Unit 1: Package   1

Unit Structure

1.1 LEARNING OBJECTIVE

After studying this unit student should be able to:

- Understand need of package in java.
- Know how to create package
- Know about classpath and method to set classpath.
- Understand the jar file, how it is created and its usage.
- Utilize import statement
- Study various built in packages

1.2 INTRODUCTION

In java package can be used to create a group of classes and interfaces in same category based on their functionality. They provide the access protection and namespace. Actually package is a folder which contains other package folder or the list of class or interface files which are part of that package. when we need to use some classes or interfaces more than one place we may put them in a package and reuse them importing package when needed.

The package can be built in package or user define package.

java.lang, java.io, java.util etc are the example of built in packages. They are also called API (Application Programming Interface)

User defined package is created by the user whenever required.

The following are the benefits of using packages.

- Reuse the class
- Create a category of classes and interface
- Same class name can be use in different packages.
- Control the access of class variables and methods.

1.3 DEFINING PACKAGE

For creating a package a **package** keyword is used in a java file following a space and a package name. Package name can be any variable name. The java file in which we have declared the package should contain all the class and interface

definitions which we want to put in the package. Also package declaration must be the first line of the java program file. We also have to store this java file in the folder with same name as package name. To compile the java file which has package declaration in it we have to move to parent directory of package folder. Make sure that your java file must be in package folder. For compiling we have to use " **javac package_name\program_file.java**". This will create appropriate classes in package folder.

We can also define a package within a package. The inner package is called sub package.

For example,

➢ **Step 1 :**

Create directory with package name let say mypack in your current directory.

Move to mypack folder.

Create a java file MyClass.java as follows,

```
package mypack;  //package declaration
class Hello{
public void sayHello ( String nm )
{
  System. out. println ( " hello from " + nm );
}
}
public class MyClass
{
public static void main ( String args[] )
{
  Hello h1=new Hello();
  h1.sayHello( " Aryu " );
}
}
```

This will compile MyClass.java file and create a MyClass.class and Hello.class files in the directory mypack.

OR

We can create a MyClass.java file in current directory same as step 1. And comile it using following command
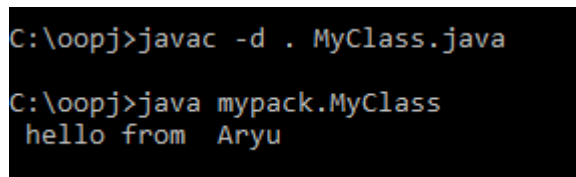
javac -d . MyClass.java

This command will compile the MyClass.java file and –d option will create a folder name with package name in . (current working) directory and put MyCLass.class file in that directory. i. e. the –d option will automatically create a package folder in folder mention after –d option and our class file will be stored in that folder.

> **Step 2:**

The program can be compiled using following command

java mypack.MyClass


```
C:\oopj>javac -d . MyClass.java

C:\oopj>java mypack.MyClass
 hello from  Aryu
```

Figure-53 Output of program

1.4 UNDERSTANDING CLASSPATH

CLASSPATH is an environment variable. To view value of current CLASSPATH variable we can use echo command like following,

 echo %CLASSPATH%

CLASSPATH variable is used to search for location of class file used to run java program by java compiler and JVM. JVM use this variable to search compiled classes. We can assign a path of folder containing class file or a path of jar file as a value of classpath variable. It can store path of multiple folders or jar file. Each path should be separated by ; symbol. We can assign value to classpath using following command,

set CLASSPATH = %CLASSPATH%;c:\oopj

In above command we have added path of oopj folder in existing classpath value.

We can also assign path of a jar (java archive) file in classpath. Jar file is an archive which stores all class file as one file. We can create a package consists of various classes and create a jar file for that package using following command,

jar cf pack.jar pack

This will create a pack.jar file for package pack.

For example,

We want to create three class in our package mypack. For this we have to create three java files each for one class (A.java, B.java, and C.java) file in current directory and comile them using following command

javac -d . A.java
javac -d . B.java
javac -d . C.java

This command will compile all java files and –d option will create a folder name with package name in . (current working) directory and put all class files in that directory. i. e. the –d option will automatically create a package folder in folder mention after –d option and our class files will be stored in that folder.

