Chapter 9 OBJECT AND CLASS

- *Relation between an Object and a Class*
- *Access Specifiers*
- *Multiple objects of a class*
- *Static data/methods*
- *Using Current object using operator* ***this***
- *Objects as arguments ; Arrays of objects*
- *Programs on the above*

Class and Objects of Real World

A **class** bounds data and methods that are relevant to one task. An **object** is an instanc of a class. Nearly all that we see around us are objects of various classes. An object is a physical entity and a class is a conceptual view of it. Without an object, a class cannot be brought to reality. A class can have many objects.

For Example :

Class Telephone : A device that is used for talking to people.

Object telephone : Shown is an object of class telephone.

Class Clock : A covered board showing numbers from 1 to 12 in circular manner and 2 or 3 pointers which move in circular manner to show the time.

Object Clock : Shown is an object of class Clock.

Class School : A named building that has an administrative office, teaching and non-teaching staff, and students. A school also has a set of activities known as its behaviour/methods.

Object School : Various schools are objects of this class.

Objects having Attribute and Behaviour

Q. 1. What is attribute of an object of a class?

Ans. Attributes are data that are relevant to various objects of a class. They may contain different values for different objects of the same class.

For example, attributes of class Clock are Color, Cost, Size, Weight etc.

Attributes of class Book are Subject, Cost, Language, Category etc.

Q. 2. What is behaviour of an object of a class?

Ans. Behaviour is the kind of action taken by an object upon executing some process. Various objects of a class have the same kind of behaviour or methods.

For example, one behaviour of class Table-Clock is Ring-Alarm. It is not a value but is a process that has some instructions.

One behaviour of class School is Examination. It is not a value but is a process. Another behaviour of class School is Sports. Behaviour/methods have a course of action to be followed.

Applying Class and Objects in a Program

Q. 3. What is Objet Oriented Programming (OOP)?

Ans. It is a kind of programming in which data and related methods are bound together to create a class and then objects of that class are created. One class can have multiple objects.

Attributes or data of various objects of a class can have same/different values.

Methods of various objects of a class can be executed at the same/different time.

An Object Oriented Language allows us to write an Object Oriented Program.

OOP has some features :

(a) Encapsulation : All related data and methods enclosed within one boundary (a class).

(b) Inheritance : A derived (child) class being able to access data/methods of its base (parent) class. This feature helps in code reusability.

(c) Polymorphism : Ability of methods to have same name but to behave differently under different situations.

(d) Dynamic Binding : Programs written in such a way that during runtime the linking of data and methods is taken. This feature provides flexibility to a program.

(e) Data Hiding : This feature gives stress on correct way of handling of data and methods. It states that not all data and methods can be directly connected. Access specifiers are used to implement data hiding.

Q. 4. What does a class contain? What are class members?

Ans. A class contains member data, member methods and a special method called constructor. All the member methods can access and modify the member data. They can also access other member methods. The member methods contain commands to perform a specific task. All classes, in general, contain a controlling method called main().

Q. 5. How are objects related to a class?

Ans. Objects are instances of a class. A class can have multiple objects. Each object can execute the methods of the class simultaneously. The objects have certain access specifiers for accessing the methods of the class.

