DAY-31

------

1. number of words in a given string

--------------------------------------

EXAMPLE:

--------

import java.util.\*;

class Demo

{

public static void main(String[] args)

{

String str = " join studyonline to learn programming languages";

StringTokenizer s = new StringTokenizer(str);

int count = s.countTokens();

System.out.println("The number of words in the given string is: "+count);

}

}

OUTPUT:

-------

The number of words in the given string is 6

2. remove the white spaces in the given string

-----------------------------------------------

EXAMPLE:

--------

class Demo

{

public static void main(String[] args)

{

String str = " join studyonline to learn programming languages ";

System.out.println(str);

str = str.replaceAll("\\s","");

System.out.println(str);

}

}

OUTPUT:

------

join studyonline to learn programming languages

joinstudyonlinetolearnprogramminglanguages

3. Swapping Strings without using Third variable

-------------------------------------------------

EXAMPLE:

--------

class Demo

{

public static void main(String[] args)

{

String b = "hi had ur dinner? ";

String g = "hmmmm";

System.out.println("strings before swapping:"+b);

System.out.println("strings before swapping:"+g);

b = b+g;

g = b.substring(0,b.length()-g.length());

b=b.substring(g.length());

System.out.println("strings after swapping:"+b);

System.out.println("strings after swapping:"+g);

}

}

OUTPUT:

-------

strings before swapping:hi had ur dinner?

strings before swapping:hmmmm

strings after swapping:hmmmm

strings after swapping:hi had ur dinner?

4.CHECK IF THE GIVEN STRING IS ANAGRAM OR NOT.

----------------------------------------------

example for anagram

--------------------

s1 = "car";

s2 = "rac";

s1 = "from";

s2 = "form";

s1 = "silent";

s2 = "listen";

import java.util.\*;

class Demo

{

public static void main(String[] args)

{

String s1 = "listen";

String s2 = "silent";

if (s1.length()!=s2.length())

{

System.out.println("Strings are not anagram");

System.exit(0);

}

char a[]=s1.toCharArray();

char b[]=s2.toCharArray();

Arrays.sort(a);

Arrays.sort(b);

if(Arrays.equals(a,b))

{

System.out.println("Strings are anagram");

}

else

{

System.out.println("Strings are not anagram");

}

}

}

OUTPUT:

-------

Strings are anagram

-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

OBJECT ORIENTATION FEATURES:

----------------------------

ENCAPSULATION

-------------

It refers to wrapping the data(variable) and the code acting on the data(methods) as a single unit.

Encapsulation can be achived using:

1. By making the instance variable as private.

2. By creating Public setter and getter methods in order to access and view the data.

EXAMPLE:

--------

class Student

{

private String name;

private int age;

private int roll\_no;

public void SetName(String sname)

{

name=sname;

}

public void SetAge(int sage)

{

age=sage;

}

public void SetRoll\_no(int sroll\_no)

{

roll\_no=sroll\_no;

}

public String getName()

{

return name;

}

public int getAge()

{

return age;

}

public int getRoll\_no()

{

return roll\_no;

}

}

class StudentApp

{

public static void main(String[] args)

{

Student s1=new Student();

s1.SetName("raju");

System.out.println(s1.getName());

s1.SetAge(21);

System.out.println(s1.getAge());

s1.SetRoll\_no(84);

System.out.println(s1.getRoll\_no());

}

}

OUTPUT:

-------

raju

21

84

EXAMPLE:2

---------

class Student

{

private String name;

private int age;

private int roll\_no;

public void SetData(String sname, int sage, int sroll\_no)

{

name=sname;

age=sage;

roll\_no=sroll\_no;

}

public String getName()

{

return name;

}

public int getAge()

{

return age;

}

public int getRoll\_no()

{

return roll\_no;

}

}

class StudentApp

{

public static void main(String[] args)

{

Student s1=new Student();

s1.SetData("raju",21,84);

System.out.println(s1.getName());

System.out.println(s1.getAge());

System.out.println(s1.getRoll\_no());

}

}

OUTPUT:

-------

raju

21

84

NOTES:

------

1. DATA HIDING:

Data Hiding is hiding the data so that outside person cannot access the data directly. This can be achicved by using the data members (variable)

as private.

2. Tightly encapsulated class

In a class if all the data members is made as private then such class is called as tightly encapsulated class.

ADAVANTAGES OF ENCAPSULATION:

------------------------------

1. SECURITY TO THE DATA.

2. ENHANCEMENT IS EASY.

------------------------------------------------------------------------------------------------------------------------------------------------------

CONSTRUCTORS

------------------------------

There are 3 ways to intilize the object.

----------------------------------------

1. using the refrence variable

2. using the methods

3. using the constructors

using the reference variable:

-----------------------------

EXAMPLE:

--------

class Student

{

String name;

int roll\_no;

}

class Demo2

{

public static void main(String[] args)

{

Student s1 = new Student();

s1.name ="raju";

s1.roll\_no = 84;

System.out.println(s1.name);

System.out.println(s1.roll\_no);

}

}

OUTPUT:

--------

raju

84

using the methods

---------------------

EXAMPLE:

--------

class Student

{

String name;

int roll\_no;

public void SetData(String sname, int sroll\_no)

{

name=sname;

roll\_no=sroll\_no;

}

public String getName()

{

return name;

}

public int getRoll\_no()

{

return roll\_no;

}

}

class Demo2

{

public static void main(String[] args)

{

Student s1 = new Student();

s1.SetData("somu",89);

System.out.println(s1.getName());

System.out.println(s1.getRoll\_no());

}

}

OUTPUT:

-------

somu

89

using the constructors

-----------------------

EXAMPLE:

--------

class Student

{

String name;

int roll\_no;

Student(String sname,int sroll\_no)

{

name=sname;

roll\_no=sroll\_no;

System.out.println(name);

System.out.println(roll\_no);

}

}

class Demo2

{

public static void main(String[] args)

{

Student s1 = new Student("bheem",28);

Student s2 = new Student("somu",89);

}

}

OUTPUT:

-------

bheem

28

somu

89

Constructor is a special block or method where name of the constructor is same as the class name.

It is a Specialized setter to initilize the object.

Constructor doesnot have any return type.The access modifier allowed for the Constructor method is public,private,protected,default.

Constructor are always called during the object creation.