CONSTRUCTORS

CONSTRUCTOR AND ITS PURPOSE:

Constructor is a block of code that initializes the newly created object. A constructor resembles an instance method in java but its not a method as it doesn't have a return type.In short constructor and method are different. It is often referred to as a special type of method in java.

CONSTRUCTORS:

A constructor in java is a special method that is used to initialize objects.The constructors are called when an object of a class is created.It can be used to set initial values for object attributes.

SYNTAX:

```
class Name
{
//Statements involving the variables of the class
}
 class Main
{
    public static void main(String args[])
    {
        class_name object_name=new class_name();
        //this is a default constructor
      }
….
}
```

EXAMPLE:

```
class Hello
{
   int a;
   Hello()
   {
     a=10;
   }
   public class  Main
   {
       public static void main(String args[])
```

```
        {
             Hello obj=new Hello();
            System.out.println(“a=”+obj.a);
        }
    }
```

OUTPUT:

```
        a=10
```

RULES:

- Constructor should not have a return type.
- Constructor of a class must have the same name as the class name.
- Constructor in java can not be abstract,final,static and synchronized.
- Access modifiers can be used in constructor declaration to control its access i.e which other class can call the constructor.

DIFFERENCE BETWEEN METHOD AND CONSTRUCTOR:

CONSTRUCTOR	**METHOD**
A Constructor is a block of code that initializes a newly created object.	A Method is a collection of statements which returns a value upon its execution.
A Constructor can be used to initialize an object.	A Method consists of Java code to be executed.
A Constructor is invoked implicitly by the system.	A Method is invoked by the programmer.
A Constructor doesn’t have a return type.	A Method must have a return type.
A class can have many Constructors but must not have the same parameters.	A class can have many methods but must not have the same parameters.
A Constructor cannot be inherited by subclasses.	A Method can be inherited by subclasses

TYPES OF CONSTRUCTORS:

- Default constructor
- Multiple constructors
- Parameterized constructors

DEFAULT CONSTRUCTOR:

The term default constructor can refer to a constructor that is automatically generated by the compiler in the absence of any programmer-defined constructors

Note that a constructor with formal parameters can still be called without arguments if default arguments were provided in the constructor's definition.

- The default constructor has the same access modifier as the class, unless the class lacks an access modifier, in which case the default constructor has package access .
- The default constructor has no formal parameters, except in a non-private inner member class, where the default constructor implicitly declares one formal parameter representing the immediately enclosing instance of the class.

EXAMPLE:

```
class Hello
{
  int area;
  Hello()
  {
      int a=5;
      Hello obj=new Hello();
      System.out. println("constructor called ");
      area=a*a;
   }

}
 class  Main
{
    public static void main(String args[])
    {
       Hello obj=new Hello();//default constructor is created
       System.out.println("area="+obj.area);
     }
}
```

OUTPUT:

```
constructor called
area=25 //constructor is passed
```

For a class **HELLO** we initialize the object **(obj)** constructor is created as default in this example

PARAMETERIZED CONSTRUCTOR :

The default constructor does not pass any arguments to the constructor. Therefore, a parameterized constructor is used to initialize objects with different sets of values at the time of their creation. These different sets of values initialized to objects must be passed as arguments when the constructor is invoked.

EXAMPLE:

```
class Rect
{
   int a,b,area;
   Rect(int l,int br)
   {
      a=l;
      b=br;
      area=a*b;
      System.out.println("area= "+area);
    }
}
 class Main
 {
public static void main(String args[])
     {
        Rect obj=new Rect(10,20);
        //constructor is created
     }
}
```

OUTPUT:

```
area=200
```

In the above example, we have parameterized constructor with two parameters **l** and br. While creating the object obj have passed two arguments**(10,20)** so that this constructor gets invoked after creation of object.

MULTIPLE CONSTRUCTORS:

A class can have multiple constructors, as long as their signature (the parameters they take) are not the same. You can define as many constructors as you need. When a Java class contains multiple constructors, we say that the constructor is overloaded (comes in multiple versions). This is what constructor overloading means, that a Java class contains multiple constructors

EXAMPLE:

Here is a complete example that demonstrates the multiple constructors used in java programs.

```
class Rect
{
   int a,b,area;
   Rect()
   {
         a=0;
         b=0;
         area=a*b;
         System.out.println("area= "+area);
    }
   Rect(int l,int br)
   {
      a=l;
      b=br;
      area=a*b;
      System.out.println("area= "+area);
    }
}
class main
{
     public static void main(String args[])
     {
         Rect obj1=new Rect();
         Rect obj=new Rect(10,20);//constructor is created
      }
 }
```

```
class Car
{
    String Name1,Name2;
    int Price;
    Car(String name)
    {

       System.out. println("car name :"+name);
    }
    Car(String name,int price)
    {

       System.out. println("car name:"+name);
       System.out. println("price:"+price);
     }
}
class Main
{
     public static void main(String args[])
     {
         Car honda=new Car("honda"); //constructor is created
         Car maruti=new Car("maruti",400000); //two parameter constructor is created
     }
}
```

OUTPUT:

```
Car name:honda//single constructor is passed
Car name:maruti
Price:400000//parameterized constructor is passed
```

In the above example a single parameterized constructor(honda)is passed and two parameterized constructor (maruti,40000)are passed so in this program more than one parameter is passed so it comes under multiple constructor.

Practice questions:

```
class Add
{
   int a,b,sum;
   Add(int a)
   {
      System.out.println(+a);
   }
    Add(int a,int b)
   {
        sum=a+b;
        System.out.println("sum="+sum);
    }
 }
class Main
{
public static void main(String args[])
 {
    Add a=new Add(10);
    Add a1=new Add(10,20);

 }
}
```

STACK :

```
import java.util.*;
class Stack
{
    int top=-1;
    static int size;
    int st[];
    Scanner sc = new Scanner(System.in);
    public void Stack()
    {
          System.out.println("enter the size of the stack");
          size = sc.nextInt();
          int st[]=new int[size];
    }
    public void push()
    {
          if(top >= size-1)
          {
               System.out.println("stack is full");
          }
          else
          {
               int b=0;
               System.out.println("enter the element  to be inserted");
               b=sc.nextInt();
               st[++top]=b;
          }
      }
    public void pop()
    {
        if(top<0)
        {
            System.out.println("stack is empty");
         }
        else
         {
             int x=st[top--];
             System.out.println("popped element"+st[x]);
          }
     }
     public void display()
     {
          int i;
          if(top==-1)
          {
               System.out.println("stack is empty");
          }
          else
          {
               for(i=0; i<top; i++)
               {
                     System.out.println(""+st[i]);
                }
           }
     }
 }
public class Main
{
     public static void main(String[] args)
     {
           int choice,size;
           Scanner obj=new Scanner(System.in);
           Stack ob = new Stack();
           ob.Stack();
           System.out.println("1.push 2.pop 3.display 4.exit");
            while(true)
            {
                    System.out.println("enter your choice");
                    choice=obj.nextInt();
                    switch(choice)
                    {
                        case 1:ob.push();
                              break;
                        case 2:ob.pop();
                              break;
                        case 3:ob.display();
                              break;
                        case 4:
                             break;
                       }
              }
        }
}
```

<u>Bank</u>:

Define a class Bank to represent the bank account of a customer with
the following specification.
Private Members:
-Name of type character array(string)
-Account_no of type long
-Type_of_account (S for Saving Account, C for current Account) of type char
-Balance of type float
Public Members:
(i) A constructor to initialize data members as follows
-Name NULL
-Account_no 100001
-Type_of_account ‘S’
-Balance 1000
(ii) A function NewAccount() to input the values of the data
members Name, Account_no, Type_of_account and Balance with following
two conditions.
 Minimum Balance for Current account is Rs.3000
 Minimum Balance for Saving account is Rs.1000
(iii) A function Deposit() to deposit money and update the Balance amount
(iv) A function Withdrawal() to withdraw money. Money can be withdrawn
if minimum balance is as >=1000 for Saving account and >=3000 for Current
account.
(v) A function Display() which displays the contents of all the data
members for an account.

```
import java.util.Scanner;
class Bank
```

```
{
     String name;
    long Account_no;
    char Type_of_account;
    float balance;
    public void Bank()
     {
       String name="NULL";
       long Account_no=100001;
       char Type_of_account='s';
       float balance=1000;
       System.out.println("name of the account user:"+name);
    System.out.println("type of your account:"+Type_of_account);
     System.out.println("account number:"+Account_no);
     System.out.println("your total amount in your account:"+balance);
     }
    void Newaccount()
    {
    System.out.println("enter your name");
     Scanner obj=new Scanner(System.in);
     name=obj.nextLine();
      System.out.println("enter the type of your account");
       System.out.println("for savings enter's',for current enter 'c'");
     Type_of_account=obj.next().charAt(0);
     if(Type_of_account=='c')
       {
          System.out.println("Minimum Balance for current account is Rs.3000");
       }
      else
       {
         System.out.println("Minimum Balance for savings account is Rs.3000");
       }
      System.out.println("enter your account number");
     Account_no=obj.nextLong();

      System.out.println("enter the balance amount");

       balance=obj.nextFloat();

    }
```

```
public void deposit()
{
    float money;
     System.out.println("enter the amount to be deposited");
      Scanner x=new Scanner(System.in);
      money=x.nextFloat();
      balance=balance+money;
      System.out.println("your total amount in your accont"+balance);
}
public void Withdrawal()
{
    float withdrawal;
    char Type_of_account;
    System.out.println("enter the amount to be withdrawal");
        Scanner y=new Scanner(System.in);
        withdrawal=y.nextFloat();
        if(balance<withdrawal)
        {
            System.out.println("insufficient amount");

        }
        System.out.println("enter the type of your account");
        System.out.println("for savings enter's',for current enter 'c'");
        Type_of_account=y.next().charAt(0);

    if(Type_of_account=='s')
    {
      if(balance>=1000)
      {
        balance=balance-withdrawal;
        System.out.println("your total amount in your accont:"+balance);
        }
    }
    else
    {
       if(balance>=3000)
       {
           balance=balance-withdrawal;
         System.out.println("your total amount in your accont:"+balance);
       }
    }
}
```

```
void display()
{

   System.out.println("name of the account user:"+name);
    System.out.println("type of your account:"+Type_of_account);
     System.out.println("account number:"+Account_no);
     System.out.println("your total amount in your account:"+balance);

}
}
class Main
{
public static void main (String[] args)
    {
        int choice;
        Bank z=new Bank();
       z.Bank();
        z.Newaccount();


      while(true)
      {
          System.out.println("enter your choice");
          System.out.println("1.deposit 2.withdrawal 3.display4.exit");
          Scanner q=new Scanner(System.in);
          choice=q.nextInt();
          switch(choice)
          {
              case 1:z.deposit();
                     break;
              case 2:z.Withdrawal();
                     break;
              case 3:z.display();
                    break;
              case 4:
                     break;


          }
      }
    }
}
```

<u>Reservation</u>:

Develop a reservation class which has reserve method. implement the subclasses reserve train and reserve bus that overrides the reserve method of reservation class. implement a java code that accesses the super class constructors and methods.

```
import java.util.Scanner;
class Reservation
{
    void reserve()
    {
        int n;
        System.out.println(" enter the place you want");
        System.out.println("1.chennai 2.salem 3.namakal");
        Scanner ob=new Scanner(System.in);
        n=ob.nextInt();
        if(n==1)
        {
            System.out.println("your amount is 1500");
        }
        else if(n==2)
        {
            System.out.println("your amount is 650");
        }
        else
        {
            System.out.println("your amount is 500");
        }
    }
    void reserve1()
    {
        int n;
        System.out.println(" enter the place");
        System.out.println("1.chennai 2.salem 3.namakal");
        Scanner ob1=new Scanner(System.in);
        n=ob1.nextInt();
        if(n==1)
        {
            System.out.println("your amount is 1200");
        }
        else if(n==2)
        {
            System.out.println("your amount is 500");
        }
        else
        {
            System.out.println("your amount is 350");
        }
    }
    Reservation(float Date,float Time)
    {
        float date1=Date;
        float time1=Time;
        System.out.println("date:"+date1);
        System.out.println("time:"+time1);
        System.out.println("your kpn travels have been booked");
    }
     Reservation(float Date,float Time,int Age)
    {
        float date1=Date;
         float time1=Time;
        int age1=Age;
        System.out.println("date"+date1 );
        System.out.println("age"+Age);
        System.out.println("time:"+time1);
        System.out.println("your cochin express have been booked");
    }
}
public class Main
{
            public static void main(String[] args)
            {
                int n,age;
                 float date,time;
                String day;
                Scanner obj=new Scanner(System.in);
                System.out.println("enter the mode of transport");
                System.out.println("1.bus 2.train");
                n=obj.nextInt();
                if(n==1)
                {
                        System.out.println("enter the date and time");
                        date=obj.nextFloat();
                        time=obj.nextFloat();
                        Reservation p=new Reservation(date,time);
```

```
                    p.reserve();
               }
               else
               {
                   System.out.println("enter the date ,age and time");
                        date=obj.nextFloat();
                         age=obj.nextInt();
                         time=obj.nextFloat();
                         Reservation q=new Reservation(date,time,age);
                         q.reserve1();
                }
          }
}
```