Implement the following questions to understand Exception handling properly:

**Unchecked Exceptions (Runtime Exceptions)**

Unchecked exceptions extend RuntimeException and do not require explicit handling with throws or try-catch.

**1. Implement NullPointerException**

Write a Java program where you initialize a String as null and try to call the .length() method on it. Handle the exception using a try-catch block.

**Code:-**

class NullExc{

public static void main(String[] args){

String s=null;

int x=s.length();

System.out.println(x);

}}

**Handle the exception using a try-catch block.**

import java.lang.\*;

class NullExc{

public static void main(String[] args){

try{

String s=null;

int x=s.length();

System.out.println(x);}

catch(NullPointerException e){

System.out.println("NullPointerException Caught");

}

}}

**2. Implement ArithmeticException**

Write a Java program that performs **division by zero** and catches the ArithmeticException.

**Code :-**

class ExceptionDemo1{

public static void main(String[] args) {

int i=10;

int result = i/0;

System.out.println(result);

}

}

**Handle the exception using a try-catch block.**

import java.lang.\*;

class ExceptionDemo1{

public static void main(String[] args) {

int i=10;

try{

int result = i/0;

System.out.println(result);

}

catch(ArithmeticException e){

System.out.println("no cannot devide by 0");

}

}

}

**3. Implement ArrayIndexOutOfBoundsException**

Create an array of 5 elements and try to access an index that does not exist (e.g., index 10). Handle the exception properly.

**Code:-**

class ArrayException{

public static void main(String[] args) {

int arr[]= {1,3,4,6,7};

for(int i=0;i<=5;i++){

System.out.println(arr[8]);}

}}

Error:- Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException: Index 8 out of bounds for length 5

at ArrayException.main(ArrayException.java:7)

**Handle the exception using a try-catch block.**

class ArrayException{

public static void main(String[] args) {

try{

int arr[]= {1,3,4,6,7};

for(int i=0;i<=5;i++){

System.out.println(arr[8]);}}

catch(ArrayIndexOutOfBoundsException e){

System.out.println("ArrayIndexOutOfBoundsException cought");

}

}}

**4. Implement NumberFormatException**

Write a Java program that tries to convert a **non-numeric string** to an integer using Integer.parseInt() and catches the NumberFormatException.

**Code:-**

class NumberExc{

public static void main(String[] args){

String s="67a535";

int a= Integer.parseInt(s);

System.out.println(a);}}

**Handle the exception using a try-catch block.**

class NumberExc{

public static void main(String[] args){

String s="67a535";

try{

int a= Integer.parseInt(s);

System.out.println(a);}

catch (NumberFormatException e) {

System.out.println("Error: Invalid number format. " );

}}}

**5. Implement IllegalArgumentException**

Write a Java method setAge(int age) that throws an IllegalArgumentException if the age is negative or greater than 150.

**Handle the exception using a try-catch block.**

**Checked Exceptions**

Checked exceptions extend Exception and must be either **handled using try-catch** or **declared with throws**.

**6. Implement IOException**

Write a Java program that attempts to read from a file that does not exist and catches IOException.

**7. Implement FileNotFoundException**

Write a Java program that tries to open a file that does not exist using FileReader, and handle the FileNotFoundException.