TryOut: try - catch block

```
class DemoException
{
public static void main(String[] args)
        {
        try {
                int quotient = 5/0;
                System.out.println(“The ans is : ”+quotient);
            }
        catch(NullPointerException exception) {
                System.out.println("Inside NullPointerException");
            }
        catch(ArithmeticException exception){
                System.out.println("Inside ArithmeticException");
                }
        catch(Exception exception){
                System.out.println("There is exception in code");
                }

        System.out.println("Method execution ends");

        }
}
```

TryOut: finally keyword

```
class DemoException
{
public static void main(String[] args)
        {
        try {
                int quotient = 5/0;
                System.out.println(“The ans is : ”+quotient);
            }
        catch(NullPointerException exception) {
```

```
            System.out.println("Inside NullPointerException");
        }
    catch(ArithmeticException exception){
            System.out.println("Inside ArithmeticException");
            }
    catch(Exception exception){
            System.out.println("There is exception in code");
            }
    finally{
            System.out.println(“inside finally block”);
            }
    System.out.println("Method execution ends");

    }
}
```

Tryout: throw keyword

```
class DemoException
{
public static void main(String[] args)
    {
    try {
        if(10 < 20)
            throw new Exception("This is Custom Exception.");
        System.out.println("Please proceed to the check-out");
       }
     catch(Exception e) {
        System.out.println(e.getMessage());
       }

    }
}
```

Tryout: throws

The below code will have compilation errors in checkStock() method due to the exception not being handled.

```
class MobileShopee{
    static int stockAvailable = 400;
    public static void checkStock(int quantityRequired) {
        if(stockAvailable < quantityRequired)
            throw new Exception("There is not enough stock
available.");
        System.out.println("Please proceed to the check-out");
    }
    public static void buyMobiles(int quantityRequired) {
        checkStock(550);
        System.out.println("Please pay for the items in your cart.");
    }
    public static void main(String[] args) {
        buyMobiles(550);
    }
}
```

First Scenario:

```
class MobileShopee {
    static int stockAvailable = 400;
    public static void checkStock(int quantityRequired) throws
Exception{
        if(stockAvailable < quantityRequired)
            throw new Exception("There is not enough stock
available.");
        System.out.println("Please proceed to the check-out");
    }
    public static void buyMobiles(int quantityRequired) {
        try{
            checkStock(550);
            System.out.println("Thank you for shopping at
MobileShopee");
        } catch(Exception exception) {
```

```
                System.out.println(exception.getMessage());
            }
        }
        public static void main(String[] args) {
            buyMobiles(550);
        }
    }
```

Second Scenario:

```
class MobileShopee{
    static int stockAvailable = 400;
    public static void checkStock(int q       uantityRequired) throws
Exception{
        if(stockAvailable < quantityRequired)
            throw new Exception("There is not enough stock available.");
        System.out.println("Please proceed to the check-out");
    }
    public static void buyMobiles(int quantityRequired) throws Exception{
        checkStock(550);
        System.out.println("Please pay for the items in your cart.");
    }
    public static void main(String[] args) {
        try{
            buyMobiles(550);
        } catch (Exception exception) {
            System.out.println(exception.getMessage());
        }
    }
}
```

Third Scenario:

```
class MobileShopee{
   static int stockAvailable = 400;
   public static void checkStock(int quantityRequired) throws Exception{
      if(stockAvailable < quantityRequired)
         throw new Exception("There is not enough stock available.");
      System.out.println("Please proceed to the check-out");
   }
   public static void buyMobiles(int quantityRequired) throws Exception{
      checkStock(550);
      System.out.println("Please pay for the items in your cart.");
   }
   public static void main(String[] args) throws Exception{
      buyMobiles(550);
   }
}
```

the exception will get propagated to the Runtime exception. The Runtime exception will then print the exception stack trace in the output window.

TryOut : unchecked exception

```
class MarksCalculator {

        public static void calculateAverage(int... marks) {

                if (marks.length != 0) {
                        int sum = 0;
                        for (int i = 0; i < marks.length; i++) {
                                sum += marks[i];
                        }
                        System.out.println("Average marks: " + sum /
marks.length);
                } else {
                        throw new ArithmeticException("The marks list is not
updated");
```

```
		}

	}

	public static void main(String[] args) {
		try {
			calculateAverage();
		} catch (ArithmeticException arithmeticException) {
			System.out.println(arithmeticException.getMessage());
		} catch (Exception exception) {
			System.out.println("Some error occurred");
		}
	}
}
```

Tryout : User - Defined Exceptions

```
//Userdefined Exception - ValidationException created
class UserdefinedException extends Exception{
  public UserdefinedException (String message){
    super(message);
  }
}

class Demo1{

	// throws keyword indicates that this method might throw an
exception
	public void checkAge(int age) throws UserdefinedException {
		if(age>19) {
			throw new UserdefinedException ("Not eligible to be
selected");
```

```
                //throw keyword explicitly throw an exception
            }
            else {
                System.out.println("Eligible to to be selected");

            }
        }
}


class Tester {

    public static void main(String[] args) {
        Demo1 d= new Demo1();
        int[] ageList = { 15, 16, 18, 17, 19, 20, 14,15 };
        for (int index : ageList) {
            try {
                d.checkAge(index);
            } catch (UserdefinedException  e) {
                // Uncomment below line to understand the flow
of the exception
                // e.printStackTrace();
                System.out.println("Error: "+e.getMessage());
            }
        }

}
```