3.13 Inner Classes

- **When you define a class within another class then it is called as inner class or nested class.**

Types of Inner classes

- Instance Inner classes
- Static Inner classes
- Local Inner classes
- Anonymous classes

Ex:

```
class  Outer{
      ...
       class  Inner{
        ...
     }
 }

class  Outer{
      ...
     static  class  Inner{
        ...
     }
 }

class  Outer{
      ...
class show(){
     class  Inner{
        ...
     }
 }
 }
```

3.13.1 Instance Inner Classes

- **When you define a class within another class without static modifier is called as instance inner class.**
- **Syntax to create instance inner class object from outside the outer class:**

```
Outer.Inner outob= new Outer().new Inner();
        or
Outer outob = new Outer();
Outer.Inner inob=outob. new Inner();
```

Lab499.java

```
class Outer{
int a=10;
static int b=20;
void m1(){
System.out.println("Outer - m1()");
}
static void m2(){
System.out.println("Outer - m2()");
}
class Inner{
int x=99;
final static int y=88;
void showIn(){
System.out.println("Inner - showIn()");
}
}
}

class Lab499{
public static void main(String as[]){
Outer myouter = new Outer();
myouter.m1();
myouter.m2();

//Inner in = new Inner();

Outer.Inner myinner1 = myouter.new Inner();
myinner1.showIn();

Outer.Inner myinner2 = new Outer().new
Inner();
myinner2.showIn();
}
}
```

Lab500.java

```
class Outer{
int a=10;
static int b=20;
void m1(){
System.out.println("Outer - m1()");
}
static void m2(){
System.out.println("Outer - m2()");
}
class Inner{
int x=99;
final static int y=88;
void showIn(){
System.out.println("Inner - showIn()");
System.out.println(x);
System.out.println(y);
System.out.println(a);
System.out.println(b);
m1();
m2();
}
}
}

class Lab500{
public static void main(String as[]){

Outer.Inner myinner = new Outer().new Inner();
myinner.showIn();

}
}
```

Lab501.java

```
class Outer{
int a=10;
static int b=20;

void m1(){
System.out.println("Outer - m1()");
Inner in=new Inner();
System.out.println(in.x);
System.out.println(in.y);
}

static void m2(){
System.out.println("Outer - m2()");
}

class Inner{
int x=99;
final static int y=88;
void showIn(){
System.out.println("Inner - showIn()");
m1();
m2();
}
}

}

class Lab501{
public static void main(String as[]){

Outer.Inner myinner = new Outer().new Inner();
myinner.showIn();

}
}
```

Lab502.java

```
class Hello{
int a=10;

class Hai{
int a=20;
void show(){
System.out.println("Hai - show()");
int a=30;
System.out.println(a);
System.out.println(this.a);
System.out.println(Hai.this.a);//Special
//System.out.println(super.a);
System.out.println(Hello.this.a);//Special
}
}

}

class Lab502{
public static void main(String as[]){

Hello.Hai in = new Hello().new Hai();
in.show();

}
}
```

3.13.2 Static Inner Classes

- When you define a class within another class with static modifier is called as static inner class.
- Syntax to create static inner class object from outside the outer class:

  Outer.Inner outob= new Outer.Inner();

Lab503.java

```
class Outer{
int a=10;
static int b=20;
void m1(){
System.out.println("Outer - m1()");
}
static void m2(){
System.out.println("Outer - m2()");
}
static class Inner{
int x=99;
static int y=88;
void show(){
System.out.println("Inner - showIn()");
}
}
}

class Lab503{
public static void main(String as[]){

Outer.Inner myinner = new Outer.Inner();
myinner.show();

}
}
```

Lab504.java

```
class Outer{
int a=10;
static int b=20;
void m1(){
System.out.println("Outer - m1()");
}
static void m2(){
System.out.println("Outer - m2()");
}
static class Inner{
int x=99;
static int y=88;
void show(){
System.out.println("Inner - showIn()");
System.out.println(x);
System.out.println(y);
//System.out.println(a);
System.out.println(b);
//m1();
m2();
}
}
}

class Lab504{
public static void main(String as[]){

Outer.Inner myinner = new Outer.Inner();
myinner.show();

}
}
```

