Dt : 15/9/2023

**imp*

Summary of Exception handling process:

1.Exception definition

2.Exception Vs Error

3.Exception Handling process

(i)try

(ii)catch

(iii)finally

4.Hierarchy of "Throwable"

5.Types of Exceptions

(i)Checked Exceptions

(ii)UnChecked Exceptions

6.Exception re-throwing process

7.Exception Propagation

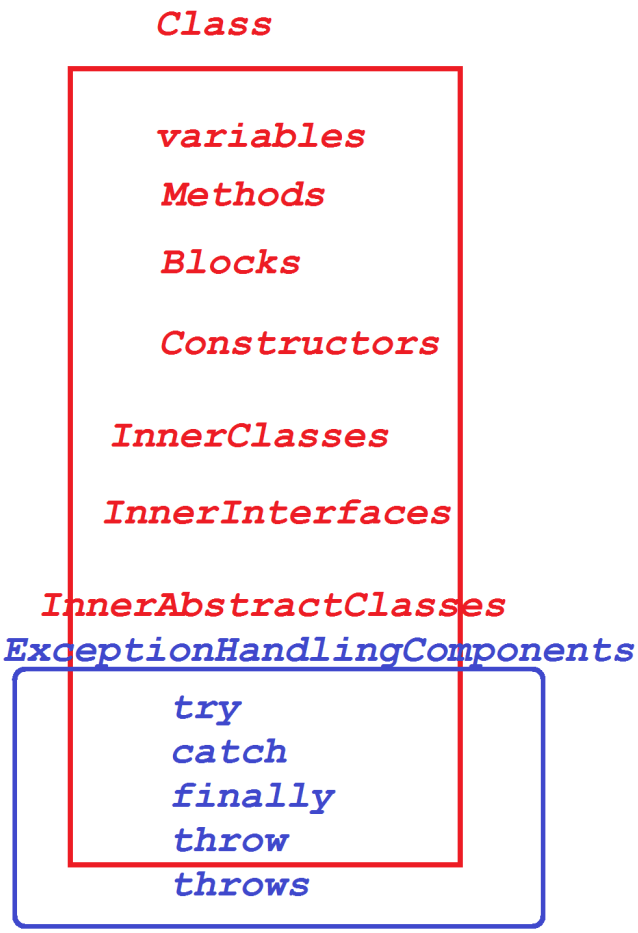
8.try-with-resource statement(Java7)

9.Enhanced try-with-resource statement(Java9)

10."java.lang.NullPointerException"

==========

Structure of Class in Java:



==========

*imp

PolyMorphism in Java:

=>The process in which programming components having more than one form is known as PolyMorphism.

Poly - Many

Morphism - Forms

Types of PolyMorphism:

=>PolyMorphism is categorized into two types:

1.Dynamic PolyMorphism

2.Static PolyMorphism

1.Dynamic PolyMorphism:

=>The PolyMorphism at execution stage is known as Dynamic PolyMorphism or Runtime PolyMorphism.

Ex:

Method Overriding process.

2.Static PolyMorphism:

=>The PolyMorphism at compilation stage is known as Static PolyMorphism or CompileTime PolyMorphism

Ex:

Method Overloading process

=>The Compiler at compilation stage will control the following keywords:

1.static

2.private

3.final

1.static:

=>"static" keyword in java specify the location of memory for Programming Components.

=>Static Components in Classes.

=>NonStatic Components in Objects.

=>The following are some important static programming Components:

(a)static variables

(b)static methods

(c)static blocks

(d)static classes

(e)static Interfaces

(f)static AbstractClasses

=>There is no concept of static Constructors in Java.

**imp*

2.private:

=>"private" is a access modifier and which specify the Programming Components must be accessed only inside the Class or Inside the Interface.

=>The following are the private programming components:

(a)private Variables

(b)private Methods

(c)private Constructors

(d)private Classes

=>There is no concept of Private Blocks,private Interfaces and Private abstract Classes.

(a)private Variables:

=>The variables which are declared with 'private' keyword are known as private variables.

Coding Rule:

=>Private variables are accessed only inside the Class,which means methods of same class can accesses private variables.

(b)private Methods:

=>The methods which are declared with 'private' keyword are known as private methods.

Coding Rule:

=>Private methods are accessed only inside the class,which means accessed by the NonPrivate methods of same class.

