7-5-2020

FEBRUARY

S	M	T	W	T	F	S
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28

MARCH

S	M	T	W	T	F	S
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31				

Packages in Java

9th week

057-308

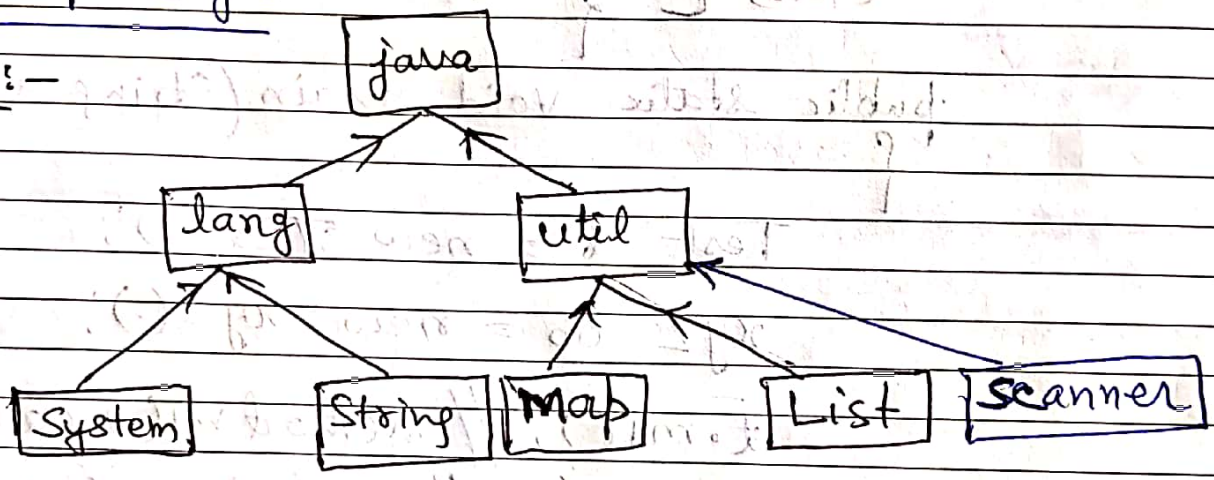
26

Thursday

Packages — It is a ~~Mas~~ namespace that organizes a set of related classes and interfaces.

It is good programming practice to keep things organized by placing related classes and interfaces into packages.

eg:-



Program to create a package:-

```
package mypack1;
class Test
{
    void m1()
    {
    }
}
```

```
package mypack1;
class xyz
{
    void m2()
    {
    }
}
```

mypack1

Test.class

xyz.class

Notes

Both the files should be stored into the same folder name similar to the package name.

Now, we can use the both classes in our program as follows:—

```
import mypack1.*;
class C {
public static void main(String args[])
{
    Test t = new Test();
    xyz ob = new xyz();
    t.m1(); // method m1() called
    ob.m2(); // method m2() called
}
}
```

Package:— Packages in java are used to prevent naming conflicts to control access etc.

or

A package can be defined as a grouping of related types (classes, interfaces etc). providing access protection and name space management.

Notes

Examples of existing packages in java are java.io, java.lang, java.awt, java.util, java.net, java.applet etc.

java.lang package has all fundamental classes and need not to be imported because this package is available by default in java run-time environment.

Example:—

```
package mypack;
public class c1
{ int a;
  public int b=10;
  private int c=5;
  protected int d;
  public void m1()
  {
  System.out.println(a+b+c+d);
  }
}
```

```
import mypack.*;
class packtest
{
public static void main(String a[])
{
c1 obj = new c1();
obj.m1();
}
}
```

o/p is:—

15

01 Sunday

Make a folder — mypack then save it with c1.java. go to the folder Compile.

Access Modifiers in Packages:— (Important)

	Private	No Modifier	Protected	Public
Same class	YES	YES	YES	YES
Same Package Sub-class	NO	YES	YES	YES
Same Package Non-subclass	NO	YES	YES	YES
Diff. Package Subclass	NO	NO	YES	YES
Diff. Package Non-Subclass	NO	NO	NO	YES

Can protected Access Modifier is accessible from outside the package ?

Ans— yes, but only in Inherited class.

Remember → All classes which has package can be stored in a single folder name similar to the name of the package in which it contain.

