```
Packages
================
Information regarding packages:-
1) The package contains group of related classes and interfaces.
2) The package is an encapsulation mechanism it is binding the
related classes and interfaces.
3) We can declare a package with the help of package keyword.
4) Package is nothing but physical directory structure and it is
providing clear-cut separation between the project modules.
5) Whenever we are dividing the project into the packages(modules)
the sharability of the project will be increased.
Syntax:-
Package package_name;
Ex:- package com.klu;


The packages are divided into two types
1) Predefined packages
2) User defined packages
Predefined packages:-
============================
The java predefined packages are introduced by sun peoples these
packages contains predefined classes and interfaces.
Ex:- java.lang
java.io
java.awt
java.util
java.net………………………..etc
java.lang:-
=============
The most commonly required classes and interfaces to write a sample
program is encapsulated into a separate package is called java.lang
package.
 Ex:- String(class)
    StringBuffer(class)
    Object(class)
    Runnable(interface)
    Cloneable(nterface)
Note:-
the default package in the java programming is java.lang if we are
importing or not importing by default this package is available for
our programs.
java.io package:-
=====================
The classes which are used to perform the input output operations
that are present in the java.io packages.
Ex:- FileInputStream(class)
FileOutputStream(class)
FileWriter(class)
```

```
FileReader(class)
java.net package:-
==================
The classes which are required for connection establishment in the network that classes are present in the java.net package.
Ex:- Socket
ServerSocket
InetAddress
URL
java.awt package:-
The classes which are used to prepare graphical user interface those classes are present in the java.awt package.
Ex: Button(class)
Checkbox(class)
Choice(Class)
List(class)
User defined packages:-
1) The packages which are declared by the user are called user defined packages.
2) In the single source file it is possible to take the only one package. If we are trying to take two packages at that situation the compiler raise a compilation error.
3) In the source file it is possible to take single package.
4) While taking package name we have to follow some coding standreds.
Whenever we taking package name don't take the names like pack1, pack2, swamy, sri……… these are not a proper coding formats.
Rules to follow while taking package name:-(not mandatory but we have to follow)
1) The package name is must reflect with your organization name. the name is reverse of the organization domain name.
Domain name:- www.kluniversity.com
Package name:- package com.kluniversity;
2) Whenever we are working in particular project(Bank) at that moment we have to take the package name is as follows.
Project name :- Bank
package :- package com.kluniversity.Bank;
3) The project contains the module (deposit) at that situation our package name should reflect with the module name also.
Domain name:- www.kluniversity.com
Project name:- Bank
Module name:- deposit
package name:- Package com.kluniversity.bank.deposit;

For example the source file contains the package structure is like this:-
package com.kluniversity.bank.deposit;
Note:-
If the source file contains the package statement then we have to compile that program with the help of fallowing statements.
```

```
D:\>javac -d . Test.java
```

After compilation of the code the folder structure is as shown below.

```
D:\KLUniversity\OOPCourse\mypack>javac -d . KluTest.java

D:\KLUniversity\OOPCourse\mypack>tree/f
Folder PATH listing for volume Windows8Rama
Volume serial number is CEEB-6FD6
D:.
│   A.java
│   KluTest.java
│
│   MyTest1.java
│
│   PubTest.java
│
│   ReadData.java
│
│   Test.class
│   Test.java
│
│   Testing.java
│
│
├───com
│   └───kluniversity
│       └───bank
│           └───deposit
│                   KluTest.class
│
└───klu
    ├───accestest
    │       PubTest.class
    │
    ├───cse
    │   └───mytest
    │           Testing.class
    │
    └───eee
        └───java
                ReadData.class
```

