ABHISHEK DHAR
23201
E-10

ASSIGNMENT - 06

EXCEPTION HANDLING

Title → Exception Handling

Aim → Implement a program to handle arithmetic exception, array index out of bounds. User enters array of elements. Average of elements is displayed. It numbers were not integers, program would throw a no. format exception. If there are zero elements, program would throw Arithmetic exception and display it.

Objectives → To understand Exception Handling.

Theory →

- What is an exception?

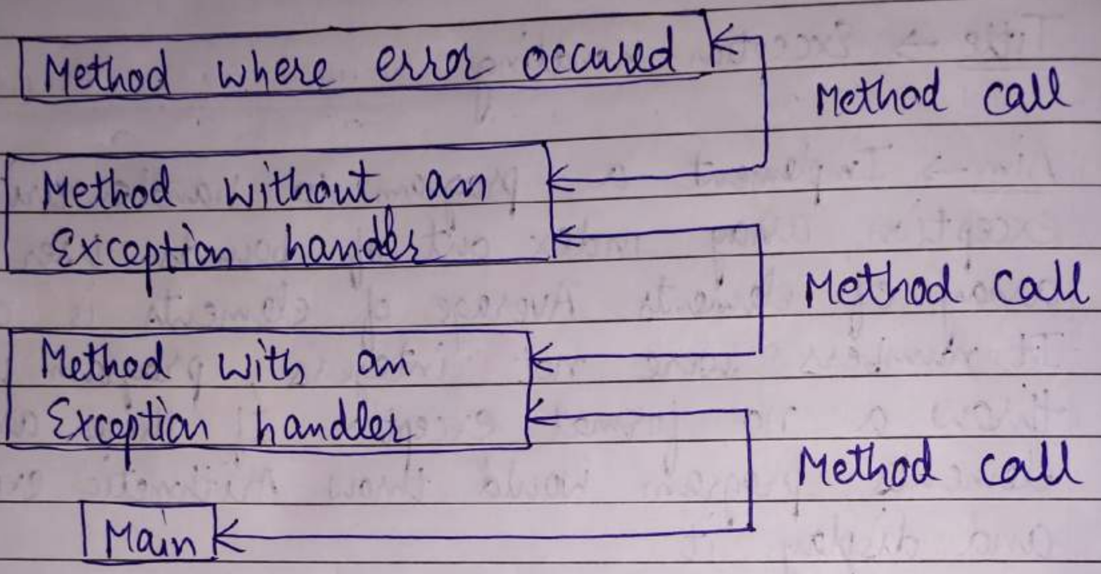
- This term is a shortand for "exceptional event"
- It occurs during execution of a program, that disrupts natural flow of program.

- Exception Handling

- When an error occurs within a method, method creates an object and hands it off to runtime system.

- The object called an 'exceptional object' contains information about error, including its type and

State of program where error occured
- This list is K/a call stack as shown


Method where error occured
Method call
Method without an Exception handler
Method call
Method with an Exception handler
Method call
Main

- The runtime system searches call stack for a method that contains a block of code that can handle exception.
This block is K/a exception handler

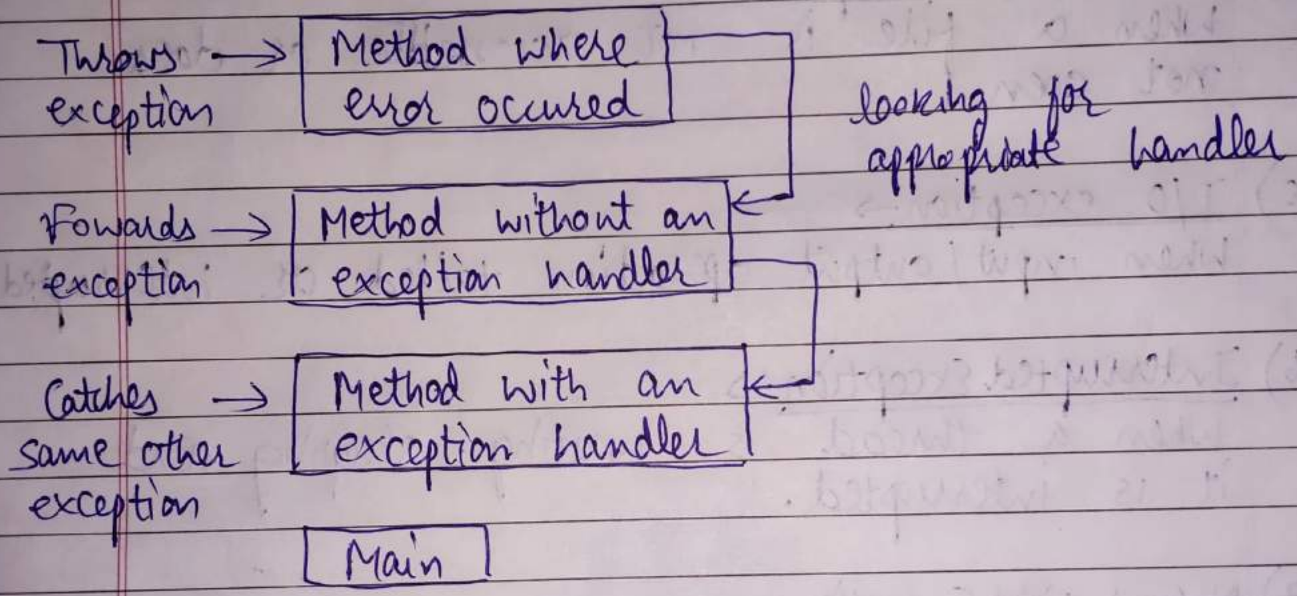
- The search begins with method in which error occured and proceeds through call stack in reverse order in which methods where called.