A.java

```
package mypack;  //package declaration


public class A
{
int a;
A() {a = 0; }
A( int x ) { a = x; }
```

```
package mypack;  //package declaration
public class B
{
int b;
B() {b = 0; }
B( int y ) { b = y; }
void printB() { System. out. println ( " b = " + b); }
}
```

```
package mypack;  //package declaration


public class C
{
int c;
C() {c = 0; }
C( int z ) { c = z; }
void printC() { System. out. println ( " c = " + c); }
}
```

After compilation the mypack folder is created with three class file in it. Now to create a jar file we have to execute following command,

jar cf pack.jar mypack

To include this package in class path we can use the pack.jar file as follows,

set CLASSPATH = %CLASSPATH%;c:\oopj\pack.jar

Now, we can use these three classes in all our java files by importing them in program.

1.5 ACCESS PROTECTION

The purpose of creating package is to encapsulate the similar classes as a container of classes, interfaces and sub packages. Classes and interfaces act as a container of member variables and member functions. In java we can have four categories of regarding the access of class members and classes among packages.

These categories are,

1)   Subclass of a class within same package i.e. class and subclass are in same folder.

2)   Independent classes of the same package

3)   Class and subclass both stored in different package

4)   Independent classes which are not stored in same package

There are mainly four access modifiers use to set the protection of the class members within the package or outside the package. They are friendly, private, protected and public.

We have already discuss them with example in section 4.2 of Block-1 chapter 4 (recall the Table 8). The private member of the class cannot be access anywhere outside the class whereas public member of the class can be accessed from everywhere. The protected member of a class can be accessed within the class as well as the subclass inside or outside the package.

The class can also be declared either friendly (default) or public. The default access allows calss to be used within the package only. The public class can be used inside as well as outside of the package.

For Example:

```
// MyPack1_A1.java

package mypack1;

public class MyPack1_A1 {

int a ;

private int pri_a ;

protected int pro_a ;

public int pub_a ;

public MyPack1_A1() {

System. out. println ("Base class constructor called");

a = 0;

pri_a = 0;

pro_a = 0;

pub_a = 0;

}

public MyPack1_A1(int w, int x, int y, int z) {

System. out. println ("Base class constructor called");
```

```java
a = w;

pri_a = x;

pro_a = y;

pub_a = z;

}

public printA1()

{

System. out. println  ( a );

System. out. println  ( pri_a );

System. out. println  ( pro_a );

System. out. println  ( pub_a );

}

}
```

This class MyPack1_A1 is in package mypack1 and has four data members which are of default, private, protected and public type.

```java
// MyPack1_A2.java

 package mypack1;

class MyPack1_A2 extends MyPack1_A1 {

public MyPack1_A2() {

System. out. println ("Derived class constructor called");

a = 0;

pri_a = 0; // error1

pro_a = 0;

pub_a = 0;

}

public MyPack1_A2(int w, int x, int y, int z) {
```

```
System. out. println ("Derived class constructor called");

a = w;

pri_a = x;   //error2

pro_a = y;

pub_a = z;

}

public printA2()

{

System. out. println  ( a );

System. out. println  ( pri_a ); //error3

System. out. println  ( pro_a );

System. out. println  ( pub_a );

}

}
```

The class MyPack1_A2 also belongs to mypack1 package and is a subclass of MyPack1_A1 class which is in the package mypack1. In MyPack1_A2 we can access member variables a, pro_a and pub_a of class MyPack1_A1. We can not access pri_a of MyPack1_A1 into MyPack1_A2. Hence if we compile above code it gives three error shown as a comment in the code. That is because we can not access private member of a class out side the class. We can access default member because both class are in the same package. We can access protected members because the second class is a subclass of first class.

```
//MyPack1_B

package mypack1;

class MyPack1_B {

MyPack1_B() {

MyPack1_A1 x = new MyPack1_A1( 1, 2, 3, 4);

System. out. println (" non subclass but same package class ");
```

```
System. out. println ("a = " + x.a);

System. out. println ("pri_a = " + x.pri_a);  // error1

System. out. println ("pro_a "+ x.pro_a);  //error2

System. out. println ("pub_a = " + x.pub_a);

}

}
```

