JAVA Lab
Lab experiment number 2

Name: Aamir Ansari **Roll no.** 01 **Batch** A

Aim:

Implement Java programs to illustrate the concept of classes, objects, constructor, method overloading and array of objects

Theory:

Method overloading and constructor overloading

1. Method overloading:

   Method Overloading is a feature that allows a class to have more than one method having the same name, if their argument lists are different. It is similar to constructor overloading in Java, that allows a class to have more than one constructor having different argument lists.
   There are three types of Method overloading:
   a. By number of parameters
   b. By data type of parameters
   c. By sequence of data type of parameters

2. Constructor overloading

   Like methods, constructors can also be overloaded. Constructor overloading is a concept of having more than one constructor with different parameters list, in such a way so that each constructor performs a different task.
   Its types are similar to that of Method overloading.

Constructor Chaining

Constructor chaining is the process of calling one constructor from another constructor with respect to the current object.

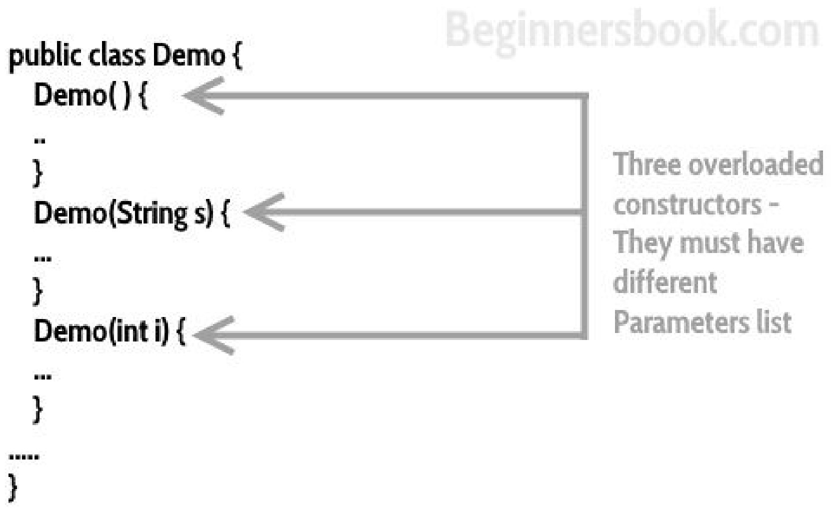
Constructor chaining can be done in two ways:

1. **Within same class:** It can be done using this() keyword for constructors in same class

2. **From base class:** by using super() keyword to call constructor from the base class.

The real purpose of Constructor Chaining is that you can pass parameters through a bunch of different constructors, but only have the initialization done in a single place.
This allows you to maintain your initializations from a single location, while providing multiple constructors to the user.
If we don't chain, and two different constructors require a specific parameter, you will have to initialize that parameter twice, and when the initialization changes, you'll have to change it in every constructor, instead of just the one.
As a rule, constructors with fewer arguments should call those with more

For example:



```
public class Demo {
  Demo( ) {
  ..
  }
  Demo(String s) {
  ...
  }
  Demo(int i) {
  ...
  }
.....
}
```

Beginnersbook.com
Three overloaded constructors - They must have different Parameters list

Usage of “this” and “super” keywords

This and super are reserved keywords in java i.e, we can’t use them as an identifier.

1. Usage of “this”

   ‘this’ is a reference variable that refers to the current object and can be used in various ways:

   A. Using this() to invoke current class constructor
   B. Using ‘this’ keyword to return the current class instance
   C. Using ‘this’ keyword as method parameter
   D. Using ‘this’ keyword to invoke current class method

2. Usage of “super”

   The super keyword in java is a reference variable that is used to refer to parent class objects.
   The keyword “super” came into the picture with the concept of Inheritance. It is majorly used in the following contexts:

   A. Use of super with variables
   B. Use of super with methods
   C. Use of super with constructors

Array of Objects

A Java array of objects, as defined by its name, stores an array of objects. Unlike a traditional array that stores values like string, integer, Boolean, etc an array of objects stores OBJECTS. The array elements store the location of the reference variables of the object.

Syntax:

```
Class obj[]= new Class[array_length]
```

To create and array of object, do the following:

1. The following statement creates an Array of Objects.

   ```
   Class_name [] objArray;
   ```

2. Alternatively, you can also declare an Array of Objects as shown below:

   ```
   Class_name objArray[];
   ```

   Both the above declarations imply that objArray is an array of objects.

