**ASSIGNMENT 7**

**SET A**

**Q.1 Write a java program to accept a number from the user, if number is zero then throw user defined exception ― Number is 0, otherwise check whether no is prime or not.**

import java.io.\*;

class NumberZeroException extends Exception

{

public String toString()

{

return("Number is 0");

}

}

class PrimeNumber

{

int a;

BufferedReader br=new BufferedReader(new InputStreamReader(System.in));

PrimeNumber()

{

try

{

System.out.println("Enter any integer to check prime ");

a=Integer.parseInt(br.readLine());

if(a==0)

throw new NumberZeroException();

}

catch(NumberZeroException ex)

{

System.out.println(ex);

}

catch(IOException ex1)

{

System.out.println("Enter proper number");

}

}

public void prime()

{

int cnt=0;

for(int i=2;i<=a/2;i++)

if(a%i==0)

{

cnt++;

break;

}

if(cnt==0)

System.out.println(a+" Number is prime");

else

System.out.println(a+" Number is not prime");

}

public static void main(String args[])

{

PrimeNumber pn=new PrimeNumber();

pn.prime();

}

}

OUTPUT

Enter any integer to check prime

7

7 Number is prime

**Q.2 Write a Java program that reads a list of numbers from a file and throws an exception if any of the numbers are positive.**

import java.io.File;

import java.io.FileNotFoundException;

import java.util.Scanner;

public class Positive\_Number\_Checkk {

public static void main(String[] args) {

try {

checkNumbersFromFile("test.txt");

System.out.println("All numbers are non-positive.");

} catch (FileNotFoundException e) {

System.out.println("Error: " + e.getMessage());

} catch (PositiveNumberException e) {

System.out.println("Error: " + e.getMessage());

}

}

public static void checkNumbersFromFile(String fileName) throws FileNotFoundException, PositiveNumberException {

File file = new File(fileName);

Scanner scanner = new Scanner(file);

while (scanner.hasNextLine()) {

int number = Integer.parseInt(scanner.nextLine());

if (number > 0) {

throw new PositiveNumberException("Positive number found: " + number);

}

}

scanner.close();

}

}

class PositiveNumberException extends Exception {

public PositiveNumberException(String message) {

super(message);

}

}

**Q.3 Define a class MyDate (day, month, year) with methods to accept and display a MyDate object. Accept date as dd, mm, yyyy. Throw user defined exception “InvalidDateException” if the date is invalid. Examples of invalid dates : 12 15 2015, 31 6 1990, 29 2 2001**

import java .io.\*;

class InvalidDateException extends Exception

{

}

class MyDate

{

int day,mon,yr;

void accept(int d,int m,int y)

{

day=d;

mon=m;

yr=y;

}

void display()

{

System.out.println("Date is valid : "+day+"/"+mon+"/"+yr);

}

}

class DateMain

{

public static void main(String arg[]) throws Exception

{

System.out.println("Enter Date : dd mm yyyy ");

BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

int day=Integer.parseInt(br.readLine());

int mon=Integer.parseInt(br.readLine());

int yr=Integer.parseInt(br.readLine());

int flag=0;

try

{

if(mon<=0 || mon>12)

throw new InvalidDateException();

else

{

if(mon==1 || mon==3 || mon==5 || mon==7 || mon==8 || mon==10 || mon==12)

{

if(day>=1 && day <=31)

flag=1;

else

throw new InvalidDateException();

}

else if (mon==2)

{

if(yr%4==0)

{

if(day>=1 && day<=29)

flag=1;

else throw new InvalidDateException();

}

else

{

if(day>=1 && day<=28)

flag=1;

else throw new InvalidDateException();

}

}

else

{

if(mon==4 || mon == 6 || mon== 9 || mon==11)

{

if(day>=1 && day <=30)

flag=1;

else throw new InvalidDateException();

}

}

}

if(flag== 1)

{

MyDate dt = new MyDate();

dt.accept(day,mon,yr);

dt.display();

}

}

catch (InvalidDateException mm)

{

System.out.println("Invalid Date");

}

}

}

Output:

Enter Date : dd mm yyyy

31

12

2012

Date is valid : 31/12/2012

=== Code Execution Successful ===

**Q.4 Write a Java program to create a method that takes an integer as a parameter and throws an exception if the number is odd.**

public class Exception\_OddNumber {

public static void main(String[] args) {

int n = 18;

trynumber(n);

n = 7;

trynumber(n);

}

public static void trynumber(int n) {

try {

checkEvenNumber(n);

System.out.println(n + " is even.");

} catch (IllegalArgumentException e) {

System.out.println("Error: " + e.getMessage());

}

}

public static void checkEvenNumber(int number) {

if (number % 2 != 0) {

throw new IllegalArgumentException(number + " is odd.");

}

}

}

Output:

18 is even.

ERROR!

Error: 7 is odd.