The above class MyPack1_B also belongs to package mypack1. This class is not subclass of any class of the package mypack1. The MyPack1_B class creates an instance of MyPack1_A1 and tries to access all members of the class MyPack1_A1 using its object. Here, we can access only default and public members because default members can be accessed within classes of the same package and public members can be accessed everywhere. We can not access private outside the class. And hence class MyPack1_B is not subclass of MyPack1_A, we can not access protected members. Hence we got two error while compilation of above code.

Now we are creating package mypack2 and two classes in it one is subclass of a class of mypack1 and the other is non subclass.

```
package mypack2;

class MyPack2_A1 extends MyPack1_A1 {

MyPack2_A1() {

System. out. println ("derived class of mypack1 package constructor

called");

 System. out. println ("a = " + a);   //error 1

System. out. println ("pri_a = " + pri_a);   //error 2

System. out. println ("pro_a = " + pro_a);

System. out. println ("pub_a = " + pub_a);

}

}
```

The above class MyPack2_A1 we can not access Private and default members of MyPack1_A1. This is because private members can access within a class only and default members can accessed within package only. Here MyPack2_A2 is not in the same package. Hence we got error 1 and error 2 while compiling above code. We can access protected and public members of a class MyPack1_A1 because MyPack2_A1 is a subclass of it.

```
//MyPack2_ B


package mypack2;

class MyPack2_B {

MyPack2_B() {

mypack1.MyPack1_A1 co = new mypack1.MyPack1_A1 ();

System. out. println ("independent class of the other package constructor");

System. out. println ("a ,= " + co.a);   //error1

System. out. println ("pri_a = " + co.pri_a);   //error 2

System. out. println ("pro_a = " + co.pro_a);  //error 3

System. out. println ("pub_a = " + co.pub_a);

}

}
```

The above class MyPack2_B belongs to package mypack2. In the constructor of this class an instance of MyPack1_A1 is created. Using the instance of MyPack1_A1 we tries to access all member variables of the class. We can only access public members of a class which belongs to outside package and not a parent class of our class. Hence we got error 1 , error 2 and error 3, as we try to access default, private and protected member of the class respectively.

1.6 IMPORTING PACKAGE

In C/C++ to use library function we must include the header file congaing that library function in our program. Similarly in java to use the classes and their members we should import the package and class in our program. Once a package is created with all its member classes, we can use those classes in our java program by import statement. The syntax of import statement is given below,

import package_name.class_name;

OR

Import package_name.sub_package.class_name;

We can use the class class_name of package package_name or package_name.sub_package in our program using above import statement in our program.

For example,

import mypack1.MyPack1_A1;

allow us to use MyPack1_A1 class in our program. If we want to use all classes of a package we can use following,

import mypack1.*;

For importing package, the package folder must be set as a classpath OR the jar file for that package must be created and path of that jar file must be added in classpath.

> **Different ways of using package**

We may use class of a package using following ways,

class MyClass
{
mypack1.MyPack1_A1 x = new mypack1.MyPack1_A1() //fully qualified name
....
}

OR

import mypack1.*;
class MyClass

```
{
MyPack1_A1 x=new MyPack1_A1();
.....
}
OR

import mypack1.Mypack1_A1;
class MyClass
{
MyPack1_A1 x=new MyPack1_A1();
.....
}
```

➢ **Static import**

In java static import allows us to use static members of class directly (without using class name). It reduce the coding when we need to access static members more frequently.

For example,

```
import static java.lang.Math.*;
import static java.lang.System.*;
class ExStIm
{
public static void main String args [] )
{
out.println(" Hello ");
out.println(" 2 power 4 is " + pow ( 2, 4 ));
}
}
```

➢ **Name Collision**

While developing java application sometimes same class name needs to be used for different classes created for different purpose. The package allows you to create different class with same name but in different packages.