Example to See Accessibility in packages:—

```
package mypack;
public class pack1
{
    int a;
    public int b=10;
    private int c=5;
    protected int d;
public void m1()
{
System.out.println("Sum is :"+(a+b+c+d));
}
}
```

Store this file in mypack folder with name (pack1.java) then compile it only.

```
import mypack.*;
class packtest extends pack1
{
public static void main(String args[])
{
```

```
packtest obj = new packtest();
obj.b = 25;
obj.d = 6;
obj.m1();
}
}
```

output will be:-

Sum is: 36

Store this file in parent folder of mypack with name packtest.java

eg:-

Suppose you have —

```
D:\Java\mypack> cd..
D:\Java> javac packtest.java
D:\Java> java packtest
```

04 Wednesday

10th week 063-302

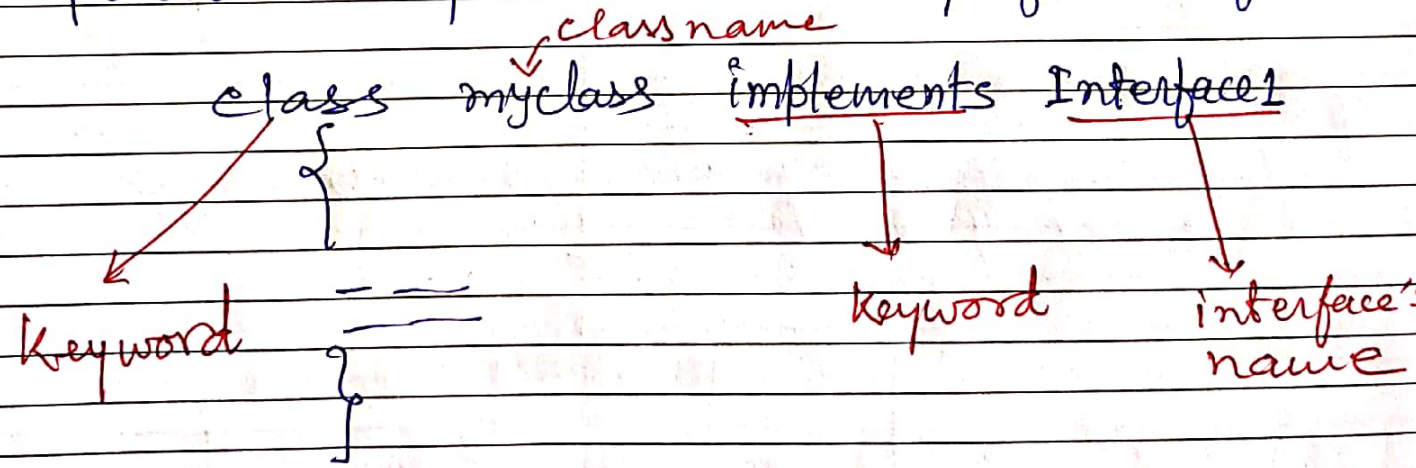
March

Interface :— An interface is a collection of abstract methods. Any class can implements the interface. It contains behaviour that a class implements.

→ One important thing is that unless the class that implements the interface is abstract, all the methods of the interface need to be defined in the class.

In other words, a class must have to implement all the methods of the interface from which it is implemented.

→ In java, multiple inheritance can be possible only with the help of interface.



If Interface1 has 3 methods, then myclass has to define all of them.

But if we put abstract before class then we need not to define all of them.

Notes

Example of multiple inheritance using interfaces:-

```
//i1.java
interface i1
{ public void m1();
  int i=10;
}

interface i2
{
  public void m1();
  int i=20;
}

class interface1 implements i1,i2
{
  public static void main(String
                          args[])
  {
  interface1 obj = new interface1();
  obj.m1();
  }
public void m1()
  {
  System.out.println(i1.i);
  System.out.println(i2.i);
  }
}
```

Output will be:-

10
20

In above program, the value of variable i in both interfaces are final and it cannot be changed in method m1() implemented inside the class interface1.

To resolve the ambiguity, we have to use interface name before variable i to get the proper values of i — i1.i and i2.i

10th week
065-300

March

06 Friday

Similarity between Interface & Classes

1. Interface can any no of methods.
2. .java extension.
3. Bytecode in .class file.
4. Interface appear in packages

Difference between Interface and classes

1. Interface cannot be instantiated
2. No constructor
3. All the methods of interface are abstract.
4. Interface is implemented by a class.
5. An interface can extend multiple interfaces.
6. Cannot contain instance fields.
7. All the variables of interface are static and Final.