JAVA EXCEPTION HANDLING

Exception

- ArithmeticException occurs

If we divide any number by zero, there occurs an ArithmeticException.

- **int** a=50/0;//ArithmeticException

- NullPointerException occurs

If we have a null value in any variable, performing any operation on the variable throws a NullPointerException.

- String s=**null**;
- System.out.println(s.length());//NullPointerException

NumberFormatException occurs

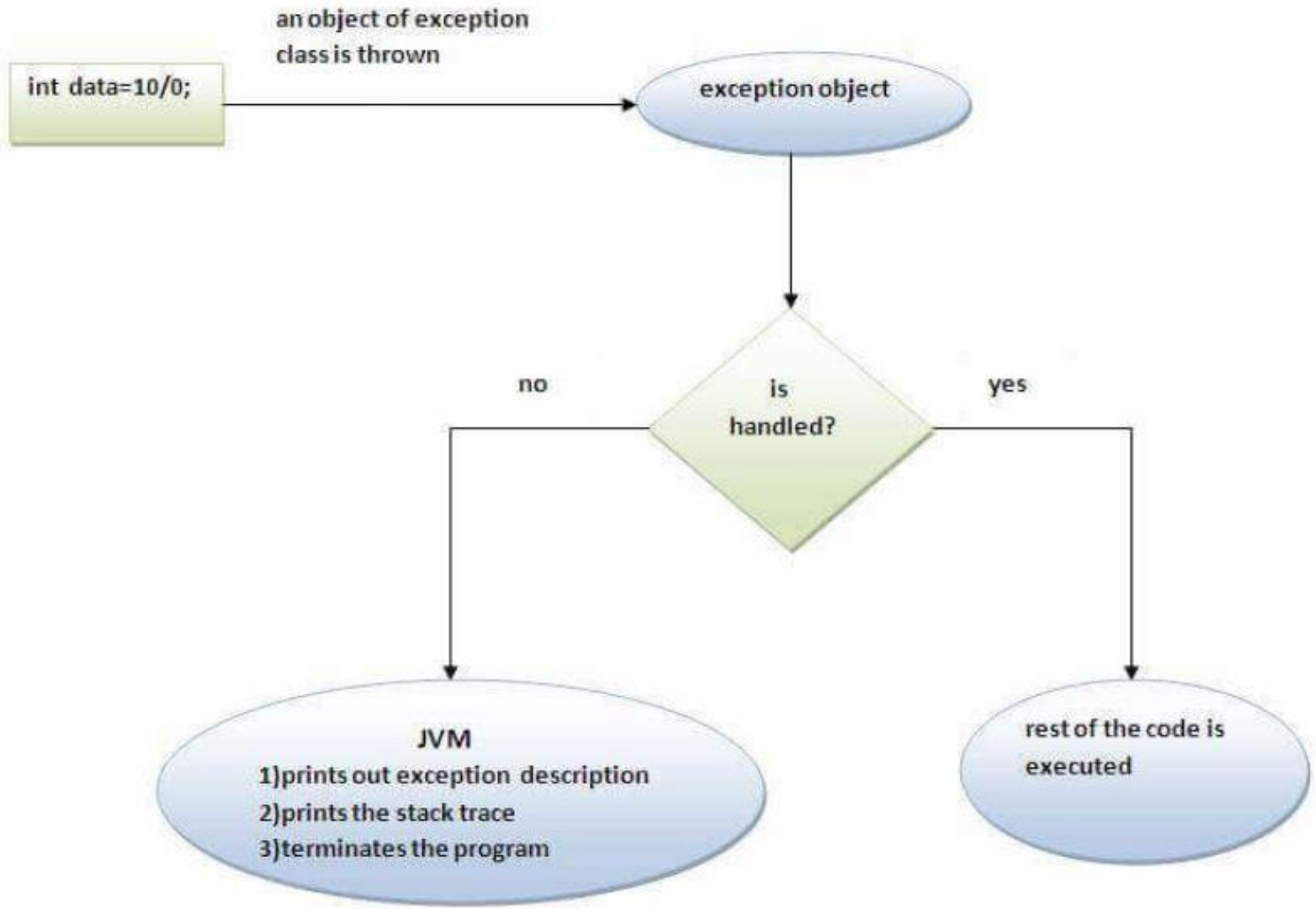
- The wrong formatting of any value may occur NumberFormatException.
- Suppose I have a string variable that has characters, converting this variable into digit will occur NumberFormatException.
- String s="abc";
- **int** i=Integer.parseInt(s);//NumberFormatException

- ArrayIndexOutOfBoundsException occurs

If you are inserting any value in the wrong index, it would result in ArrayIndexOutOfBoundsException as shown below:

