***Dt : 27/10/2022***

***\*imp***

***InnerInterfaces in Java:***

***(i)InnerInterfaces in Classes:***

***=>we can also declare InnerInterfaces in Classes and which can be***

***Static member InnerInterface or NonStatic member InnerInterface.***

***(ii)InnerInterfaces in Interfaces:***

***=>we can also declare InnerInterfaces in Interfaces and which are***

***automatically Static Member InnerInterfaces.***

***(iii)InnerInterfaces in AbstractClasses:***

***=>we can also declare InnerInterfaces in AbstractClasses and which can***

***be Static member InnerInterfaces or NonStatic member InterInterfaces.***

***Ex-program:***

***SubClass.java***

***package test;***

***public class SubClass {***

***public interface ITest1{***

***public abstract void m1(int x);***

***}//Instance member InnerInterface***

***public static interface ITest11{***

***public abstract void m11(int y);***

***}//Static member InnerInterface***

***}//OuterClass***

***ITest.java***

***package test;***

***public interface ITest {***

***public static interface ITest2{***

***public abstract void m2(int a);***

***}//Static member InnerInterface***

***}//OuterInterface***

***AClass.java***

***package test;***

***public abstract class AClass {***

***public interface ITest3{***

***public abstract void m3(int i);***

***}//Instance member InnerInterface***

***public static interface ITest33{***

***public abstract void m33(int j);***

***}//Static member InnerInterface***

***}//OuterAbstractClass***

***DemoInnerInterfaces.java(MainClass)***

***package maccess;***

***import test.\*;***

***public class DemoInnerInterfaces {***

***public static void main(String[] args) {***

***System.out.println("\*\*\*\*InnerInterface in Class\*\*\*\*\*");***

***SubClass.ITest1 ob1 = (int x)->***

***{***

***System.out.println("===m1(x)===");***

***System.out.println("The value x:"+x);***

***};***

***ob1.m1(12);***

***SubClass.ITest11 ob11 = (int y)->***

***{***

***System.out.println("===m11(y)===");***

***System.out.println("The value y:"+y);***

***};***

***ob11.m11(13);***

***System.out.println("\*\*\*\*InnerInterface in Interface\*\*\*\*\*");***

***ITest.ITest2 ob2 = (int a)->***

***{***

***System.out.println("===m2(a)===");***