We can create class with name A in packages ABC and ACD both. To use such class, we have to use package_name.class_name.

For example,

ABC.A   // refer to class A of ABC package

ACD.A   // refer to class A of ACD package

Example of package and import,

```java
// Apple.java
package fruit;
public class Apple
{
        int id;
        String color;
        String shape;
        public Apple()
        {
        id = 0;
    color = "";
      shape = "";
      }
        public Apple(int i, String c, String s)
        {
        id = i;
    color = c;
      shape = s;
      }
        public void printApple()
        {
        System. out. println (" ID : " + id);
        System. out. println (" Color : " + color);
        System. out. println (" Shape : " + shape);
        }
```

```java
}

// Grape.java
package fruit;
public class Grape
{
        int id;
        String color;
        int size;
        public Grape ()
        {
        id = 0;
    color = "";
        size = 0;
        }
        public Grape (int i, String c, int s)
        {
        id = i;
    color = c;
        size = s;
        }
        public void printGrape()
        {
        System. out. println (" ID : " + id);
        System. out. println (" Color : " + color);
        System. out. println (" Size : " + size);
        }

}

// Banana.java
package fruit;
public class Banana
```

```java
{
    int id;
    String color;
    String unit;
    public Banana ()
    {
    id = 0;
color = "";
    unit = "";
    }
    public Banana (int i, String c, String u)
    {
    id = i;
color = c;
    unit = u;
    }
    public void printBanana()
    {
    System. out. println (" ID : " + id);
    System. out. println (" Color : " + color);
    System. out. println (" Unit : " + unit);
    }

}
```

Figure-54 Compiling java files

Figure-55 Setting class path

```
//ExPack.java
import fruit.Apple;
import fruit.Grape;
import fruit.Banana;

public class ExPack{
public static void main ( String args [] )
{
Apple a = new Apple (1, "red", "round" );
Grape g = new Grape ( 2, "green", 55 );
Banana b = new Banana ( 3, "yellow", "dozon" );
a. printApple ();
g. printGrape ();
b. printBanana ();
}
}
```

Figure-56 Output of program

1.7 BUILT IN PACKAGES

Built in packages are the readily available packages in java which can directly be used while programming using java. They have a readily available classes and interfaces. They are also called APIs (Application programming interfaces) or Library packages.

For example,

➢ **java.lang**

it is by default imported in all java programs. It contains Object class, all wrapper classes, Math class, String class, StringBuffer class etc.

➢ **java.io**

It contains all classes related to Input Output. Using these classes our java program can interact with IO Devices. Some of the classes /interfaces are, InputStream, Reader, Writer, PrintWriter, BufferedReader etc.

➢ **java.util**

it is also called collection framework. It has various classes which can be used to improve performance of java program. Some of the classes /interfaces are Scanner, ArrayList, Vector, LinkedList etc.

➢ **java.applet**

This package has various classes which helps us to implement applet program in java. Some of the classes /interfaces are Applet, Graphics, etc.

> **java.net**

It has various network programming related classes. Using these classes we can implement java programs on remote machine which can interact with each other. Some of the classes /interfaces are Socket, DatagramSocket, InetAddress, URL etc.
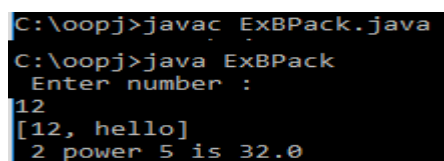
> **java.sql**

This package supports a java program to interact with DBMS software. Our java program can send data to database and can access database using these classes. Some of the classes /interfaces are Connection, Driver, DriverManager, ResultSet etc.