=== Code Execution Successful ===

**SET C**

**Q.1 Write a java program to validate PAN number and Mobile Number. If it is invalid then throw a user defined Exception “Invalid Data”, otherwise display it.**

import java.io.\*;

// Custom exception class

class InvalidDetails extends Exception {

public InvalidDetails(String message) {

super(message);

}

}

class Main {

static int n;

public static void main(String args[]) {

DataInputStream dr = new DataInputStream(System.in);

try {

System.out.print("\*\*\*\*\*\*\*\*\* Do you want to validate? \*\*\*\*\*\*\*\*\*\n"

+ "1. Mobile Number Press : 1\n"

+ "2. PAN Card Press : 2\n"

+ "Enter choice: ");

n = Integer.parseInt(dr.readLine()); // Read choice input

switch (n) {

case 1:

System.out.print("Enter Mobile Number: ");

String num = dr.readLine();

// Validate mobile number (starts with 7, 8, or 9 and has 10 digits)

if (num.matches("(0/91)?[7-9][0-9]{9}")) {

System.out.println("Valid Mobile Number!");

} else {

throw new InvalidDetails("Invalid Mobile Number!");

}

break;

case 2:

System.out.print("Enter PAN Number: ");

String str = dr.readLine();

// Validate PAN number (Format: 5 letters, 4 digits, 1 letter)

if (str.matches("[A-Z]{5}[0-9]{4}[A-Z]{1}")) {

System.out.println("Valid PAN Card Number!");

} else {

throw new InvalidDetails("Invalid PAN Card Number!");

}

break;

default:

throw new InvalidDetails("Invalid Option selected!");

}

} catch (InvalidDetails nz) {

System.out.println(nz.getMessage()); // Print the custom error message

} catch (NumberFormatException e) {

System.out.println("Invalid input. Please enter a valid number.");

} catch (IOException e) {

System.out.println("Input error: " + e.getMessage());

}

}

}

Output:

Note: /tmp/us4g5kAyPL/Main.java uses or overrides a deprecated API.

Note: Recompile with -Xlint:deprecation for details.

\*\*\*\*\*\*\*\*\* Do you want to validate? \*\*\*\*\*\*\*\*\*

1. Mobile Number Press : 1

2. PAN Card Press : 2

Enter choice: 1

Enter Mobile Number: 9674531238

Valid Mobile Number!

=== Code Execution Successful ===

**Q.2 Write a java program to check whether a given candidate is eligible for voting or not. Handle user defined as well as system defined Exception.**

import java.util.Scanner;

class InvalidAgeException extends Exception {

public InvalidAgeException(String message) {

super(message);

}

}

public class VotingEligibility

{

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

try {

System.out.print("Enter your age: ");

int age = scanner.nextInt();

if (age < 18) {

throw new InvalidAgeException("You are not eligible to vote. Age must be 18 or above.");

} else {

System.out.println("You are eligible to vote.");

}

} catch (InvalidAgeException e) {

System.out.println("User-defined exception: " + e.getMessage());

} catch (Exception e) {

System.out.println("System-defined exception: " + e.getMessage());

} finally {

scanner.close();

}

}

}

Output:

Enter your age: 20

You are eligible to vote.

=== Code Execution Successful ===

**Q.3 Write a Java program that reads a list of integers from the user and throws an exception if any numbers are even.**

import java.util.ArrayList;

import java.util.List;

import java.util.Scanner;

public class EvenNumberException extends Exception {

public EvenNumberException(int number) {

super("Number " + number + " is even.");

}

}

public class Main {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

List<Integer> numbers = new ArrayList<>();

System.out.println("Enter a list of integers (enter -1 to stop):");

while (true) {

int number = scanner.nextInt();

if (number == -1) {

break;

}

numbers.add(number);

}

try {

for (int number : numbers) {

if (number % 2 == 0) {

throw new EvenNumberException(number);

}

}

System.out.println("All numbers are odd.");

} catch (EvenNumberException e) {

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

}

}

}

Output:

Enter a list of integers (enter -1 to stop):

1

3

4

5

-1

Number 4 is even.

**Q.4 Write a java program to accept a number from a user, If it is greater than 200 then throw a user defined exception “Number is out of Range” otherwise do the addition of digits of that number. (Use static keyword)**

import java.util.Scanner;

public class NumberRangeException {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("Enter a number: ");

int num = scanner.nextInt();

try {

checkRange(num);

int sum = addDigits(num);

System.out.println("Sum of digits: " + sum);

} catch (OutOfRangeException e) {

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

}

}

public static void checkRange(int num) throws OutOfRangeException {

if (num > 200) {

throw new OutOfRangeException("Number is out of range");

}

}

public static int addDigits(int num) {

int sum = 0;

while (num > 0) {

sum += num % 10;

num /= 10;

}

return sum;

}

}

class OutOfRangeException extends Exception {

public OutOfRangeException(String message) {

super(message);

}

}

Output:

Enter a number: 89

Sum of digits: 17

=== Code Execution Successful ===