Lab505.java

```
class Outer{
int a=10;
static int b=20;
void m1(){
System.out.println("Outer - m1()");
Inner in = new Inner();
in.show();
}
static void m2(){
System.out.println("Outer - m2()");
Inner in = new Inner();
in.show();
}
static class Inner{
int x=99;
static int y=88;
void show(){
System.out.println("Inner - showIn()");
}
}
}
```

```
class Lab505{
public static void main(String as[]){

Outer out = new Outer();
out.m1();
out.m2();

Outer.Inner myinner = new Outer.Inner();
myinner.show();


}
}
```

3.13.3 Local Inner Classes

- When you define a class within methods, blocks or constructors is called as local inner class.
- Local Inner class must be accessed from the same methods, blocks or constructors where it is defined.

Lab506.java

```
class Hello {

{
System.out.println("Hello - I.B");
class Inner{ }
}

static {
System.out.println("Hello - I.B");
class Inner{ }
}

Hello(){
System.out.println("Hello - Con");
class Inner{ }
}
```

```
void m1(){
System.out.println("Hello - I.M");
class Inner{ }

}
static void m2(){
System.out.println("Hello - S.M");
class Inner{ }
}
}

class Lab506{
public static void main(String as[]){
System.out.println("Hello Guys");
}
}
```

Lab507.java

```
class Hello {

int a=10;
static int b=20;

void m1(){
System.out.println("m1 - begin");
class Inner{
int x=10;
final static int y=20;
void show(){
System.out.println("Inner - show()");
System.out.println(x);
System.out.println(y);
System.out.println(a);
System.out.println(b);
}
}
Inner inner = new Inner();
inner.show();

System.out.println("m1 - end");
}

void m2(){
//Inner inner = new Inner();
}
}
```

```
class Lab507{
public static void main(String as[]){
//Inner inner = new Inner();
Hello h= new Hello();
h.m1();

}
}
```

3.13.4 Anonymous Classes

- When you define a class without specifying the name then it is called as Anonymous class.
- When you want to declare and instantiate a class at the same time then you can use Anonymous class.
- Anonymous class definitions can be placed anywhere inside class, blocks, methods or constructors.
- Anonymous class can be used to write the sub class of an abstract class or an interface.

Lab508.java

```
interface Shape{
void draw();
}

class Circle implements Shape{
public void draw(){
System.out.println("Circle - draw");
}
}

class Lab508{
public static void main(String as[]){

Shape shape1 = new Circle();
shape1.draw();

}
}
```

Lab509.java

```
interface Shape{
void draw();
}

class Lab509{
public static void main(String as[]){

Shape shape1 = new Shape(){
public void draw(){
System.out.println("Annonymous1 Shape - draw");
}
};

shape1.draw();

Shape shape2 = new Shape(){
public void draw(){
System.out.println("Annonymous2 Shape - draw");
}
};

shape2.draw();
}
}
```

Lab510.java

```
interface Shape{
void draw();
}

class Hello{
int a=99;
Shape myshape = new Shape(){
public void draw(){
System.out.println("Annonymous - draw");
}
};

}
```

```
class Lab510{
public static void main(String as[]){
Hello h= new Hello();
System.out.println(h.a);
System.out.println(h.myshape);
h.myshape.draw();

}
}
```

SUMMARY

Instance Inner classes

1) All the static and instance members of Outer class can be accessed inside the Instance inner class directly.
2) Instance Inner class members cannot be accessed from outer class directly.
3) Instance inner class members can be accessed from outer class with object of instance inner class.
4) You can use instance inner class name directly in Outer class.
5) You cannot use instance inner class name directly from outside the Outer class.
6) You can use following syntax to access instance inner class outside the outer class:
    1. Outer.Inner in = new Outer(). new Inner();

       or

    2. Outer out = new Outer();

       Outer.Inner in = out. new Inner();
7) Instance inner class cannot contain static declarations i.e. you cannot define static variables or static methods inside the instance inner class.
8) Outer class can be either default or public.
9) Inner class can be private or protected or public or static.
10) Inner class can be abstract and you can use private and abstract combination with inner class.
11) Inner class can extend another inner class or it can implement another inner interface.
12) Inner class can be nested i.e. you can write inner class inside another inner class.
13) An interface can contain another inner interfaces or inner classes.
14) You can use this keyword with Outer class name to access the members of Outer class when the same members available in inner class and locally.