***System.out.println("The value a:"+a);***

***};***

***ob2.m2(14);***

***System.out.println("\*\*\*\*InnerInterface in AbstractClass\*\*\*\*\*");***

***AClass.ITest3 ob3 = (int i)->***

***{***

***System.out.println("===m3(i)===");***

***System.out.println("The value i:"+i);***

***};***

***ob3.m3(15);***

***AClass.ITest33 ob33 = (int j)->***

***{***

***System.out.println("===m33(j)===");***

***System.out.println("The value j:"+j);***

***};***

***ob33.m33(16);***

***}***

***}***

***o/p:***

***\*\*\*\*InnerInterface in Class\*\*\*\*\****

***===m1(x)===***

***The value x:12***

***===m11(y)===***

***The value y:13***

***\*\*\*\*InnerInterface in Interface\*\*\*\*\****

***===m2(a)===***

***The value a:14***

***\*\*\*\*InnerInterface in AbstractClass\*\*\*\*\****

***===m3(i)===***

***The value i:15***

***===m33(j)===***

***The value j:16***

***=========================================================***

***\*imp***

***InnerAbstractClasses in Java:***

***(i)InnerAbstractClasses in Class:***

***=>we can also declare InnerAbstractClasses in Classes and which can be***

***Static member InnerAbstractClasses or NonStatic member InnerAbstractClasses***

***(ii)InnerAbstractClasses in Interfaces:***

***=>we can also declare InnerAbstractClasses in Interfaces and which are***

***automatically Static member InnerAbstractClasses.***

***(iii)InnerAbstractClasses in AbstractClasses:***

***=>we can also declare InnerAbstractClasses in AbstractClasses and which***

***can be Static member InnerAbstractClasses or NonStatic member InnerAbstract***

***Classes.***

***Ex:(Assignment)***

***====================================================================***

***Summary of Programming Components:(Java Alphabets)***

***(a)Variables***

***1.Primitive DataType variables(Values)***

***(i)Static***

***(ii)NonStatic***

***=>Instance***

***=>Local***

***2.NonPrimitive DataType variables(Object references)***

***(i)Static***

***(ii)NonStatic***

***=>Instance***

***=>Local***

***(b)Methods***

***1.Static methods***

***(i)pre-defined methods***

***(ii)User defined methods***

***2.Non-Static methods(Instance methods)***

***(i)pre-defined methods***

***(ii)User defined methods***

***(c)Blocks***