3. So, if you have a class 'Employee' then you can create an array of Employee objects as given below:

   ```
   Employee[] empObjects;
           OR
   Employee empObjects[];
   ```

   The declarations of the array of objects above will need to be instantiated using 'new' before being used in the program.

4. You can declare and instantiate the array of objects as shown below:

   ```
   Employee[] empObjects = new Employee[2];
   ```

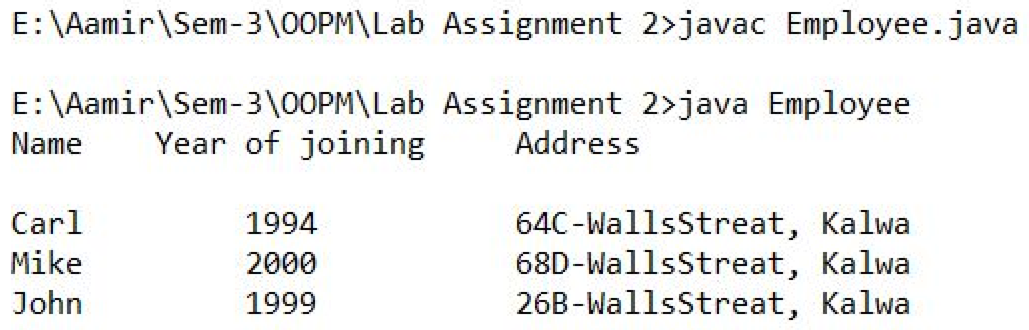
5. Once the array of objects is instantiated, you have to initialize it with values. As the array of objects is different from an array of primitive types, you cannot initialize the array in the way you do with primitive types.

Program:

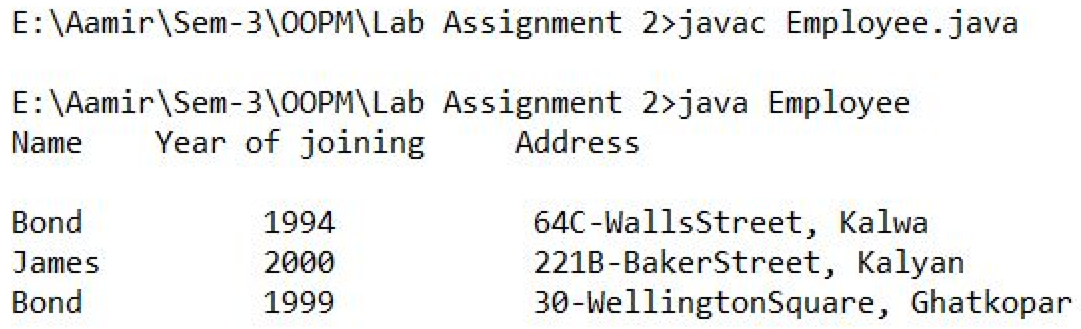
Write a program that would print the information (name, year of joining, salary, address) of three employees by creating a class named 'Employee'. (Hint: illustrate the concept of class and object)

```
//code

class Employee {
	String empName;
	int empYear;
	String empAddress;

	Employee(String name, int year, String address) {
		empName = name;
		empYear = year;
		empAddress = address;
	}
	void display() {
		System.out.println(empName+"        "+empYear+"          "+empAddress);
	}

	public static void main(String args[]) {
		Employee emp1 = new Employee("Bond ", 1994, "64C-WallsStreet, Kalwa");
		Employee emp2 = new Employee("James", 2000, "221B-BakerStreet, Kalyan");
		Employee emp3 = new Employee("Bond ", 1999, "30-WellingtonSquare,
Ghatkopar");
		System.out.println("Name    Year of joining    Address");
		System.out.println();
		emp1.display();
		emp2.display();
		emp3.display();
	}
}
```

//output



```
E:\Aamir\Sem-3\OOPM\Lab Assignment 2>javac Employee.java

E:\Aamir\Sem-3\OOPM\Lab Assignment 2>java Employee
Name    Year of joining     Address

Carl         1994           64C-WallsStreat, Kalwa
Mike         2000           68D-WallsStreat, Kalwa
John         1999           26B-WallsStreat, Kalwa
```



