox jumps over the lazy Dog”.

System.***out***.println("Whether the given two strings are equal ? "+str.equals("The quick brown Fox jumps over the lazy Dog"));

//Checking whether the String is equal to “THE QUICK BROWN FOX JUMPS OVER THE LAZY DOG”.

System.***out***.println("Whether the given two strings are equal ? "+str.equals("THE QUICK BROWN FOX JUMPS OVER THE LAZY DOG"));

//Finding the index position of the character “a”.

System.***out***.println("Index position of the character “a” is "+str.indexOf('a'));

//Finding the last index position of the character “e”.

System.***out***.println("Last index position of the character “e” is "+str.lastIndexOf('e'));

//Finding the length of the String.

System.***out***.println("The length of the String is "+str.length());

//Checking whether the String matches to “The quick brown Fox jumps over the lazy Dog”.

System.***out***.println("Whether the given two strings matches ? "+str.equalsIgnoreCase("The quick brown Fox jumps over the lazy Dog"));

//Replacing the word “The” with the word “A”.

System.***out***.println("Replacing the word “The” with the word “A”."+str.replace("The", "A"));

//initializing variable with the starting index of animals

**int** startIndexOfFox=str.indexOf("fox");

**int** startIndexOfDog=str.indexOf("dog");

//checking for which animal comes first in the string

**if**(startIndexOfFox<startIndexOfDog)

{

//Spiting the above string into two such that two animal names do not come together.

System.***out***.println("First Part of Split is "+str.substring(0, startIndexOfFox+3));

System.***out***.println("Second Part of Split is "+str.substring(startIndexOfFox+3,str.length()));

}

**else**

{

//Spiting the above string into two such that two animal names do not come together.

System.***out***.println("First Part of Split is "+str.substring(0, startIndexOfDog+3));

System.***out***.println("Second Part of Split is "+str.substring(startIndexOfDog+3,str.length()));

}

//Printing the animal names alone separately from the given string.

System.***out***.println("Names of the animals are "+str.substring(startIndexOfFox, startIndexOfFox+3)+" , "+str.substring(startIndexOfDog, startIndexOfDog+3));

//Print the above string in completely lower case.

System.***out***.println("String in completely lower case : " +str.toLowerCase());

//Print the above string in completely upper case.

System.***out***.println("String to in completely upper case : " +str.toUpperCase());

}

**catch**(StringIndexOutOfBoundsException e) //handling StringIndexOutOfBoundsException

{

System.***out***.println(e);

}

}

}

**Lab Exercise No:**53

**Exercise Objective(s):***syntax*

**Exercise:***Using Lab Exercise 22, catch and demonstrate the required exceptions.*

**Solution:**

**package** hsbc.training.day4;

**public** **class** ArrayExceptionHandling

{

**public** **static** **void** main(String args[])

{

**int** arr[][] = **new** **int**[][] { {2,3,8},

{1,-2,3},

{7,6,0}};

//displaying the 2D array

System.***out***.println(" 2D Array ==>");

*display2dArray*(arr);

//calling the arrSquare method

System.***out***.println("Square of 2D array ==> ");

*arrSquare*(arr);

}

//creating display2dArray method to display the 2D array

**public** **static** **void** display2dArray(**int** a[][])

{

**try**

{

**for**(**int** i=0;i<=a.length;i++) //generating the ArrayIndexOutOfBoundsException

{

**for**(**int** j=0;j<a[i].length;j++)

{

System.***out***.print(a[i][j] + " ");

}

System.***out***.println();

}

System.***out***.println();

}

**catch**(ArrayIndexOutOfBoundsException e) //handling the ArrayIndexOutOfBoundsException

{

System.***out***.println(e);

}

}

//squaring the 2D array

**public** **static** **void** arrSquare(**int** a[][])

{

**int** squareArr[][] = **new** **int**[a.length][a.length];

**for**(**int** i=0;i<a.length;i++)

{

**for**(**int** j=0;j<a.length;j++)

{

squareArr[i][j]=(**int**) Math.*pow*(a[i][j], 2);

}

}

*display2dArray*(squareArr);

}

}

**Lab Exercise No:**54

**Exercise Objective(s):***Exception class methods*

**Exercise:***By using multiple catch blocks, write a class to demonstrate the order of the execution of the*

*catch blocks usingNegativeArraySizeException,ArrayIndexOutOfBoundsException,*

*StringIndexOutOfBoundsException, IndexOutOfBoundsException, NullPointerException,*

*ArithmeticException and print the stack trace for each exception.*

**Solution:**

**package** hsbc.training.day4;

**import** java.util.Scanner;

**public** **class** ExceptionHandling

{

**public** **static** **void** main(String[] args)

{

Scanner sc=**new** Scanner(System.***in***);

**try**

{

**int** arr[]=**new** **int**[-7]; //generating NegativeArraySizeException

**int** a[]=**new** **int**[] {11,22,33};

**for**(**int** i=0;i<=a.length;i++) //generating ArrayIndexOutOfBoundsException

{

System.***out***.println(a[i]);

}

String str="Computer";

//generating StringIndexOutOfBoundsException

System.***out***.println("Display charachter at 9th position ==>"+str.charAt(9));

String str1=**null**;

//generating NullPointerException

System.***out***.println(str1.toLowerCase());

**int** div=4/0; //generating ArithmeticException

}

**catch** (NegativeArraySizeException e)

{

System.***out***.println(e);

e.printStackTrace();

}

**catch**(ArrayIndexOutOfBoundsException e)

{

System.***out***.println (e);

e.printStackTrace();

}

**catch**(StringIndexOutOfBoundsException e)

{

System.***out***.println(e);

e.printStackTrace();

}

**catch**(NullPointerException e)

{

System.***out***.println(e);

e.printStackTrace();

}

**catch**(ArithmeticException e)

{

System.***out***.println (e);

e.printStackTrace();

}

}

}

**Lab Exercise No:**55

**Exercise Objective(s):***User-defined exceptions*

**Exercise:***In the Lab Exercise 46, handle the expected exceptions by writing custom defined exceptions.*

**Recommended duration:** *20 Mins*

**Solution Guidance (if applicable):** *NA*

**Lab Exercise No:**56

**Exercise Objective(s):***finally keyword*

**Exercise:***Create a class such that it resets the value of the objects it used to null after its usage in all*

*cases.*

**Recommended duration:** *20 Mins*

**Solution Guidance (if applicable):** *Use finally block*

**Lab Exercise No:**57

**Exercise Objective(s):***finally keyword*

**Exercise:***Create a class such that a method uses the try catch block with the return type of String.*

**Recommended duration:** *20 Mins*

**Solution Guidance (if applicable):** *Use finally block*

**Lab Exercise No:**58

**Exercise Objective(s):***User-defined exceptions*

**Exercise:***Createa class called Employee which asks the user to input the name and the age of a*

*employee. Raise a custom defined exception when the user enters an employee name*

*that has been already entered and raise another exception if the age is negative or less*

*than 18 or greater than 60.*

**Recommended duration:** *20 Mins*

**Solution Guidance (if applicable):***NA*