** What is the deifference between **throw and throws :**
----
throw : [To throw the exception object explicitly]
----

* In case of predefined exception, try block is reposonsible to create the **exception object** with the help of JVM and throw that exception object to the nearest catch block.

* If we want to throw an exception object explicitly then we should use **throw keyword.**

* If we want to use throw keyword with any class object (throw new Foo()) then that class must be a **Throwable class that means It must in the Exception Hierarchy**

Example :

```
class Foo
{
    public Foo()
    {

    }
}
public class ExceptionDemo
{
    public static void main(String[] args)
    {
      try
      {
            throw new Foo();
      }
      catch (Exception e)
      {
            System.out.println(e);
      }
    }
}
```

Note : We will get a compilation error because Foo is not a Throwable type

* throw is used to throw the expection object explicitly so after throw keyword statement we can't write another statement otherwise It will become un-reachable.

throws :
----

* We should use throws keyword at **method declaration** to describe that, this particular method may OR may not generate a prticular type of exception at runtime.

* Declaring a method as throws indicates that the current method is not interested to handle the exception and try to throw the exception object to the caller method OR JVM for handling purpose.

* Generally, We declare throws keyword at method declaration with checked exception but we can also use with unchecked exception that describes, If any exception encounter at runtime then exception will be propagated to the caller method, but with unchecked exception it is optional.

Types of Exception :
----

* Exception are of two types :

1) Predefined OR Built-In Exception
2) Userdefined OR Custom Exception

Predefined Exception :-

The Exceptions which are already defined by Java software people for some specific purposes are called predefined Exception or Built-in exception.
Ex :

IOException, ArithmeticException and so on

Userdefined Exception :-

The exceptions which are defined by user according to their own use and requirement are called User-defined Exception.

Ex:-

InvalidAgeException, GreaterMarksException

**** How to develop our own custom exception :
----

* In order to develop our own custom exception we should follow the the following steps :

1) We can develop both checked and un-checked type of custom exception.
   a) In case of checked exception our userdefined class must extends from java.lang.Exception
   b) In case of unchecked exception our userdefined class must extends from java.lang.RuntimeException

Example :

```
public class InvalidAgeException extends Exception  //Checked [Handling is compulsory]
{
}

public class InvalidAgeException extends RuntimeException  //UnChecked [Handling not reqd]
{
}
```

* We should declare a no argument constructor (If we don't want to pass the error message) and parameterized constructor with String as a parameter (If we want to pass the error message) in our user-defined class.

* In order to pass the error message in the super class we should use **super()**

* We should throw the exception object by using throw keyword.

//Program on Checked Exception :

```
package com.ravi.exception;

import java.util.Scanner;

class InvalidAgeException extends Exception
{
   //Version Compatibility
   private static final long serialVersionUID = 1L;

   public InvalidAgeException()
   {

   }

   public InvalidAgeException(String errorMessage)
   {
      super(errorMessage);
   }
}

public class CustomCheckedException {

   public static void main(String[] args)
   {
      Scanner sc = new Scanner(System.in);
      try(sc)
      {
         System.out.print("Enter your Age :");
         int age = sc.nextInt();
         validateAge(age);
      }
      catch(InvalidAgeException e)
      {
         System.out.println(e.toString());
         System.out.println("..................");
         System.out.println(e.getMessage());
         System.out.println("..................");
         e.printStackTrace();
      }

   }

   public static void validateAge(int age) throws InvalidAgeException
   {
      if(age<18)
      {
         throw new InvalidAgeException("Age is Invalid");
      }
      else
      {
         System.out.println("You can go for a Movie");
      }
   }
}
```

//Program on unchecked Exception :

```
package com.ravi.exception;

import java.util.Scanner;

class GreaterMarksException extends RuntimeException
{
   private static final long serialVersionUID = 1L;

   public GreaterMarksException()
   {
   }

   public GreaterMarksException(String errorMessage)
   {
     super(errorMessage);
   }
}


public class CustomUncheckedException {

   public static void main(String[] args)
   {
      Scanner sc = new Scanner(System.in);
      try(sc)
      {
         System.out.print("Enter your Marks :");
         int marks = sc.nextInt();
         validateMarks(marks);
      }
      catch(GreaterMarksException e)
      {
         System.out.println(e.toString());
         System.out.println("..................");
         System.out.println(e.getMessage());
         System.out.println("..................");
         e.printStackTrace();
      }
      System.out.println("Thank you !!");
   }

   public static void validateMarks(int marks)
   {
      if(marks > 100)
      {
         throw new GreaterMarksException("Invalid Marks");
      }
      else
      {
         System.out.println("Your Marks is :"+marks);
      }


   }

}
```

Array in java :

What is a variable in java ?
----

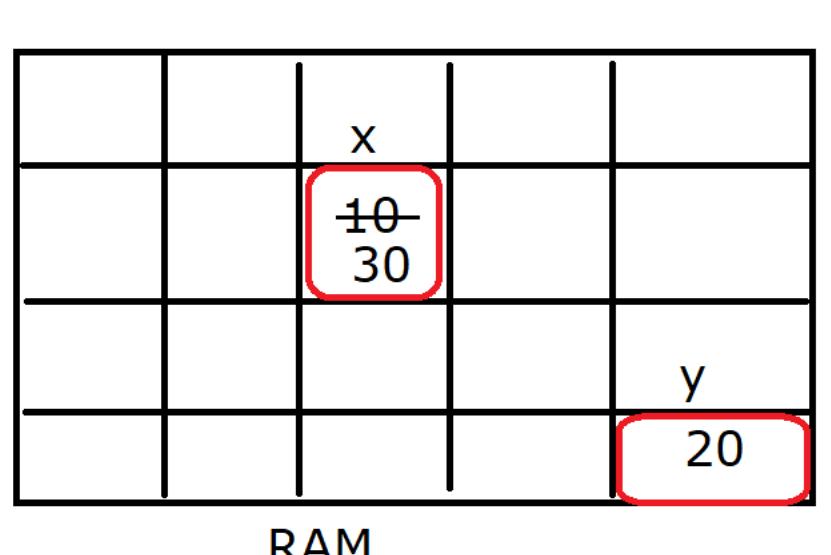
* A variable is a **name** given for the memory location. It is used to store the data in a non contigous memory location.

int x = 10;

int y = 20;

x = x + y;

Variable
Vary + able



RAM

Drawback of an Ordinary Variable :
----

1) It can hold only one value at a time.
2) It is available in RAM so it is temporarly memory.

* In order to hold multiple values in a single variable we should use next level of concept i.e array

* In order to store the data permanently we should use file concept.

What is an Array in java ?
