***Dt : 8/9/2022***

***(c)Using 'is equal to'(==) operator***

***=>"is equal to" operator will compare the references of an object,***

***which means it will not compare the content of an objects.***

***Ex : DemoString7.java***

***package maccess;***

***public class DemoString7 {***

***public static void main(String[] args) {***

***String s1 = "nit";***

***String s2 = "nit";***

***System.out.println("====String literal process=====");***

***if(s1==s2) {***

***System.out.println("Strings are equal...");***

***}else {***

***System.out.println("Strings are Not-equal...");***

***}***

***String s3 = new String("hyd");***

***String s4 = new String("hyd");***

***System.out.println("====new operator process=====");***

***if(s3==s4) {***

***System.out.println("Strings are equal...");***

***}else {***

***System.out.println("Strings are Not-equal...");***

***}***

***}***

***}***

***o/p:***

***====String literal process=====***

***Strings are equal...***

***====new operator process=====***

***Strings are Not-equal...***

***=======================================================***

***faq:***

***define 'String Constant pool'?***

***=>The separate partition of Heap Area where String objects are***

***created is known as 'String Constant pool'.***

***Note:***

***=>'String Constant pool' will restrict duplicate string object***

***creations.***

***=========================================================***

***faq:***

***wt is the difference in creating objects using 'String literal***

***process' and 'new operator process'?***

***(i)In String literal process the execution control will check the***

***String Contant pool,is any object having same data***

***=>If object not available then new Object is created***

***=>If object is available,then reference of existing object is***

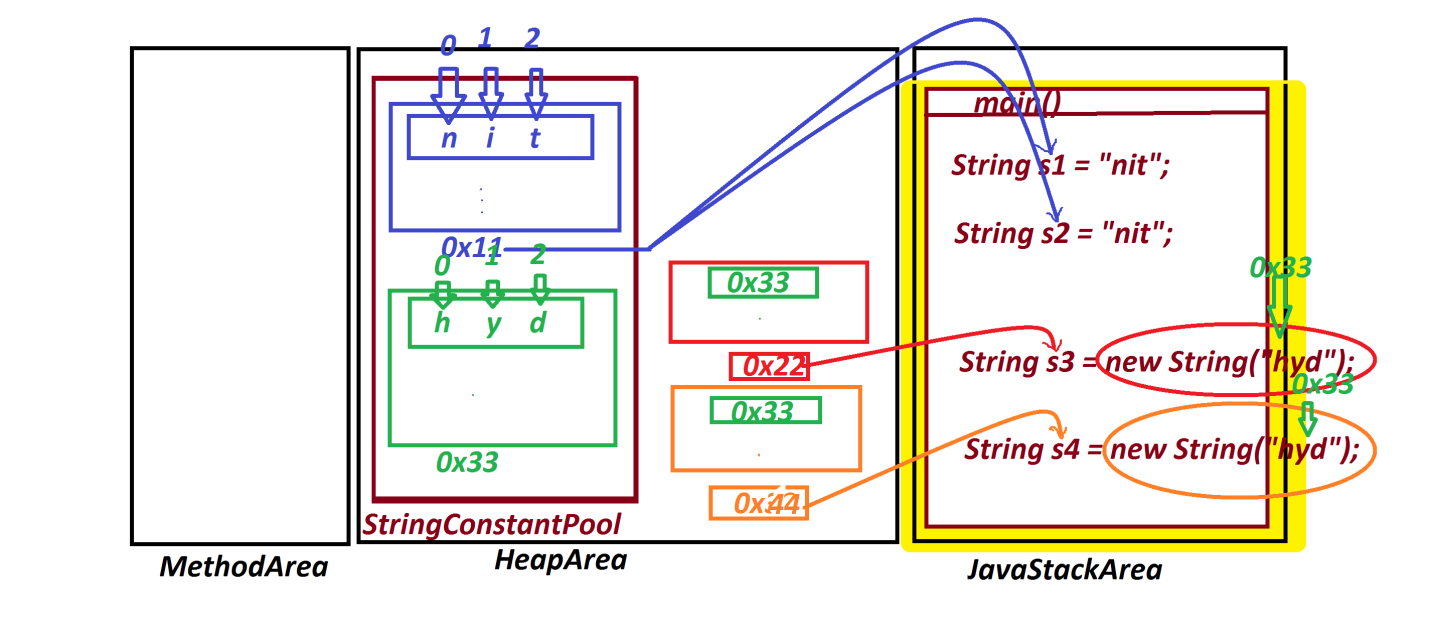
***used without creating new object.***

***(ii)In 'new operator process' the object is created directly in***

***HeapArea,but the object will hold reference of object created***

***in String Constant Pool.***

***Diagram:***

******

***=========================================================***

***Note:***

***=>'is equal to'(==) operator must be used on primitive datatype***

***variables and must not be used on Non-Primitive datatype variables.***

***==========================================================***

***\*imp***

***(2)java.lang.StringBuffer class:***

***=>The objects which are created using 'StringBuffer' class are***

***known as "Mutable Objects".(The objects once created can be modified***

***are known as Mutable objects)***

***=>The following are four constructors from 'StringBuffer':***

***public java.lang.StringBuffer();***

***public java.lang.StringBuffer(int);***

***public java.lang.StringBuffer(java.lang.String);***

***public java.lang.StringBuffer(java.lang.CharSequence);***

***case-1 : Object creation using 'java.lang.StringBuffer()'***

***syntax:***

***StringBuffer sb = new StringBuffer();***

***=>In this syntax the StringBuffer object is created with the***

***default capacity 16.(characters)***

***=>when the length crossed the capacity 16,then the capacity***

***increases automatically by doubling the capacity and adding 2.***

***=>when the length crossed the capacity 34,the capacity increases***

***character by character.***

***=>we use append() method to add the data to the StringBuffer***

***object.***

***Ex : DemoBuffer1.java***

***package maccess;***

***import java.util.\*;***

***public class DemoBuffer1 {***

***public static void main(String[] args) {***

***Scanner s = new Scanner(System.in);***

***StringBuffer sb = new StringBuffer();***

***System.out.println("default capacity:"+sb.capacity());***

***System.out.println("length:"+sb.length());***

***System.out.println("Enter the data to be added to buffer:");***

***sb.append(s.nextLine());//adding data to buffer***

***System.out.println("===display from sb====");***

***System.out.println(sb.toString());***

***System.out.println("capacity:"+sb.capacity());***

***System.out.println("length:"+sb.length());***

***s.close();***

***}***

***}***

***o/p:***

***default capacity:16***

***length:0***

***Enter the data to be added to buffer:***

***java program nit java program nitKK***

***===display from sb====***

***java program nit java program nitKK***

***capacity:35***

***length:35***

***=================================================***