Example,

```java
import java.util.Vector;
import java.util.Scanner;
import static java.lang.Math.*;

public class ExBPack
{
        public static void main ( String args[] )
      {
        Vector v = new Vector ( 20 );
        Scanner sc = new Scanner  ( System.in );
        System. out. println (" Enter number : ");
        int x = sc.nextInt();
        v. add ( x );
        v. add ( "hello" );
       System. out. println  ( v );
       System. out. println  ( " 2 power 5 is " + pow(2,5) );
      }
}
```



```
C:\oopj>javac ExBPack.java

C:\oopj>java ExBPack
 Enter number :
12
[12, hello]
 2 power 5 is 32.0
```

Figure-57 Output of program

1.8 LET US SUM UP

Package : They are the container of java classes and interfaces related to same functionality. A package can be created using package keyword at the beginning of the java program.

Classpath: It is an environment variable which specifies the location of class files which are used by java program. We can view value of this variable using echo command and modify its value using set command.

Import: It is used to import the package and classes/ interfaces of the packages which we are using in our program. This classes /interfaces are readily available in package folder or jar file.

Static import: Using static import, we can import the classes whose static member or method can be used directly (without class name) in our program.

Access protection: By creating default, private, public and protected member variables and methods, we can restrict their access outside the class definition and or package.

Built-in package: they are the readily available class /interface libraries which can directly be used in java program by importing them when required.

1.9 CHECK YOUR PROGRESS

➤ True-False with reason.

1. Packages are collection of methods.
2. We can create a sub package in package.
3. Package creates a directory for storing class files.
4. Import keyword is used to create a package.
5. We can declare package anywhere in our program.
6. Jar file is a compressed source code java files.
7. Import static is used to call static method of class without class name.
8. We can create class with same name in two different packages.
9. Built in packages are the readily available class which one can directly use by importing them.

10. Package can provide protection to some members of the class.

➢ Match **A** and **B**.

A	**B**
1)package	a)environment variable
2)import	b)key word for creating package
3)import static	c)key word for importing class of package
4)class path	d)compressed collection of class files
5)jar file	e)use to access static member without class name

➢ Answer the following.

1. Which keyword is used to declare package?
2. Which option with javac can be used to automatically create a package folder?
3. Write a command used to create a jar file.
4. What is class path ? how can we print value of class path?
5. What is the application of static import?

➢ MCQ.

1. Which of these keywords is used to define packages in Java?

a) pkg   c) package

b) Pkg   d) Package

2. Which of this access specifies can be used for a class so that its members can be accessed by a different class in the same package?

a) Public   c) No Modifier

b) Protected   d) All of the mentioned

3. Which of these access specifiers can be used for a class so that its members can be accessed by a different class in the different package?

a) Public   c) Private

b) Protected   d) No Modifie

4. Which of the following is the correct way of importing an entire package 'pkg'?

a) import pkg.   c) import pkg.*

b) Import pkg.   d) Import pkg.*

5. What is the output of this program?

```
package pkg;
class display
{
    int x;
    void show()
    {
        if (x > 1)
            System.out.print(x + " ");
    }
}
class packages
{
    public static void main(String args[])
    {
        display[] arr = new display[3];
        for(int i = 0;i < 3; i++)
            arr[i] = new display();
        arr[0].x = 0;
        arr[1].x = 1;
        arr[2].x = 2;
        for (int i = 0; i < 3; ++i)
            arr[i].show();
    }
}
```
Note : packages.class file is in directory pkg;

a) 0   c) 2

b) 1   d) 0 1 2

6. Which of the following is an incorrect statement about packages?

a) Package defines a namespace in which classes are stored

b) A package can contain other package within it

c) Java uses file system directories to store packages

d) A package can be renamed without renaming the directory in which the classes

7. A package is container of _____

a) Methods   b) Objects

c) Classes   d) Variables

8. If a variable is declared as private , then it can be used in _____

a) Any class of any package   c) Only in the same class

b) Any class of same package   d) Only subclass in that package

9. which package is imported implicitly?

a) java.applet   c) java.lang

b) java.util   d) java.io

10. Math class is in ………package.

a) java.io   c) java.util

b) java.lang   d) java.applet

11. Syntax of pow()method is…….

a) double pow(double a, double b)   c) int pow(int a, int b)

b) double pow(int a, int b)   d) double pow(int a)

