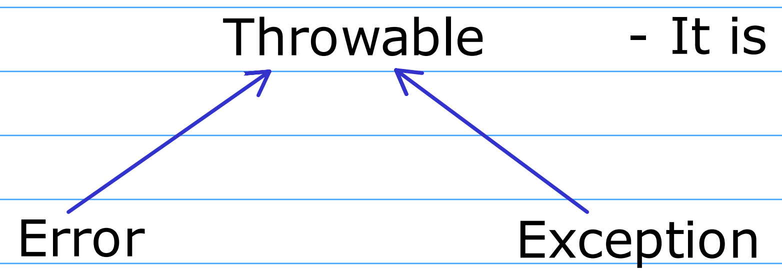
Exception Handling

- Problems that occur at run time,
- These problems needs to be handled to avoid the program crash

- Problems

1. Runtime environment (Errors)
2. Logical Problems or due to the wrong user input (Exceptions)



- It is the super class of all the errors and excetions in java

Error class represents all the problems that occur dude to runtime environment

Exception class represents all the problems that occur within your program due to logical errors or wrong input

JRE

JVM

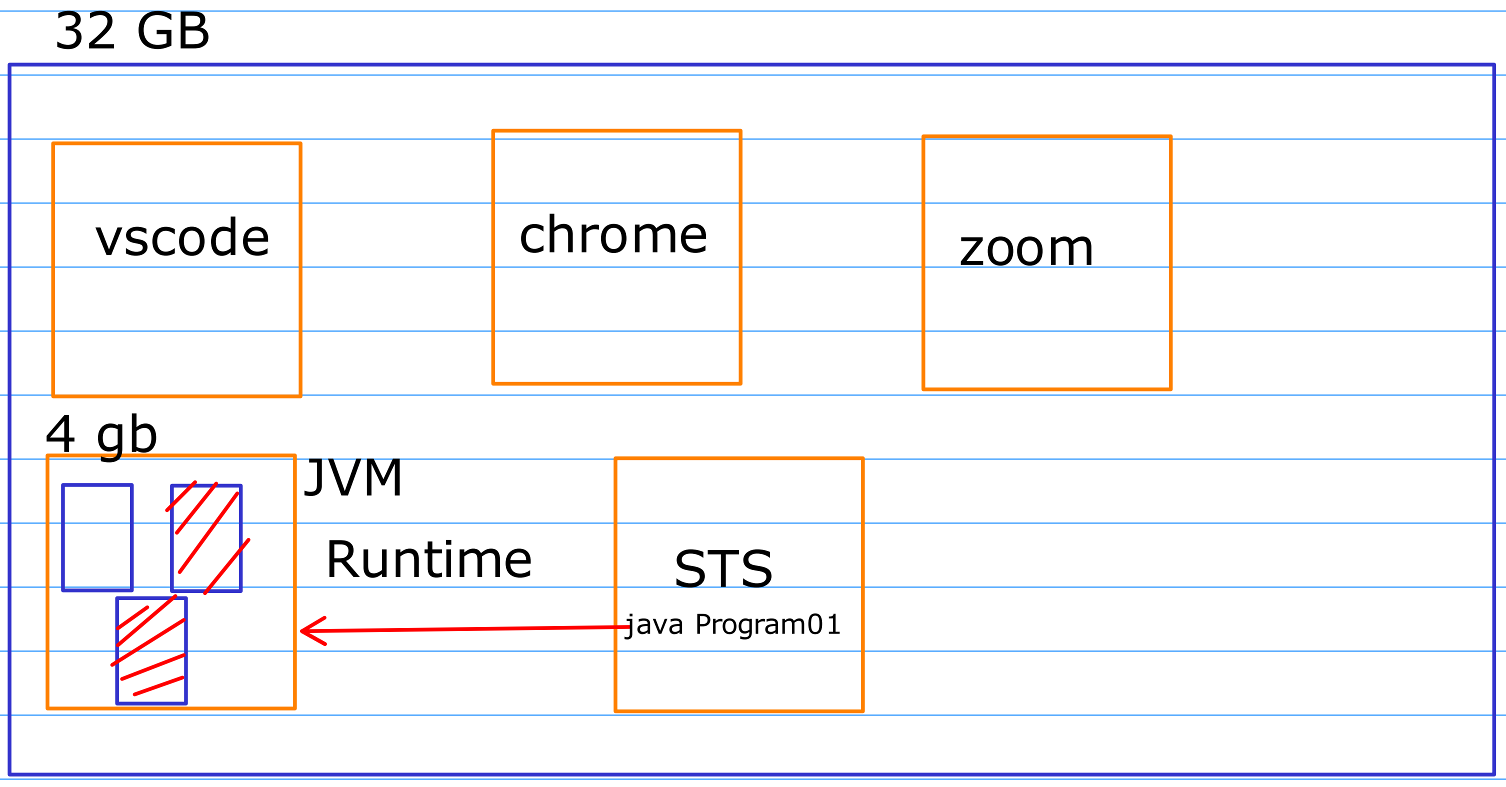
Java recommends NOT TO handle the errors Let the Program Crash