Static Inner classes

15) Only static members of Outer class can be accessed inside the Static inner class directly.
16) Instance members of Outer class can be accessed with the Outer class object.
17) Static inner class can contain static declarations i.e. you can define static variables or static methods inside the Static inner class.
18) Static Inner class members cannot be accessed from outer class directly.
19) Static inner class members can be accessed from outer class with Class name or object of static inner class.
20) You can use static inner class name directly in Outer class.

21) You cannot use static inner class name directly from outside the Outer class.

22) Static inner class can contain inner interface.

23) You can use following syntax to access instance inner class outside the outer class:

```
Outer.Inner  inob= new  Outer.Inner();
```

Local Inner classes

24) When you are writing Local Inner Class inside the Instance method then you can access both instance and static members of Outer class.

25) When you are writing Local Inner Class inside the static method then you can access only static members of Outer class.

26) Local inner class cannot be static.

27) Local inner class cannot have static declarations.

Anonymous Classes

28) You cannot define constructor for anonymous class because of no name.

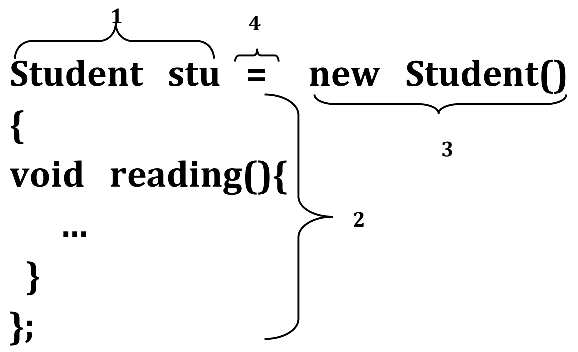
29) You cannot define reference variable of anonymous class.

30) Anonymous class will be always subclass of an interface or a class.

31) We cannot create more than one object of anonymous class.

32) We can store object of anonymous class in super type reference variable. You can access only those members with that reference variable whose signature is available in super type.

Object Creation Steps of Anonymous Class



```
Student  stu  =  new  Student()
{
void  reading(){
    ...
 }
};
```

1. Declaring reference variable of super class.
2. Writing anonymous class.
3. Crating object of anonymous class.
4. Storing that object to super class reference variable.

Inner class naming conventions

class A{
  class B{
    class C {}
  }
}
	class A{
	  static class B{
	    static class C {}
	  }
	}
		class A{
		{
		  class B{}
		  class C{}
		}
		void m1(){
		  class B{}
		  class D{}
		}
		}
			class A{ }
			class B{
			A ref1=new A(){};
			A ref2=new A(){};
			A ref3=new A(){};
			}

A$B$C.class A$B.class A.class	ABC.class A$B.class A.class	A$1B.class A$1C.class A$1D.class A$2B.class A.class	A.class B$1.class B$2.class B$3.class B.class

Assignment #14

Q1) What is inner class?

Q2) What is nested class?

Q3) What are the types of nested class available in java?

Q4) What is Instance inner class?

Q5) Can I access instance members of outer class from instance inner class directly?

Q6) Can I access static members of outer class from instance inner class directly?

Q7) Can I access members of inner class from outer class directly?

Q8) Can I define instance members in instance inner class?

Q9) Can I define static members in instance inner class?

Q10) How to create object of instance inner class from outside the outer class?

Q11) Can I declare final static variables inside the instance inner class?

Q12) What are the allowed modifiers for outer class?

Q13) What are the allowed modifiers for inner class?

Q14) Can I define inner interfaces?

Q15) What is static inner class?

Q16) Can I access instance members of outer class from static inner class directly?

Q17) Can I access static members of outer class from static inner class directly?

Q18) Can I define instance members in static inner class?

Q19) Can I define static members in static inner class?

Q20) How to create object of static inner class from outside the outer class?

Q21) What is local inner class?

Q22) Can I define local inner class inside initialization blocks?

Q23) Can I define local inner class inside constructors?

Q24) Can I define local inner class inside methods?

Q25) Can I create the object of inner class from outside the block where it is declared?

