Thursday, April 21, 2022 9:37 AM

1. One Source File can Have Only one public class and the name of the file should be same as name of the public class

2. If we want multiple classes in one source file then we can have
   a. 0 public classes + n non public classess
   b. 1 public class + n non public classes

____

```
E:\>set classpath=c:\temp;c:\temp1

E:\>javac -d c:\temp1  E:\batches\IETCoreJava\AnimalPlanet.java

E:\>java AnimalPlanet
Error: Could not find or load main class AnimalPlanet
Caused by: java.lang.ClassNotFoundException: AnimalPlanet

E:\>java animals.user.AnimalPlanet
Cheetah is a cat family animal

E:\>set classpath=c:\temp;c:\temp1;"c:\program files"
```

____

Primitive Types , scopes , access specifiers , package keyword

____

What is the package of
 System
  String

 java.lang package is checked by default , so import is not needed !!!
____

Open the JAVADOCS for java.lang package

____
For every primitive data type we have a corresponding wrapper class in **java.lang** package

Wrapper Classes ---

Primitive type	Wrapper class
byte	Byte
short	Short
int	Integer
float	Float
double	Double

char	Character
boolean	Boolean

What is the meaning of Wrapper !!!! Wrapper is a object that holds a primitive type.

Ex1

write a class study.wrappers.Example1

Main

What is the need for the Wrapper ?

1 ---- Many library classes don't work with primitives , So we can pass wrappers over there

2 ---- wrapper classes have many utility/useful methods for that data type

HW 1----

Write a class study.wrappers.PasswordSettingUtility

Main

Accept a password from the user

Check if the entered password is a strong password ----

A strong password should have

a. At least one Uppercase
b. At least one number
c. The first character should be a letter

If all the 3 conditions match then print - congrats your password is strong

Else ask the user to enter again - check again -repeat

Hint - //use utility method isUpperCase ,isDigit , isLetter of Character wrapper class

___

API classes ---- Wrapper classes , java.lang.String , java.lang.StringBuffer, java.lang.StringBuilder

String class =

How do we create object of String class ?

String s = new String("hello"); //creating String object using new

String s1 = "good day"; // creating String object using literals

___

When String is created using literals --- it is added in Constant Pool

When String is created using new --- it is not added in Constant Pool

___

What is Constant Pool --------

Constant pools for primitives and Strings

Constant Pool = collection of strings created using literals is maintained one per JVM

String s1 = "hello" ; //new object created #1
String s2 = "hi"; //new object created #2
String s3 = "bye";//new object created #3
String s4 = "hello"; //**no new object is created** , it points to #1
String s5 = "hello" ; //no new object is created , it points to #1

String constant pool

Pool	
#1	hello
#2	hi
#3	bye

Advantage --- the space used is saved as two similar **strings share the same object**

If Strings are created using literals then there are chances that the string will be SHARED by many references in same thread or different thread .
If they are shared they are PRONE to a problem called as RACE CONDITION .
To avoid this problem all Strings are made IMMUTABLE !!!!

Immutable = once created , it cannot be changed !!!

____

toUpperCase()
toLowerCase()
concat()
substring()

____

Strings are immutable ---

Advantage	space is saved due to constant pool sharing
	due to immutable shared strings are protected from race condition
Disadvantage	Every time a string change operation is done a new is created !!!

Sometimes we may need to change a string many times . At that time String class is not a good idea --too many objects will be created !!!
We need MUTABLE strings !!! Two classes in java.lang package StringBuffer , StringBuilder

String	StringBuffer	StringBuilder
java.lang	Java.lang	java.lang
Constant pool is used	NA	NA

Shared strings are protected by IMMUTABILITY	Shared strings are protected by THREAD SAFETY synchronization	Shared strings are not protected and may face race condition
When we try to modify a string , the API creates a new string and does not modify the calling string	When we try to modify a string, API will modify the calling string	When we try to modify a string, API will modify the calling string

---

Wrapper constant pools are also present in Java -------------

Example of Integer constant pools in range -128 to 127

If (obj1 == obj2) // compares addresses of the objects

If(x == y) //where x and y are primitives --- compares values of the objects

---

Parameter Passing in Java -------- parameter passing by VALUE /CALL by Value
For references ---- addresses are copied between caller and callee
For primitives ---- values are copied between caller and callee

HW -------

```
Write a class study.parampass.ExampleHW
        Main
                Create  x and y
                Print before
                Pass them to swap
                Print after

        Public static void swap(     x ,y )
        {
                Swap / interchange  their values
         }
```

Ex1- write a class study.strings.Example1

Main