Java Recomends TO Handle the exceptions

JVM

OS (Text, Data, RoData,Stack,Heap

RunTime Environment



1. try
2. catch
3. throw
4. throws
5. finally

1

```
try{

}catch(){

}
```

2

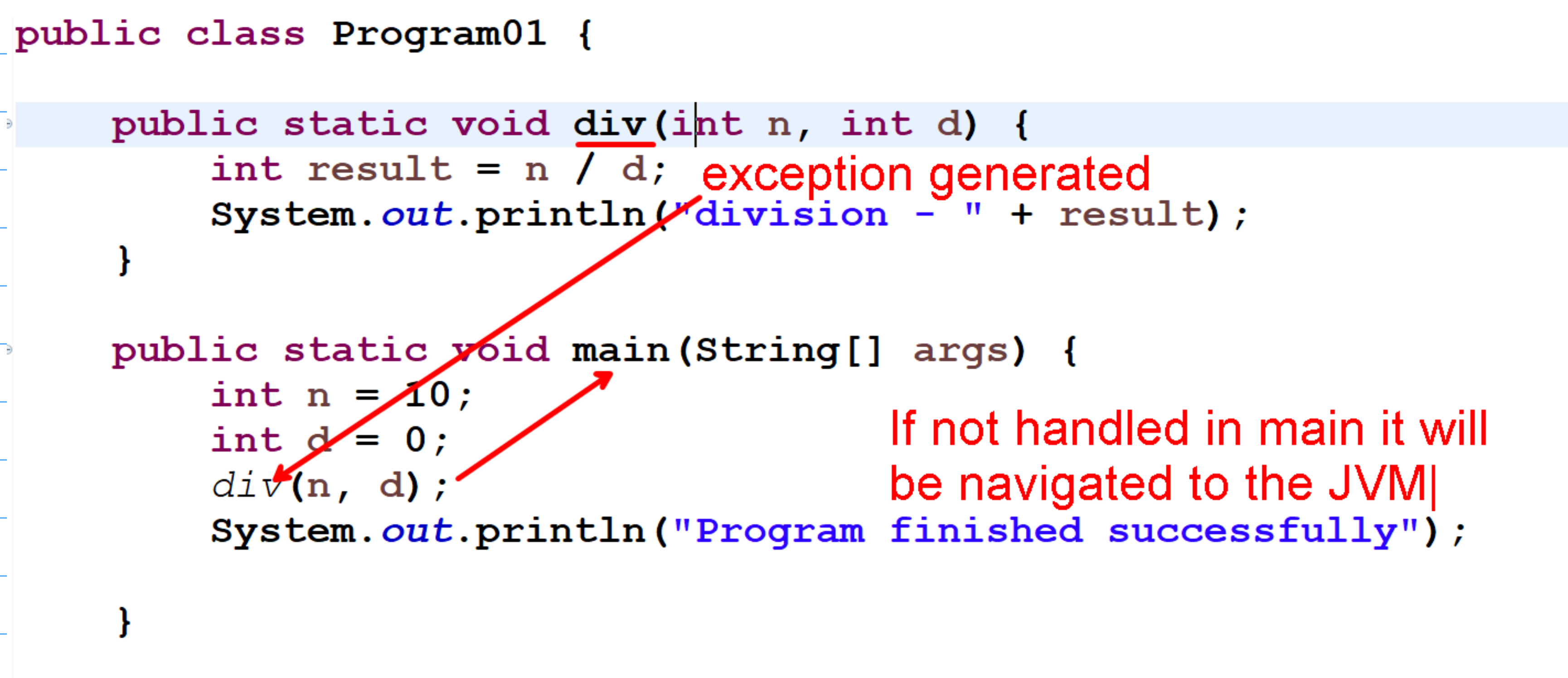
```
try{

}
finally{
}
```

3

```
try(){

}
```

syntax of try

exception Handling mechanism
generating an exception



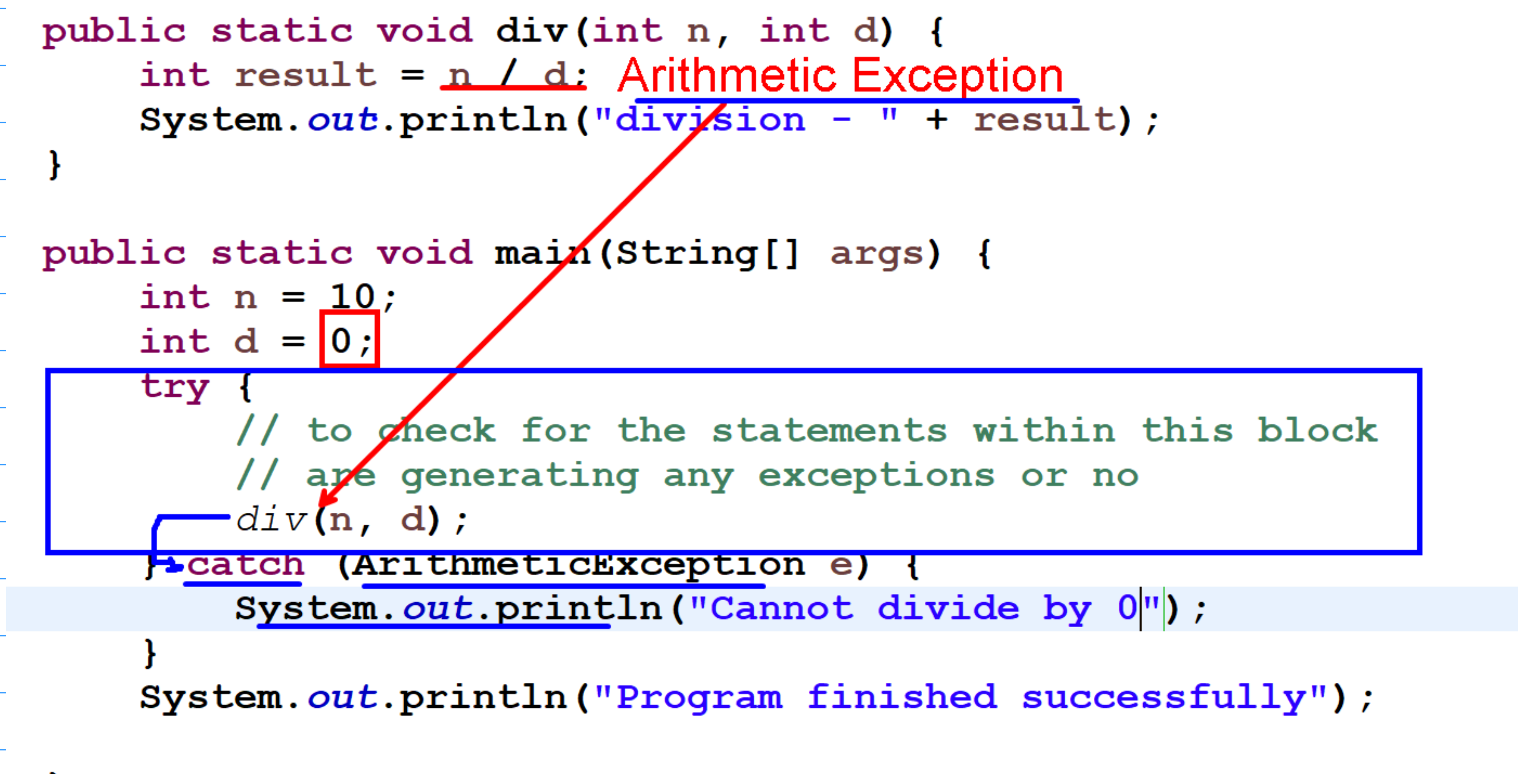
```
public class Program01 {

    public static void div(int n, int d) {
        int result = n / d;
        System.out.println("division - " + result);
    }

    public static void main(String[] args) {
        int n = 10;
        int d = 0;
        div(n, d);
        System.out.println("Program finished successfully");

    }
```

exception generated

If not handled in main it will be navigated to the JVM

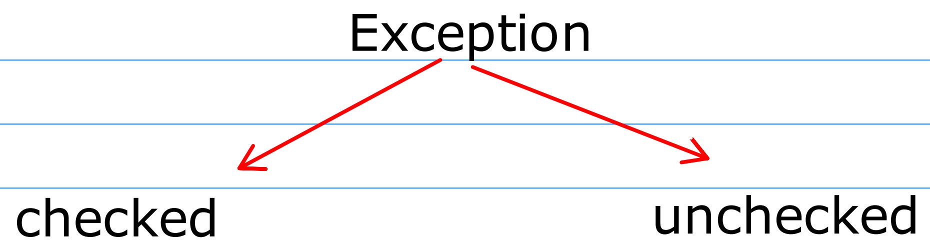
Navigating the exections towards the JVM is considered as bad programming practice

All the exceptions that are generated should be handled atleast in the main



```
public static void div(int n, int d) {
    int result = n / d;
    System.out.println("division - " + result);
}

public static void main(String[] args) {
    int n = 10;
    int d = 0;
    try {
        // to check for the statements within this block
        // are generating any exceptions or no
        div(n, d);
    } catch (ArithmeticException e) {
        System.out.println("Cannot divide by 0");
    }
    System.out.println("Program finished successfully");
```

Arithmetic Exception



checked	unchecked
The exceptions that are mandatory to be handled. else the code will not be compiled	The exceptions that are optional to be handled
Exception class itself and all its subclass except RuntimeException class are all checked exception	Runtime exception class itself and all its subclass are called as unchecked exception

1. try -> chcek for exceptions if any generated from the statements within it
2. catch -> Handle the exections if any generated from the try block
3. throw -> Is to generate new exceptions
4. throws -> navigate the checked exception from the current method to its caller method
5. finally -> used to close the resources

DMC

DAC CDAC System

DITISS - Skills (OOP,JAVA,Python,..)

Full Stack python Mobile