Note :-
If it a predefined package or user defined package the packages contains number of classes.
Ex 1:-
keyword
Reverse of domain name

```
Project name
Module name
Java compiler
Tells to compiler to create separate directory structure
Place the directory structure in current working folder(D:\)
Java source file name

package com.kluniversity.bank.deposit;
class KluTest
{}{
    public static void main(String[] args)
    {}{
        System.out.println("Welcome to KL University");
    }
}
Compilation : javac -d . KluTest.java
├───com
│   └───kluniversity
│       └───bank
│           └───deposit
│                   KluTest.class
Execution :java com.kluniversity.bank.deposit.KluTest
output:
Welcome to KL University
------------------------------------------------------
Ex:- (compilation error)
package com.klu.bank.deposit;
package com.klu.online.corejava;
class Test
{}{
public static void main(String[] args)
{}{
System.out.println("package example program");
}
}
Reason:-
Inside the source file it is possible to take the single package not
possible to take the multiple packages.
-------------------------------------------
Ex 2:-
package com.klu.OnlineExam.corejava;
class Test
{}{
public static void main(String[] args)
{}{
System.out.println("package example program");
}

}
```

```
class A
{}{
}
class B
{}{
}
class C
{}{
}
Compilation :- javac -d . Test.java
Com
|
|------klu
|
|-------OnlineExam
|
|----corejava
|
|--------Test.class
|--------A.class
|--------B.class
|--------C.class
Execution :- java com.klu.onlineexam.Test
Note:-
The package contains any number of .classes the .class files
generation totally depends upon the number of classes present on the
source file.
import session:-
The main purpose of the import session is to make available the java
predefined support into our program.
Predefined packages support:-
Ex1:-
import java.lang.String;
String is a predefined class to make available predefined string
class to the our program we have to use import session.
Ex 2:-
Import java.awt.*;
To make available all predefined class present in the awt package
into our program. That * represent all the classes present in the awt
package.

User defined packages support:-
I am taking two user defined packages are
1) package pack1;
class A
{}{
}
class B
{}{
}
2) package pack2
class D
{}{
}
```

Ex 1:-

Import pack1.A;

A is a class present in the pack1 to make available that class to the our program we have to use import session.

Ex 2:-

Import pack1.*;

By using above statement we are importing all the classes present in the pack1 into our program. Here * represent the all the classes.

Note:-

If it is a predefined package or user defined package whenever we are using that package classes into our program we must make available that package into our program with the help of import statement.

Public:-

==============

This is the modifier applicable for classes, methods and variables (only for instance and static variables but not for local variables).

--If a class is declared with public modifier then we can access that class from anywhere (within the package and outside of the package).

--If we declare a member(variable) as a public then we can access that member from anywhere but Corresponding class should be visible i.e., before checking member visibility we have to check class visibility.

Ex:-

```
public class Test // public class can access anywhere
{}{
public int a=10; //public variable can access any where
public void m1() //public method can access any where
{}{
System.out.println("public method access in any package");
}
public static void main(String[] args)
{}{
Test t=new Test();
t.m1();
System.out.println(t.a);

}
}
```

default:-

==============

This is the modifier applicable for classes, methods and variables (only for instance and static variables but not for local variables).

If a class is declared with <default> modifier then we can access that class only within that current package but not from outside of

the package.
Default access also known as a package level access.
The default modifier in the java language is default.
Ex:-

```
class Test
{}{
void m1()
{}{
System.out.println("m1-method");
}
void m2()
{}{
System.out.println("m2-method");
}
public static void main(String[] args)
{}{
Test t=new Test();
t.m1();
t.m2();
}
}
```

Note :-
in the above program we are not providing any modifier for the methods and classes at that situation the default modifier is available for methods and classes that is default modifier. Hence we can access that methods and class with in the package.
Private:-
==============
private is a modifier applicable for methods and variables.
If a member declared as private then we can access that member only from within the current class.
If a method declare as a private we can access that method only within the class. it is not possible to call even in the child classes also.

```
class Test
{}{
private void m1()
{}{
System.out.println("we can access this method only with in this class");
}
public static void main(String[] args)
{}{
Test t=new Test();

t.m1();
}
};
```