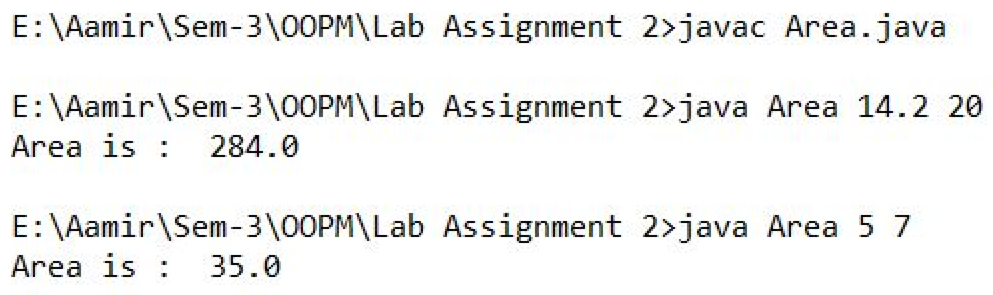
```
E:\Aamir\Sem-3\OOPM\Lab Assignment 2>javac Employee.java

E:\Aamir\Sem-3\OOPM\Lab Assignment 2>java Employee
Name    Year of joining     Address

Bond          1994           64C-WallsStreet, Kalwa
James         2000           221B-BakerStreet, Kalyan
Bond          1999           30-WellingtonSquare, Ghatkopar
```

Write a program to print the area of a rectangle by creating a class named 'Area' having two methods. First method named as 'setDim' takes the length and breadth of the rectangle as parameters and the second method named as 'getArea' returns the area of the rectangle. Length and breadth of rectangle are entered through the command line.

//code

```
class Area {
        double len, bre;

        void setDim(double l, double b) {
                this.len = l;
                this.bre = b;
                System.out.println("Area is :  "+getArea());
        }
        double getArea() {
                return (len * bre);
        }

        public static void main(String args[]) {
                double length = Double.parseDouble(args[0]);
                double breadth = Double.parseDouble(args[1]);
                Area obj = new Area();
                obj.setDim(length, breadth);
        }
}
```

//output



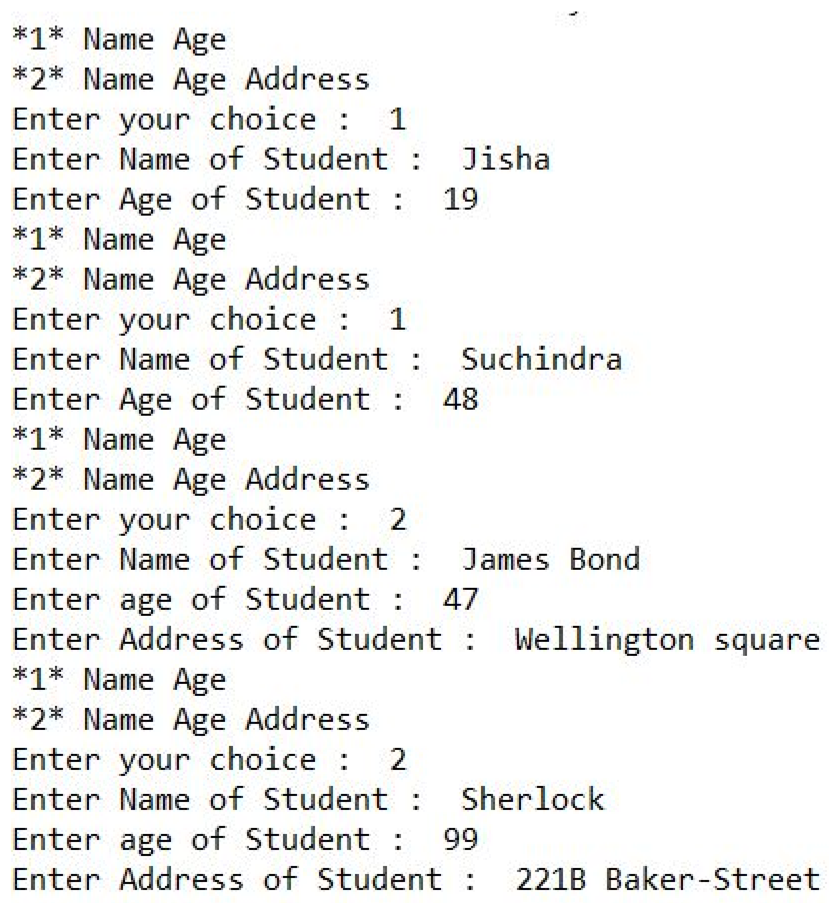
```
E:\Aamir\Sem-3\OOPM\Lab Assignment 2>javac Area.java

E:\Aamir\Sem-3\OOPM\Lab Assignment 2>java Area 14.2 20
Area is :  284.0

E:\Aamir\Sem-3\OOPM\Lab Assignment 2>java Area 5 7
Area is :  35.0
```