12. Write a output of following:

    class MathEx

    {
        public static void main(String args[])
        {
                double a = 123.34;
                double b = 234.56;
                System. out. println (" a =" + Math.ceil(a));
                System. out. println (" b =" + Math.floor(b));
        }
    }

a) a=123 b=234   c) a=124 b=234

b) a=124 b=235   d) a=123 b=235

13. Write a output of following

    Class MathTest

    {
       public static void main(string arg[]
        {
                double a = 123.456;
                System. out. println (Math.rint(a));
        }
    }

a) 123.46   b) 123

c) 124   d) 123.0

14. The data type wrapper classes are in……….package

a) java.lang   c) java.util

b) java.io   d) java.applet

15. syntax of getTime() method is …………..

a) date getTime()   c) long getTime()

b) void getTime(date d)   d) long getTime(date d)

16. Find errors if any otherwise write output:

    class Ex

    {

       Public static void main(String args[])

      {

         Date d = new date();

         System. out. println (" date is :" + d);

      }

    }

a) it will print the whole current date and time

b) it will print the current date

c) error:date class and constructor not found

d) error:can not print object d

17. The random class is in ……..package

a) java.io   c) java.lang

b) java.util   d) java.applet

18. The…….class creates a dynamic array

a) vector   c) random

b) calendar   d) object

19. To add element in vector ,…….method is used

a) add item()   c) addelement()

b) insertitem()   d) insertelement

20. To know the size of a vector ……..method is used

a) length()   c) capacity()

b) size()   d) getsize()

21. Which of the following is/are true about packages in Java?

a) Every class is part of some package.

b) All classes in a file are part of the same package.

c) If no package is specified, the classes in the file go into a special unnamed package

d) If no package is specified, a new package is created with folder name of class and the class is put in this package.

a) Only a, b and c   c) Only d

b) Only a, b and d   d) Only a and c

22. Which of the following is/are advantages of packages?

(a) Packages avoid name clashes

(b) Classes, even though they are visible outside their package, can have fields visible to packages only

(c) We can have hidden classes that are used by the packages, but not visible outside.

(d) All of the above

23. Predict the output of following Java program

import static java.lang.System.*;

```
class StaticImportDemo
{
  public static void main(String args[])
  {
     out.println(" hello ");
  }
}
```
(a) Compiler Error   (c) hello

(b) Runtime Error   (d) None of the above

24. Predict the output of following program

/* Hello.java */

package a;

```java
public class Hello {
    public void doIt()
    {
        printMessage();
    }
    void printMessage()
    {
        System. out. println (" Hello ");
    }
}
/* World.java */
package b;
import a.Hello;
public class World {
    private static class GFG extends Hello {
        void printMessage()
        {
            System. out. println ("World");
        }
    }
    public static void main(String[] args)
    {
        GFG gfg = new GFG();
        gfg.doIt();
    }
}
```

(a) Compiler Error   (c) Hello

(b) Runtime Error   (d) None of the above

25.
```java
/ / Hello.java
package a;
public class Hello {
```

```java
      void printMessage()
      {
          System. out. println ("Hello");
      }
}


// World.java
package b;
import a.Hello;
public class World extends Hello {
    void printMessage()
    {
        System. out. println ("World");
    }
    public static void main(String[] args)
    {
        Hello gfg = new World();
        gfg.printMessage();
    }
}
```

(a) Compiler Error   (c) Hello

(b) Runtime Error   (d) World

26. In java string is _____

a) Array of characters   c) a single character

b) An object of String class   d) Both A and B

27. Which method is used to find the position of a particular substring from a string?

a) substring()   c) charAt()

b) getChars()   d) indexOf()

28. The syntax of charAt() method is _____

a) int charAt(int no)   c) char charAt(int no)

b) int charAt(char ch)   d) char charAt(char no)

29. Which of the following is a string comparision method ?

a). startsWith()   c). substring()

b).endsWith()   d). reqionMatches()