- **int** a[]=**new int**[5];
- a[10]=50; //ArrayIndexOutOfBoundsException

JAVA EXCEPTION HANDLING

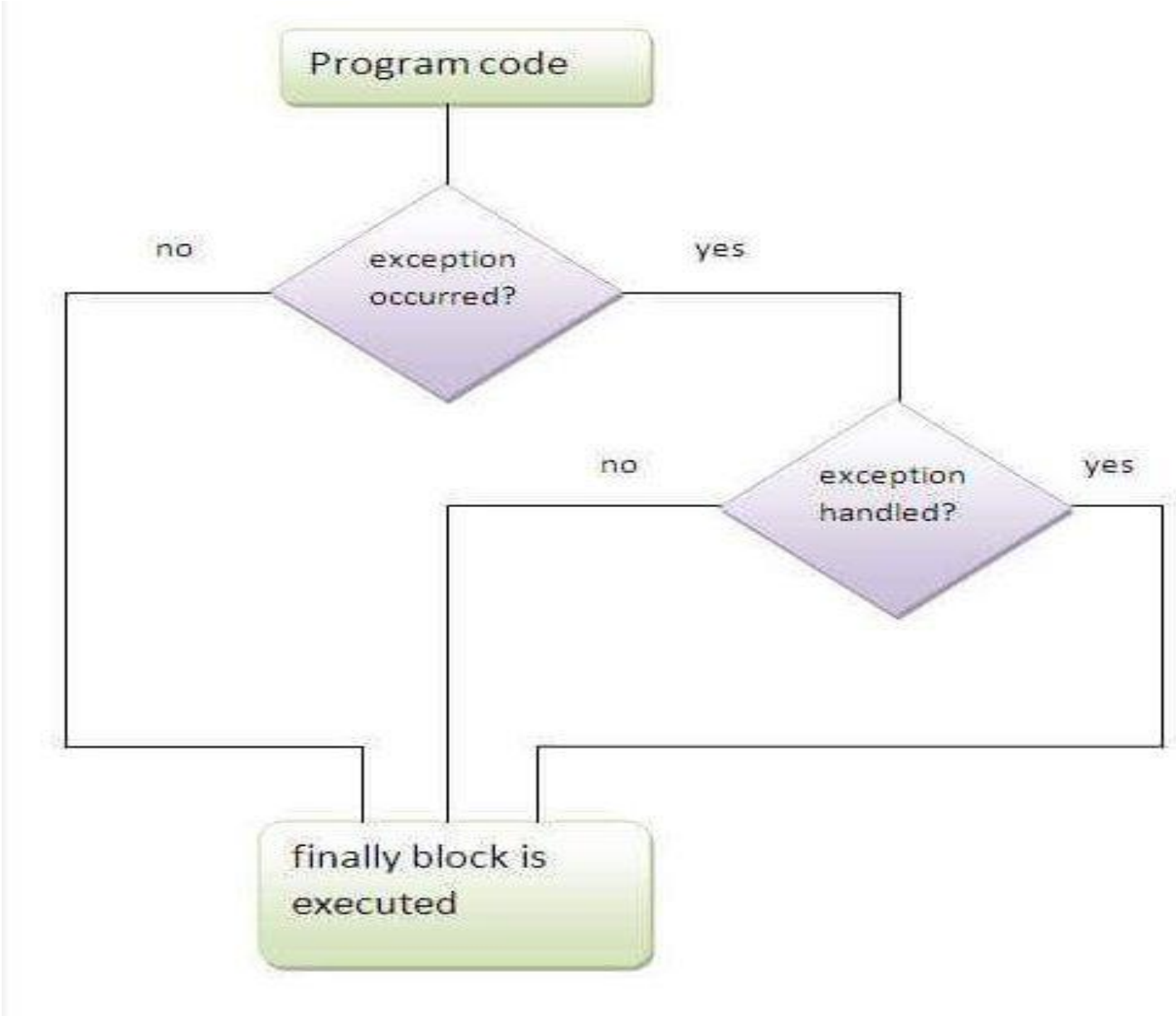


JAVA FINALLY BLOCK

- Java finally block is a block that is used to execute important code such as closing connection, stream etc.

- Java finally block is always executed whether exception is handled or not.

- Java finally block follows try or catch block


Program code
exception occurred?
no
yes
exception handled?
no
yes
finally block is executed

FINALLY BOCK

```
class TestFinallyBlock{
  public static void main(String args[]){
  try{
   int data=25/5;
   System.out.println(data);
  }
  catch(NullPointerException e){System.out.println(e);}
  finally{System.out.println("finally block is always executed
");}
  System.out.println("rest of the code...");
  }
}
```

- Case1:**exception doesn't occur**.
- Case2: **exception occurs and not handled**
- **Case3: exception occurs and handled**.

output

- Case1: 5

finally block is always executed

rest of the code...

Case2: finally block is always executed

Exception in thread main
java.lang.ArithmeticException:/ by zero

Case3: Exception in thread main
java.lang.ArithmeticException:/ by zero finally block is always executed rest of the code...

Java throw keyword

- The Java throw keyword is used to explicitly throw an exception.

- Syntax

throw exception;

If the age is less than 18, we are throwing ArithmeticException otherwise print a message welcome to vote.

```
public class TestThrow1{
   static void validate(int age){
     if(age<18)
      throw new ArithmeticException("not valid");
     else
      System.out.println("welcome to vote");
   }
   public static void main(String args[]){
     validate(13);
     System.out.println("rest of the code...");
  }
}
```

Java Exception Keywords

- Try

The "try" keyword is used to specify a block where we should place exception code. The try block must be followed by either catch or finally. It means, we can't use try block alone.

- Catch

The "catch" block is used to handle the exception. It must be preceded by try block which means we can't use catch block alone. It can be followed by finally block later.

- Finally

The "finally" block is used to execute the important code of the program. It is executed whether an exception is handled or not.

- Throw

The "throw" keyword is used to throw an exception.

- Throws

The "throws" keyword is used to declare exceptions. It doesn't throw an exception. It specifies that there may occur an exception in the method. It is always used with method signature.