***1.Static blocks***

***2.NonStatic blocks(Instance blocks)***

***(d)Constructors***

***=>NonStatic Constructors***

***(e)Classes***

***1.static classes(Only InnerClasses)***

***2.NonStatic classes***

***(f)Interfaces***

***1.static Interfaces(Only InnerInterfaces)***

***2.NonStatic Interfaces***

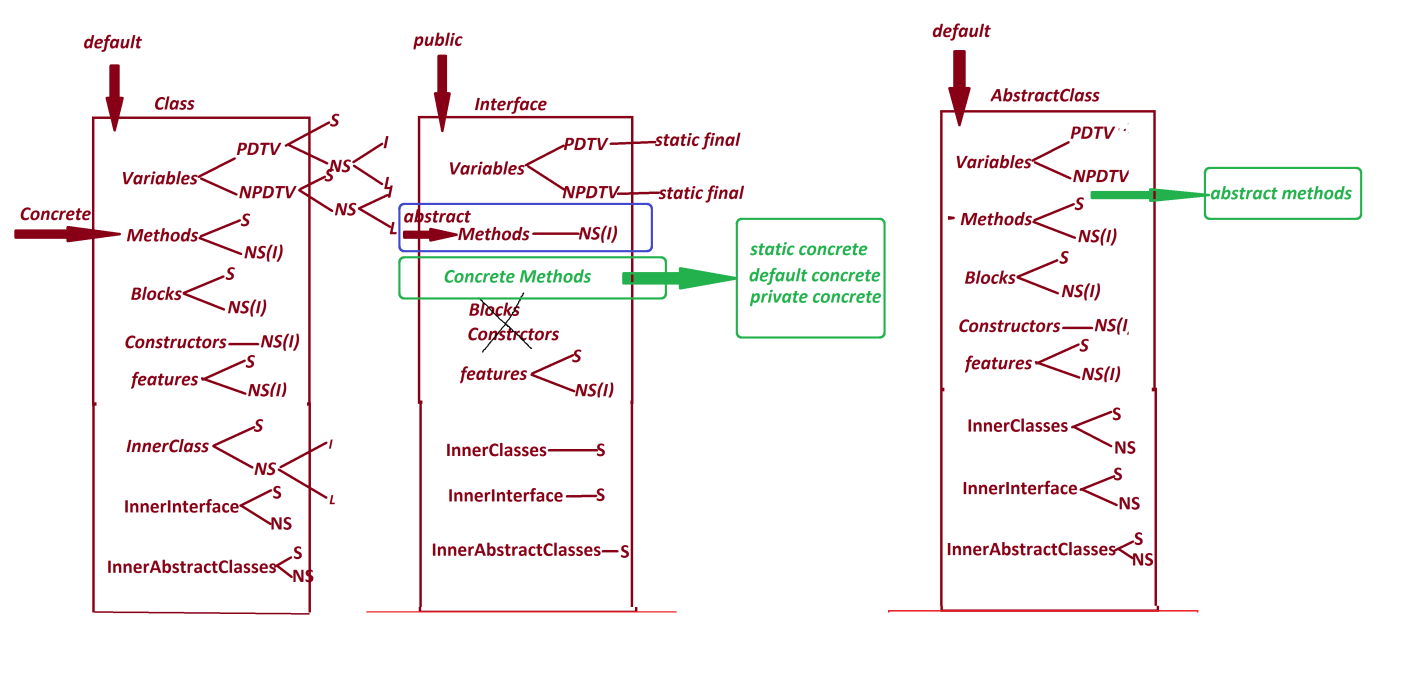
***(g)AbstractClasses***

***1.static abstract classes(Only InnerAbstractClasses)***

***2.NonStatic abstract classes***

***============================================================***

***Comparision:***

******

***==========================================================***

***\*imp***

***Exception Handling process:***

***define Exception:***

***=>The disturbance which is occured from the application is known as***

***"exception"***

***define Exception Handling process?***

***=>The process which is used to handle the exception is known as "Exception***

***handling process".***

***=>we use the following blocks in Exception Handling process:***

***1.try block***

***2.catch block***

***3.finally block***

***=>These blocks are executed automatically when the exception is raised.***

***1.try block:***

***=>try block will hold the statements which are going to raise the***

***exception.***

***syntax:***

***try***

***{***

***//statements***

***}***

***behaviour of try block:***

***=>when the exception is raised in try block,then one object is created for***

***Exception\_Type\_class and the object reference is thrown onto catch block.***

***2.catch block:***

***=>catch block will hold the object reference thrown from the try-block***

***and the required msg is generated from catch block.***

***syntax:***

***catch(Exception\_type\_class ref\_var)***

***{***

***//msg***

***}***

***3.finally block:***

***=>finally block is part of exception handling process but executed***

***independently without depending on exception.***

***=>In realtime finally block will hold resource closing operations like***

***IO close,file close,DB close,N/w close,...***

***syntax:***

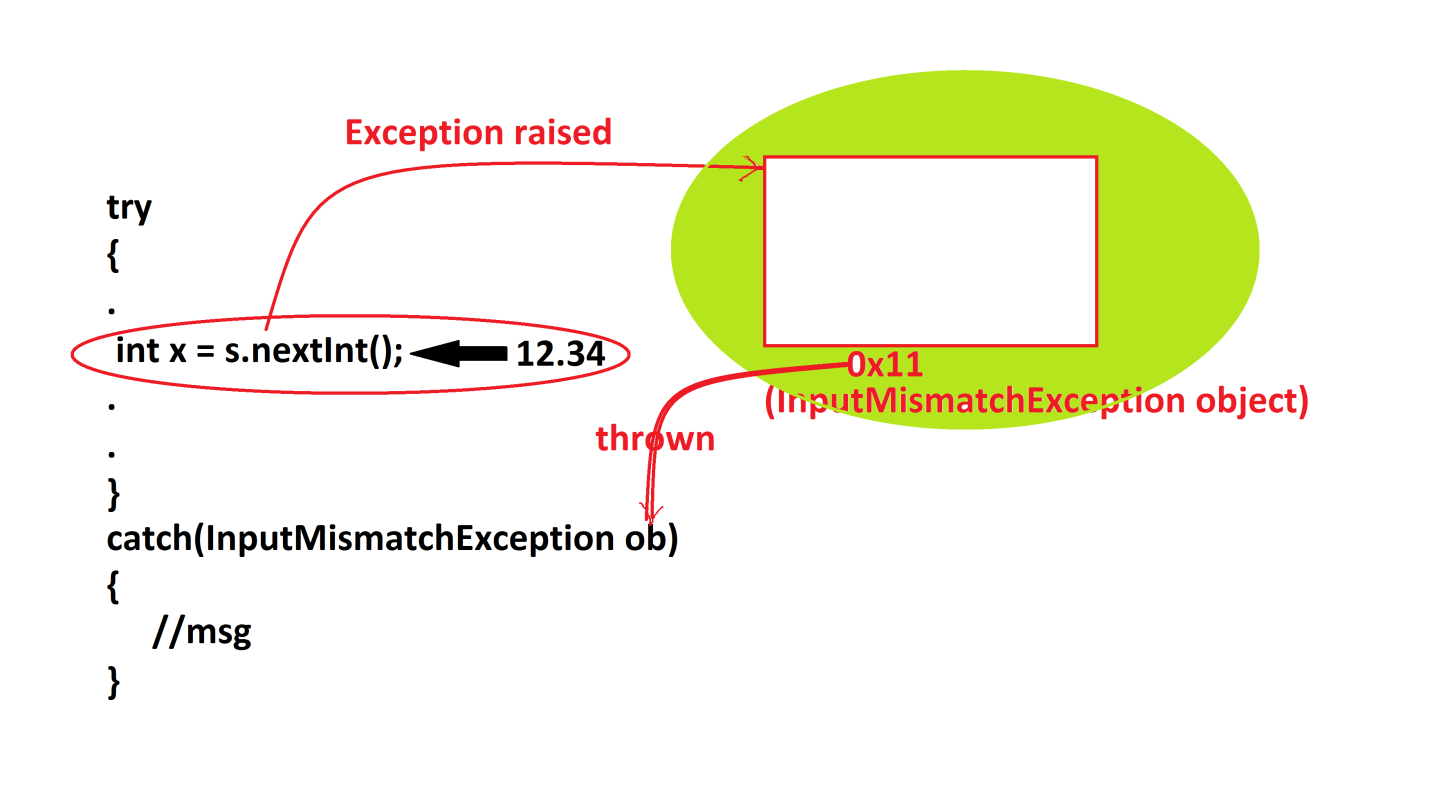
***finally***

***{***

***//statements***

***}***

***Diagram:***

******

***===================================================================***