30. The ___ class creates a fixed length string.

a) String   c) Character

b) StringBuffer   d) All of above

31. Which method is used to specify the minimum capacity of the StingBuffer object?

a) capcity()   c) ensurureCapacity()

b) setCapacity()   d) setLength()

32. The syntax of delete() method is _____

a) StringBuffer delete(char ch)

b) StringBuffer delete(char ch,int startIndex)

c) StringBuffer delete(int startIndex,int endIndex)

d) StringBuffer delete(char ch1,char ch2)

33. Which method is used to convert a string in uppercase?

a) uppercase()   c) changeCase()

b) toUpperCase()   d) capitalize()

34. Write output of following:

```
    class str
    {
       public static void main(String args[])
        {
            StringBuffer s = new StringBuffer( "ABCDE" );
            s.setChatAt ( 3, 'X' );
            System. out. println  ( s );
        }
    }
```
a) ABCXDE   c) ABXDE

b) ABCXE   d) ABXCDE

35. syntax of replace() of string class is _____

a) String replace(char ch1, char ch2)

b) void replace(char ch1, char ch2)

c) String replace(char ch1, int i)

d) void replace(char ch1, int i)

36.Wrapper classes are found in _____ package

a) java.lang

c) java.io

b) java.util

d) java.net

1.10 CHECK YOUR PROGRESS: POSSIBLE ANSWERS

➢ True-False with reason

1.   False. Package is a collection of classes

2.   True.

3.   True

4.   False. Import is used to include the class/classes of a package which we want to use in our program.

5.   False. Package must be declared at the beginning of the program

6.   False. Jar file is a compressed class files.

7.   True.

8.   True.

9.   True

10.   True

➢ Match **A** and **B**.

A	**B**
1)package	a)environment variable
2)import	b)key word for creating package
3)import static	c)key word for importing class of package
4)class path	d)compressed collection of class files
5)jar file	e)use to access static member without class name

Answer :

1) - b ,   2) - c,   3) – e,   4) – a,   5) – d

➢ Answer the following.

1. "import" keyword is used to declare package.

2. "-d" option is used with javac to automatically create a package folder.

3. Command used to create a jar file:

   jar cvf a1.jar pkg1

   Here a1.jar is name of the file created

   pkg1 is package/folder name which contains the class files

4. Classpath stores the location of class files or path of jar file which has classes. To print class path *"echo %CLASSPATH%"* command is used on command prompt.

5. Static import is used to access static member without class name.

➢ MCQ

1) c	13) b	25) d
2)c	14) a	26) b
3) a	15) c	27) d
4) c	16) a	28) c
5) c	17) b	29) d
6) d	18) a	30) a
7) c	19) c	31) a
8) c	20) b	32)c
9) b	21) a	33) b
10) b	22)d	34) b
11) a	23) c	35) a
12)c	24) a	36) a

1.11 FURTHER READING

1) "Java 2: The Complete Reference" by Herbert Schildt, McGraw Hill Publications.

2) "Effective Java" by Joshua Bloch, Pearson Education.

3) Java package tutorial with example - Java tutorial and examples
   http://java.candidjava.com/tutorial/Java-package-tutorial-with-example.htm

4) Java 101: Packages organize classes and interfaces | JavaWorld
   https://www.javaworld.com/.../core-java-packages-organize-classes-and-interfaces.htm

5) How to Create PACKAGE in Java: Learn with Example Program.

6) https://www.guru99.com/java-packages.html.

1.12 ASSIGNMENTS

1) Create a package name Vehicle. In this package create classes named bicycle, motor cycle, car, bus and truck with appropriate attributes and methods in them. Compile the java files and create the classes in package Vehicle. Now prepare jar file containing this package. Put this jar file in class path. Create a java program outside this package which is using this package by importing it. Also create object of each class and call methods in main method. Use appropriate access modifier while creating classes.

2) Create a package name forest. Create a sub package named animals in it. In animal sub package create classes for tiger, lion, bear and fox with appropriate attributes and methods in them. Modify the class path so that one can use the package forest (without creating jar file). Now create java program with main method which import this package and demonstrate use of this package in it. Use appropriate access modifier while creating classes.