- When appropriate hander is found runtime system passes exception to the handler.

- An exception handler is considered appropriate if type of exception object thrown matches type that can be handled by handler.

- Exception handler choosen is said to catch the exception.

- If runtime system exhaustively searches all methods on call stack without finding an appropriate handler, the program terminates. As shown below.


Throws exception
Method where error occured
looking for appropriate handler
Fowards exception
Method without an exception handler
Catches same other exception
Method with an exception handler
Main

* Types of Exception →

Built in and User defined.

→ Built in exceptions →

1) Arithmetic →
When exceptional condition has occured in arithmetic operation. Eg : / by 0

2) ArrayIndexOutofBoundException →
When array accessed with illegal index.

3) ClassNotFoundException →
When we try to access a class where defination is not found.

4) FileNotFoundException →
When a file is not accessable or does not open.

5) I/O exception →
When input/output operation failed or interrupted.

6) InterruptedException →
When a thread is waiting. sleeping and it is interrupted.

7) NoSuchfieldException →
When class does not contain field specified.

8) NoSuchMethodException →
Accessing a method which is not found.

9) NullPointerException →
When refering to members of a null object.

10) NumberFormatException →
When a method cannot convert string into numeric

11) Run Time Exception →
Any Exception that occurs during runtime.

12) String Index Out of Bound Exception →
Thrown by string class to indicate illegal accessing of string index.

Example →

```
public class rough {
 try { public static void main (String [] args) {
  int [] arr = new int [ ] {1, 2, 3, 4, 5};
   System.out.println (arr[9])
  }
 }
 catch (Exception e) {
  System.out.println (e);
  }
}
```

Output →

```
java.lang.ArrayIndexOutofBoundsException Index
9 out of bounds for length 5.
```

* Working →

- Try block encounters an exception when index 9 of array is trying to get accessed.
- Catch block appoints an exception handler 'e' to the situation and that 'e' is displayed on the screen.

* Advantages →

1) Separating error handling code from regular code →
- Block of code that might throw exception like division of integers can be separated and handled without affecting other part of code.

2) Propagating errors up the call stack →
- A ~~[illegible]~~ method can duck any exceptions thrown within it, thereby allowing a method far up the stack to catch it.

3) Grouping and differentiating error types →
- We can handle specific errors differentialy.
Eg — FileNotFound and I/O exceptions can be handled by using two different catch statements.

* Disadvantages →

1) Exception can trap only Runtime error.
∴ PQ/SQL program can trap & recover from compile-time errors.

2) Exceptions can mask statement that caused error!

* Validations →

Case 1 →

Enter value greater than or equal to size of array, then ArrayIndexOutofBoundsException is thrown.

Case 2 →
If num1, num2 were not Integers, program would throw a no. format exception.

Case 3 →
If num2 were zero, program would throw an arithmetic exception and display it.

Case 4 →
User enters two no's num1, num2.
Division at num1 and num2 is displayed.

* Class diagram →

Class division
num1, num2 : int
read_n_display(): void divide ( ): float exception ( ): void

Input →
Integers for dividend & divisors are given as input
Dividend = 4   Divisor = 2

Output →
2.0