UNIT V

MULTITHREADING IN JAVA

Exception handling in try catch finally in Java

1. **Control flow in try-catch clause OR try-catch-finally clause**
   - **Case 1:** Exception occurs in try block and handled in catch block
   - **Case 2:** Exception occurs in try-block is not handled in catch block
   - **Case 3:** Exception doesn't occur in try-block
2. **try-finally clause**
   - **Case 1:** Exception occurs in try block
   - **Case 2:** Exception doesn't occur in try-block
   -

CONTROL FLOW IN TRY-CATCH OR TRY-CATCH-FINALLY

1. Exception occurs in try block and handled in catch block: If a statement in try block raised an exception, then the rest of the try block doesn't execute and control passes to the **corresponding** catch block. After executing the catch block, the control will be transferred to finally block(if present) and then the rest program will be executed.

- **Control flow in try-catch:**

```
// Java program to demonstrate control flow of try-catch clausewhen exception occur in try block
// and handled in catch block
class GFG
{
   public static void main (String[] args)
   {
      // array of size 4.
      int[] arr = new int[4];
      try
      {
         inti = arr[4];

         // this statement will never execute
         // as exception is raised by above statement
         System.out.println("Inside try block");
```

```
        }
        catch(ArrayIndexOutOfBoundsException ex)
        {
            System.out.println("Exception caught in Catch block");
        }

        // rest program will be executed
        System.out.println("Outside try-catch clause");
    }
}
```

Output

```
Exception caught in Catch block
Outside try-catch clause
```

- **Control flow in try-catch-finally clause :**

```
// Java program to demonstrate control flow of try-catch-finally clausewhen exception occur in try
//blockand handled in catch block
class GFG
{
    public static void main (String[] args)
    {
        // array of size 4.
        int[] arr = new int[4];
        try
        {
            inti = arr[4];
            // this statement will never executeas exception is raised by above statement
            System.out.println("Inside try block");
        }
                catch(ArrayIndexOutOfBoundsException ex)
        {
            System.out.println("Exception caught in catch block");
        }
        finally
        {
            System.out.println("finally block executed");
        }
        // rest program will be executed
        System.out.println("Outside try-catch-finally clause");
    }
}
```

Output

```
Exception caught in catch block
finally block executed
Outside try-catch-finally clause
```

2. Exception occurred in try-block is not handled in catch block: In this case, the default handling mechanism is followed. If finally block is present, it will be executed followed by the default handling mechanism.

- **try-catch clause :**

```
// Java program to demonstrate
// control flow of try-catch clause
// when exception occurs in try block
// but not handled in catch block
class GFG
{
   public static void main (String[] args)
   {
      // array of size 4.
      int[] arr = new int[4];
      try
      {
         inti = arr[4];

         // this statement will never execute
         // as exception is raised by above statement
         System.out.println("Inside try block");
      }
      // not a appropriate handler
      catch(NullPointerException ex)
      {
         System.out.println("Exception has been caught");
      }
      // rest program will not execute
      System.out.println("Outside try-catch clause");
   }
}
```

Run Time Error:

```
Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException: 4
atGFG.main(GFG.java:12)
```

- **try-catch-finally clause :**

```
// Java program to demonstrate control flow of try-catch-finally clausewhen exception occur in try //
// blockbut not handled in catch block
```

```
class GFG
{
   public static void main (String[] args)
   {

      // array of size 4.
      int[] arr = new int[4];

      try
      {
         inti = arr[4];

         // this statement will never execute
         // as exception is raised by above statement
         System.out.println("Inside try block");
      }
      // not a appropriate handler
      catch(NullPointerException ex)
      {
         System.out.println("Exception has been caught");
      }
      finally
      {
         System.out.println("finally block executed");
      }
      // rest program will not execute
      System.out.println("Outside try-catch-finally clause");
   }
}
```

Output :

```
finally block executed
```

Run Time error:

```
Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException: 4
atGFG.main(GFG.java:12)
```

3. Exception doesn't occur in try-block: In this case catch block never runs as they are only meant to be run when an exception occurs. finally block(if present) will be executed followed by rest of the program.

- **try-catch clause :**

```
// Java program to demonstrate try-catch
```

```
// when an exception doesn't occurred in try block
class GFG
{
   public static void main (String[] args)
   {
      try
      {
         String str = "123";
         intnum = Integer.parseInt(str);
         // this statement will execute
         // as no any exception is raised by above statement
         System.out.println("Inside try block");
      }
      catch(NumberFormatException ex)
      {
         System.out.println("catch block executed...");
      }
      System.out.println("Outside try-catch clause");
   }
}
```

Output

```
Inside try block
Outside try-catch clause
```

- **try-catch-finally clause**

```
// Java program to demonstrate try-catch-finally
// when exception doesn't occurred in try block
class GFG
{
   public static void main (String[] args)
   {
   try
   {
      String str = "123";
      intnum = Integer.parseInt(str);
      // this statement will execute
      // as no any exception is raised by above statement
      System.out.println("try block fully executed");
   }
   catch(NumberFormatException ex)
   {
      System.out.println("catch block executed...");
```

```
    }
    finally
    {
        System.out.println("finally block executed");
    }
    System.out.println("Outside try-catch-finally clause");
    }
}
```

Output

```
try block fully executed
finally block executed
Outside try-catch-finally clause
```

Control flow in try-finally

In this case, no matter whether an exception occurs in try-block or not **finally will always be executed.** But control flow will depend on whether an exception has occurred in the try block or not.

1. **Exception raised:** If an exception has occurred in the try block then the control flow will be finally block followed by the default exception handling mechanism.

```
// Java program to demonstrate control flow of try-finally clausewhen exception occur in try block
class GFG
{
   public static void main (String[] args)
   {

      // array of size 4.
      int[] arr = new int[4];
      try
      {
         inti = arr[4];
         // this statement will never execute
         // as exception is raised by above statement
         System.out.println("Inside try block");
      }
      finally
      {
         System.out.println("finally block executed");
      }
      // rest program will not execute
```

```
        System.out.println("Outside try-finally clause");
    }
}
```

Output :

```
finally block executed
Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException: 4
atGFG.main(GFG.java:11)
```

2. Exception not raised: If an exception does not occur in the try block then the control flow will be finally block followed by the rest of the program

```
// Java program to demonstrate control flow of try-finally  when exception doesn't occur in try block
class GFG
{
   public static void main (String[] args)
   {
      try
      {
         String str = "123";
         intnum = Integer.parseInt(str);
         // this statement will execute
         // as no any exception is raised by above statement
         System.out.println("Inside try block");
      }
      finally
      {
         System.out.println("finally block executed");
      }
      // rest program will be executed
      System.out.println("Outside try-finally clause");
   }
}
```

Output

```
Inside try block
finally block executed
Outside try-finally clause
```