***Dt : 27/8/2022***

***Note:***

***(i)we can create any number of objects to a class and the***

***multiple objects are independent by their memory location***

***on HeapArea.***

***(ii)The class loads onto MethodArea only once to generate***

***multiple objects.***

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

***faq:***

***wt is the diff b/w***

***(i)Methods***

***(ii)Blocks***

***=>Methods are executed on method\_call,but blocks are executed***

***automatically without calling.***

***=>Blocks will have highest priority in execution than methods.***

***=>Static blocks will have highest priority in execution than***

***Static methods***

***=>Instance blocks will have highest priority in execution***

***than Instance methods***

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

***Assignment:(Solution)***

***wap to read and display Customer details using reference***

***variables as parameters?***

***Ex : DemoMethods11.java***

***import java.util.Scanner;***

***class Customer //SubClass***

***{***

***String custId,custName,custMailId,custCity;***

***long custPhNo;***

***}***

***class ReadCustDetails//SubClass***

***{***

***Customer read(Scanner s)***

***{***

***Customer c = new Customer();***

***System.out.println("Enter the CustId:");***

***c.custId = s.nextLine();***

***System.out.println("Enter the CustName:");***

***c.custName = s.nextLine();***

***System.out.println("Enter the CustMailId:");***

***c.custMailId = s.nextLine();***

***System.out.println("Enter the CustPhNo:");***

***c.custPhNo = Long.parseLong(s.nextLine());***

***System.out.println("Enter the CustCity:");***

***c.custCity = s.nextLine();***

***return c;***

***}***

***}***

***class DisplayCustDetails //SubClass***

***{***

***void display(Customer c)***

***{***

***System.out.println("====CustomerDetails====");***

***System.out.println("CustId:"+c.custId);***

***System.out.println("CustName:"+c.custName);***

***System.out.println("CustMailId:"+c.custMailId);***

***System.out.println("CustPhoneNo:"+c.custPhNo);***

***System.out.println("CustCity:"+c.custCity);***

***}***

***}***

***class DemoMethods11 //MainClass***

***{***

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

***{***

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

***ReadCustDetails rcd = new ReadCustDetails();***

***Customer c = rcd.read(s);***

***DisplayCustDetails dcd = new DisplayCustDetails();***

***dcd.display(c);***

***}***

***}***

***o/p:***

***Enter the CustId:***

***C11223456***

***Enter the CustName:***

***Raj***

***Enter the CustMailId:***

***raj@gmail.com***

***Enter the CustPhNo:***

***9898981234***

***Enter the CustCity:***

***Hyd***

***====CustomerDetails====***

***CustId:C11223456***

***CustName:Raj***

***CustMailId:raj@gmail.com***

***CustPhoneNo:9898981234***

***CustCity:Hyd***

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

***\*imp***

***Constructors in Java:***

***=>Constructors are the methods which are having the same name***

***of the class and executed while object creation,because the***

***Constructor call is available in Object creation syntax.***

***Note:***

***=>while declaring constructors we mustnot use return\_type,***

***because the constructors are Class\_return\_type.***

***structure of Constructor:***

***Class\_name(para\_list)***

***{***

***//Constructor\_body***

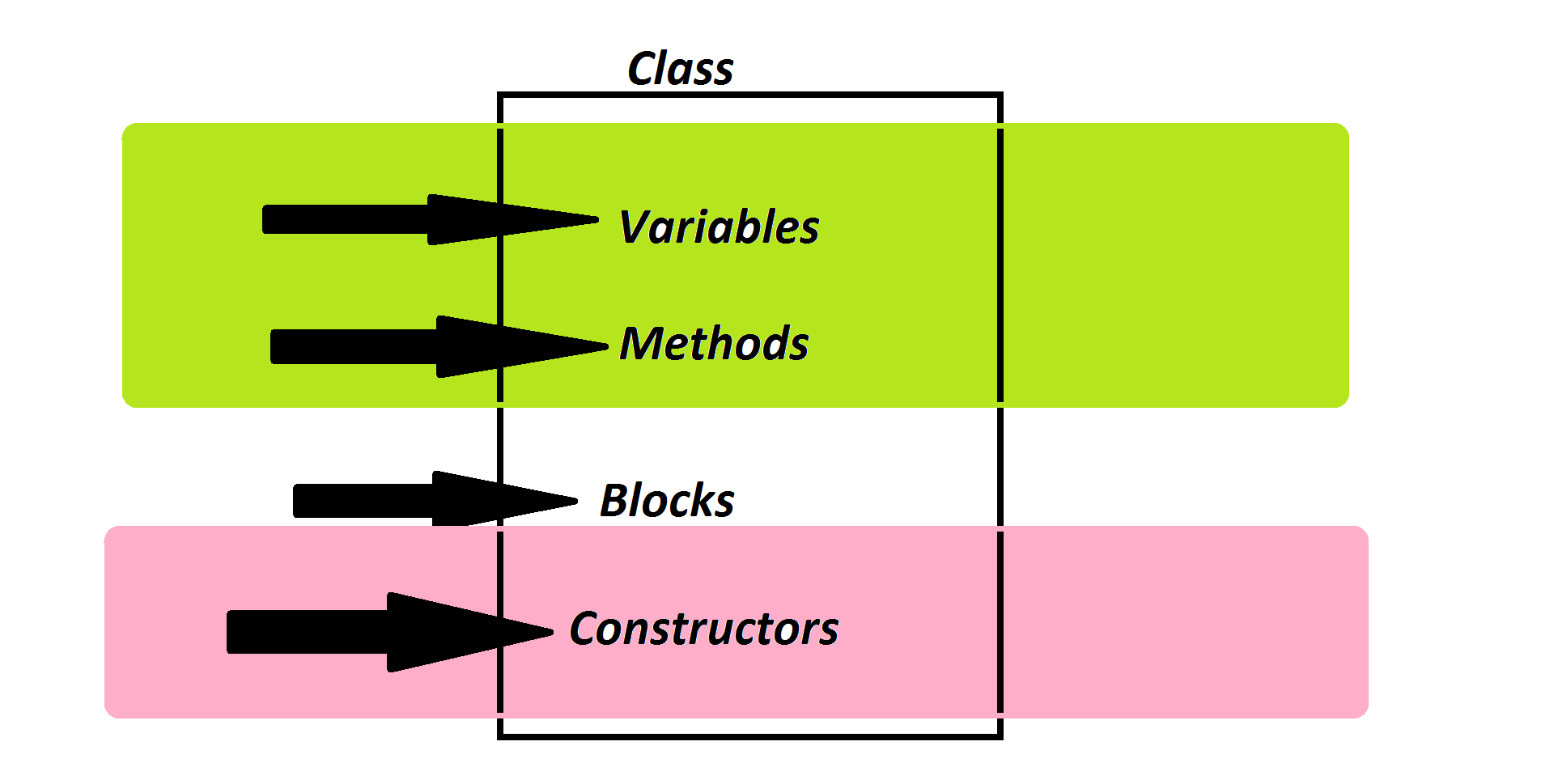
***}***

***=>Constructors in Java are categorized into two types:***

***1.Constructors without parameters***

***2.Constructors with parameters***

***--------------------------------------------------------------***

******