Q26) What is anonymous class?

Q27) What is the use of anonymous class?

Q28) Can I write constructors inside the anonymous class?

Q29) What will be name provided by Java compiler for anonymous class?

Practice Test #14

Q.No	Question	Options	Answer

1	class A{
	  class B{}
	}
	class B{}
		What are the class file generated
		
		A) Compilation Error
		B) A.class
		B.class
		A$B.class
		C) A.class
		B.class
		D) A$B.class
		A.class	

2	class A{
	  class B{}
	}
	class A$B{}
		What are the class file generated
		
		A) Compilation Error
		B) A.class
		B.class
		A$B.class
		C) A.class
		B.class
		D) A$B.class
		A.class	

3	class A{ }
	class A$B{
	int x=10;
	}
	
	class Test1{
	public static void main(String[] args){
	A$B ref=new A$B();
	System.out.println(ref.x);
	}
	}
		A) Compilation Error
		B) 10
		C) Runtime Error
		D) None of above	

4	class A{
	static void show(){
	B ref=new B();
	System.out.println(ref.xy);
	}
	
	class B{
	  int xy=99;
	 }
	}
	
	class Test4{
	public static void main(String[] args){
	new A().show();
	}
	}
		A) Compilation Error
		B) 99
		C) Runtime Error
		D) None of above	

5	class A{
	static void show(){
	B ref=new A().new B();
	System.out.println(ref.xy);
	}
	
	class B{
	  int xy=99;
	}
	}
	
	class Test5{
	public static void main(String[] args){
	new A().show();
	}
	}
		A) Compilation Error
		B) 99
		C) Runtime Error
		D) None of above	

6	class A{
	static int xy=99;
	static class B{
	static{
	System.out.println("B S.B");
	}
	}
	}
	class Test6{
	public static void main(String[] args){
	System.out.println(A.xy);
	}
	}
		A) Compilation Error
		B) B S.B
		99
		C) 99
		D) Runtime Error
		E) None of above	

7	class A{
	int x=99;
	class B{
	int y=88;
	}
	}
	class Test7{
	public static void main(String[] args){
	A ref=new A();
	System.out.println(ref.x);
	System.out.println(ref.y);
	}
	}
		A) Compilation Error
		B) 99
		88
		C) 88
		99
		D) Runtime Error
		E) None of above	

8	class A{
	int x=99;
	class B{
	int y=88;
	}
	}
	class Test8{
	public static void main(String[] args){
	A.B ref=new A().new B();
	System.out.println(ref.x);
	System.out.println(ref.y);
	}
	}
		A) Compilation Error
		B) 99
		88
		C) 88
		99
		D) Runtime Error
		E) None of above	

9

```
class A{
static int x=99;
static class B{
static int x=88;
void show(){
 System.out.println(x);
 System.out.println(A.this.x);
}
}
}

class Test9{
public static void main(String[] args){
A.B ref=new A.B();
ref.show();
}
}
```

A) Compilation Error
B) 99
   88
C) 88
   99
D) Runtime Error
E) None of above

10

```
class A{
static int x=99;
static class B{
static int x=88;
void show(){
 System.out.println(x);
 System.out.println(A.x);
}
}
}

class Test10{
public static void main(String[] args){
A.B ref=new A.B();
ref.show();
}
}
```

A) Compilation Error
B) 99
   88
C) 88
   99
D) Runtime Error
E) None of above

11	class A{
	static class B{
	static void show(){
	 System.out.println("OK");
	}
	}
	}
	
	class Test11{
	public static void main(String[] args){
	A.B.show();
	}
	}
		A) Compilation Error
		B) OK
		C) Runtime Error
		D) None of above	

12	interface In1{
	class B{
	static void show(){
	 System.out.println("OK");
	}
	}
	}
	
	class Test12{
	public static void main(String[] args){
	In1.B.show();
	}
	}
		A) Compilation Error
		B) OK
		C) Runtime Error
		D) None of above	

13	interface In1{
	class B{
	void show(){
	 System.out.println("OK");
	}
	}
	}
	
	class Test13{
	public static void main(String[] args){
	In1.B.show();
	}
	}
		A) Compilation Error
		B) OK
		C) Runtime Error
		D) None of above	