Protected :-

===================
If a member declared as protected then we can access that member with in the current package anywhere but outside package only in child classes.
But from outside package we can access protected members only by using child reference. If we try to use parent reference we will get compile time error.
--Members can be accesses only from instance area directly i.e., from static area we can't access instance members directly otherwise we will get compile time error.
Ex:-demonstrate the user defined packages and user defined imports.
klu project source file:-
=======================

```
package com.klu;
public class StatesDemo
{}{
public void ap()
{}{
System.out.println("ANDHRA PRADESH");
}
public void tl()
{}{
System.out.println("TELENGANA");
}
public void tn()
{}{
System.out.println("TAMILNADU");
}
}
```

Tcs project source file:-

```
package com.tcs;
import com.klu.StatesDemo;//
public class StatesInfo
{}{
public static void main(String[] args)
{}{
StatesDemo sd=new StatesDemo();
sd.ap();
sd.tl();
sd.tn();
}
}
```

```
Step 1 :- javac -d . StatesDemo.java
Step 2 :- javac -d . StatesInfo.java
Step 3 :- java com.tcs.StatesInfo
```

Static import:-
1) this concept is introduced in 1.5 version.
2) if we are using the static import it is possible to call static

variables and static methods directly to the java programming.

Ex:-without static import

```
import java.lang.*;
class Test
{}{
public static void main(String[] args)
{}{
System.out.println("Hello World!");
}
}
```

Ex :- with static import

```
import static java.lang.System.*;
class Test
{}{
public static void main(String[] args)
{}{
out.println("Hello world");
}
};
```

```
Ex:-package com.klu;
public class Test
{}{
public static int a=100;
public static void m1()
{}{
System.out.println("m1 method");
}
};
```

Ex:-

```
package com.tcs;
import static com.dss.Test.*;
class Test1
{}{
public static void main(String[] args)
{}{
System.out.println(a);
m1();
}
}
```

Source file Declaration rules:-

The source file contains the fallowing elements

1) Package declaration---?optional-----?at most one package(0 or 1)--?1st statement

2) Import declaration-----?optional-----?any number of imports--------?2nd statement

3) Class declaration--------?optional-----?any number of classes---------?3rd statement

4) Interface declaration---?optional----?any number of interfaces-----?3rd statement

```
5) Comments declaration-?optional----?any number of comments----?3rd statement
a. The package must be the first statement of the source file and it is possible to declare at most one package within the source file .
b. The import session must be in between the package and class statement. And it is possible to declare any number of import statements within the source file.
c. The class session is must be after package and import statement and it is possible to declare any number of class within the source file.
i. It is possible to declare at most one public class.
ii. It is possible to declare any number of non-public classes.
d. The package and import statements are applicable for all the classes present in the source file.
e. It is possible to declare comments at beginning and ending of any line of declaration it is possible to declare any number of comments within the source file.
---------------------------------------------------------------------

package klu.eee.java;

import java.util.Scanner;
public class ReadData
{}{
private int a,b,c;
public void add(int a,int b)
{}{
c=a+b;
System.out.println("Result="+c);
}
public void sub(int a,int b)
{}{
c=a-b;
System.out.println("Result="+c);
}


}
//Testing Program
package klu.cse.mytest;
import java.util.*;
import klu.eee.java.ReadData;
class Testing
{}{
public static void main(String[] args)
{}{
int x,y;
Scanner s=new Scanner(System.in);
System.out.println("Enter Two Numbers");
x=s.nextInt();
y=s.nextInt();

ReadData data=new ReadData();
data.add(x,y);
data.sub(x,y);
}
}
output:
D:\KLUniversity\OOPCourse\mypack>java klu.cse.mytest.Testing
Enter Two Numbers
40
30
Result=70
Result=10
-----------------------------------------------------------------------
----
```