Create a class 'Student' with three data members which are name, age and address. The constructor of the class assigns default values name as "unknown", age as '0' and address as "not available". It has two methods with the same name 'setInfo'. First method has two parameters for name and age and assigns the same whereas the second method takes three parameters which are assigned to name, age and address respectively. Print the name, age and address of 10 students.

```
//code
import java.util.*;
import java.lang.*;
import java.io.*;

class Student {

	String name, addr;
	int age=0;
	Student() {
		this.name = "unknown";
		this.age = 0;
		this.addr = "not available";
	}
	void setInfo(String name, int age, String addr) {
		this.name = name;
		this.age = age;
		this.addr = addr;
	}
	void setInfo(String name, int age) {
		this.name = name;
		this.age = age;
	}
	void display() {
		System.out.println("________________________________________");
		System.out.println("Name: "+this.name);
		System.out.println("Age: "+this.age);
		System.out.println("Address: "+this.addr);
	}
}

class ObjectArray {
	public static void main(String args[]) {
		int choice, age1, age2;
		String name1, name2, address1, address2;
		Scanner sc = new Scanner(System.in);
		//input
		Student obj[] = new Student[10];
		for(int i=0 ; i<10 ; i++) {
			obj[i] = new Student();
			System.out.println("*1* Name Age");
			System.out.println("*2* Name Age Address");
			System.out.print("Enter your choice :  ");
			choice = sc.nextInt();
			sc.nextLine();
			switch(choice) {
				case 1:
					System.out.print("Enter Name of Student :  ");
					name1 = sc.nextLine();
					System.out.print("Enter Age of Student :  ");
					age1 = sc.nextInt();
					//sc.nextLine();
					obj[i].setInfo(name1, age1);
					break;
				case 2:
					System.out.print("Enter Name of Student :  ");
					name2 = sc.nextLine();
					System.out.print("Enter age of Student :  ");
					age2 = sc.nextInt();
					sc.nextLine();
					System.out.print("Enter Address of Student :  ");
					address2 = sc.nextLine();
					obj[i].setInfo(name2, age2, address2);
					break;
				default :
					System.out.println("Invalid choice");
			}
		}
		//display
		for (int i=0 ; i<10 ; i++) {
			obj[i].display();
		}
	}
}
```

//output

```
E:\Aamir\Sem-3\OOPM\Lab Assignment 2>javac ObjectArray.java

E:\Aamir\Sem-3\OOPM\Lab Assignment 2>java ObjectArray
*1* Name Age
*2* Name Age Address
Enter your choice :  2
Enter Name of Student :  Aamir Ansari
Enter age of Student :  19
Enter Address of Student :  Kalwa
*1* Name Age
*2* Name Age Address
Enter your choice :  2
Enter Name of Student :  Krishna
Enter age of Student :  19
Enter Address of Student :  Thane
*1* Name Age
*2* Name Age Address
Enter your choice :  2
Enter Name of Student :  Kanaiya
Enter age of Student :  18
Enter Address of Student :  Rajkot
*1* Name Age
*2* Name Age Address
Enter your choice :  1
Enter Name of Student :  Ninad
Enter Age of Student :  18
*1* Name Age
*2* Name Age Address
Enter your choice :  1
Enter Name of Student :  Sreekesh
Enter Age of Student :  59
*1* Name Age
*2* Name Age Address
Enter your choice :  2
Enter Name of Student :  Isha
Enter age of Student :  18
Enter Address of Student :  Kalyan
```



```
*1* Name Age
*2* Name Age Address
Enter your choice :  1
Enter Name of Student :  Jisha
Enter Age of Student :  19
*1* Name Age
*2* Name Age Address
Enter your choice :  1
Enter Name of Student :  Suchindra
Enter Age of Student :  48
*1* Name Age
*2* Name Age Address
Enter your choice :  2
Enter Name of Student :  James Bond
Enter age of Student :  47
Enter Address of Student :  Wellington square
*1* Name Age
*2* Name Age Address
Enter your choice :  2
Enter Name of Student :  Sherlock
Enter age of Student :  99
Enter Address of Student :  221B Baker-Street
```

```
Name: Aamir Ansari
Age: 19
Address: Kalwa

Name: Krishna
Age: 19
Address: Thane

Name: Kanaiya
Age: 18
Address: Rajkot

Name: Ninad
Age: 18
Address: not available

Name: Sreekesh
Age: 59
Address: not available

Name: Isha
Age: 18
Address: Kalyan

Name: Jisha
Age: 19
Address: not available

Name: Suchindra
Age: 48
Address: not available

Name: James Bond
Age: 47
Address: Wellington square

Name: Sherlock
Age: 99
Address: 221B Baker-Street
```