Note:

=>private method Overriding process is not posible,because private methods of PClass are not available to CClass.

=>But,we can perform private method Overloading process.

Ex:

p1 : PDisplay.java

```
package p1;
public class PDisplay
{
	private int a=10;
	private static int b=20;
	private void m1(int x)
	{
	System.out.println("****Instance private-
m1(x)****");
	System.out.println("The value x:"+x);
	}
	private static void m2(int y)
	{
	System.out.println("****static private-
m2(y)****");
	System.out.println("The value y:"+y);
	}
	public void dis()
	{
	System.out.println("****public-dis()****");
	System.out.println("The value a:"+a);
	System.out.println("The value b:"+b);
	this.m1(12);
	PDisplay.m2(13);
	}
}
```

p1 : CClass.java

```
package p1;
public class CClass extends PDisplay{
     private void m4(int p)
     {
       System.out.println("****CClass Instance private-
m4(p)****");
       System.out.println("The value p:"+p);
     }
     private void m4(int p,int q)
     {
       System.out.println("****CClass Instance private-
m4(p.q)****");
       System.out.println("The value p:"+p);
       System.out.println("The value q:"+q);
     }
     public void access()
     {
          this.m4(23);
          this.m4(24,25);
     }
}
```

p2 : DemoPoly1.java(MainClass)

```
package p2;
import p1.*;
public class DemoPoly1
{
     public static void main(String[] args)
     {
       PDisplay ob = new PDisplay();
       //System.out.println("The value a:"+ob.a);
       //System.out.println("The value b:"+PDisplay.b);
       ob.dis();
       CClass ob2 = new CClass();//Inheritance process
       ob2.access();
     }
}
```

o/p:

*****public-dis()*****

The value a:10

The value b:20

*****Instance private-m1(x)*****

The value x:12

*****static private-m2(y)*****

The value y:13

*****CClass Instance private-m4(p)*****

The value p:23

*****CClass Instance private-m4(p.q)*****

The value p:24

The value q:25

==========

**imp*

(c)private Constructors:

=>The constructor which is declared with "private" keyword is known as private Constructor.

Coding Rule:

=>when class is declared with private constructor,then the object must be created inside the class only.

(private Constructor will restrict the object creation from externally)

Ex:

p1 : Customer.java

```
package p1;
public class Customer
{
    public String name="Alex";
    public long phNo=9898981234L;
    private Customer() {}
    public static Customer ob = new
Customer();//Con_call
    public void getCustomer()
    {
         System.out.println("****CustomerDetails*****");
         System.out.println("CustomerName:"+name);
         System.out.println("CustomerPhoneNo:"+phNo);
    }
}
```

p2 : DemoPoly2.java(MainClass)

```
package p2;
import p1.Customer;
public class DemoPoly2
{
     public static void main(String[] args)
     {
       Customer c = Customer.ob;
       c.getCustomer();
     }
}
```

o/p:

****CustomerDetails*****

CustomerName:Alex

CustomerPhoneNo:9898981234

==========

(d)private Classes:

=>The classes which are declared with "private" keyword are known as private classes.

Coding Rule:

=>OuterClasses cannot be declared as Private classes.

=>These Private classes must be declared only as InnerClasses.

=>Private InnerClasses are executed using NonPrivate methods of same class.

p1 : SubClass1.java

```
package p1;
public class SubClass1
{
    private class SubClass2
    {
    public void m2(int x)
    {
         System.out.println("***m2(x)***");
         System.out.println("The value x:"+x);
    }
    }
    private static class SubClass3
```

```
    {
    public void m3(int y)
    {
         System.out.println("***m3(y)***");
         System.out.println("The value y:"+y);
    }
    }
    public void access(int x,int y)
    {
    SubClass2 ob2 = new SubClass2();
    SubClass3 ob3 = new SubClass3();
    ob2.m2(x);
    ob3.m3(y);
    }
}//OuterClass
```

p2 : DemoPoly3.java(MainClass)

```
package p2;
import p1.SubClass1;
public class DemoPoly3
{
    public static void main(String[] args)
    {
       SubClass1 ob1 = new SubClass1();//OuterClass
object
       ob1.access(11, 12);//OuterClass method call
    }
}
```

o/p:

m2(x)

The value x:11

m3(y)

The value y:12

==========