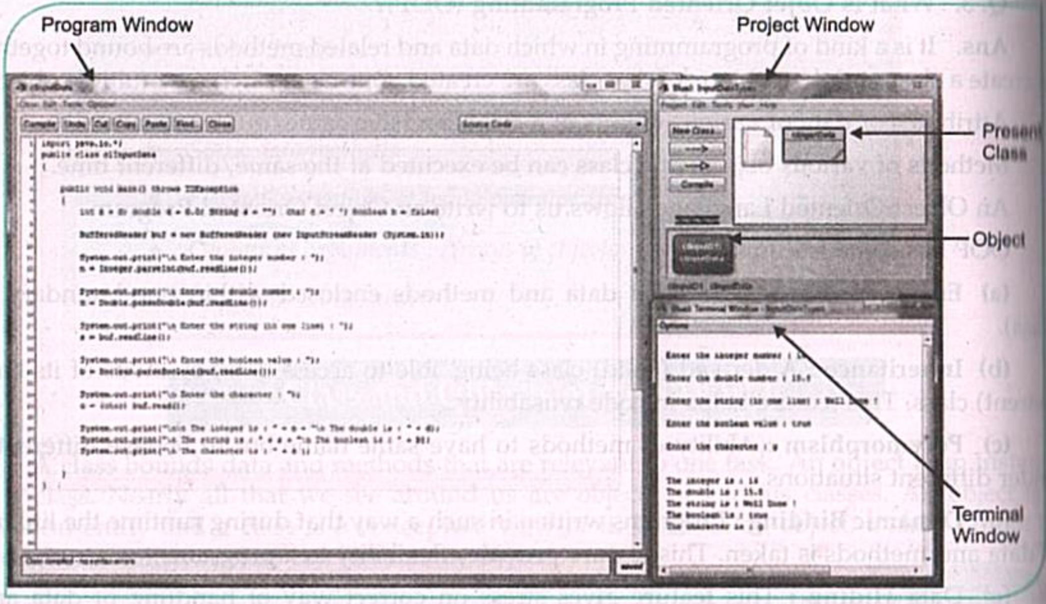
Q. 6. What are access specifiers?

Ans. Access Specifiers are keywords that help in implementing data hiding. They specify which (data/method) can be made to access by which method.

Java has the following Access Specifiers :

	Access Specifier	Access Specification
1.	public	can be accessed by any method or object
2.	private	can be accessed only by methods of the same class
3.	protected	can be accessed only by methods of the • same class • subclass • classes of the same package
4.	default (no modifier)	can be accessed only by methods of the • same class • classes of the same package

Example 1 : Shown below is a Java program in BlueJ window.



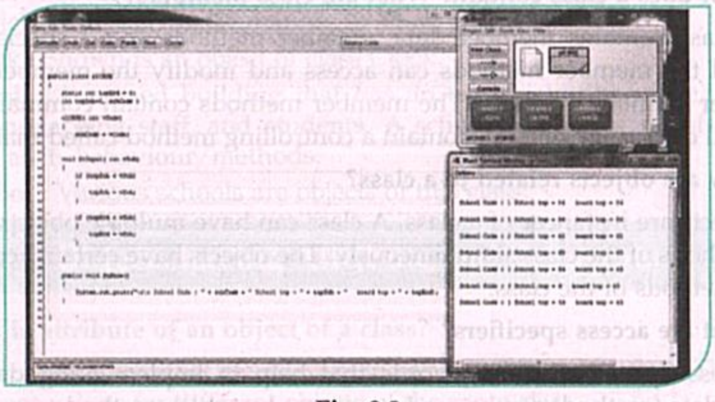
```
import java.io.*;
public class allInputData
{

    public void main() throws IOException
    {
        int n = 0; double d = 0.0; String s = ""; char c = ' '; boolean b = false;

        BufferedReader buf = new BufferedReader (new InputStreamReader (System.in));

        System.out.print("\n Enter the integer number : ");
        n = Integer.parseInt(buf.readLine());

        System.out.print("\n Enter the double number : ");
        d = Double.parseDouble(buf.readLine());

        System.out.print("\n Enter the string (in one line) : ");
        s = buf.readLine();

        System.out.print("\n Enter the boolean value : ");
        b = Boolean.parseBoolean(buf.readLine());

        System.out.print("\n Enter the character : ");
        c = (char) buf.read();

        System.out.print("\n\n The integer is : " + n + "\n The double is : " + d);
        System.out.print("\n The string is : " + s + "\n The boolean is : " + b);
        System.out.print("\n The character is : " + c );

    }
}
```

```
Class compiled - no syntax errors
```

```
Enter the integer number : 14

Enter the double number : 15.8

Enter the string (in one line) : Well Done !

Enter the boolean value : true

Enter the character : g


The integer is : 14
The double is : 15.8
The string is : Well Done !
The boolean is : true
The character is : g
```

Fig. 9.1

Example 2 : Shown below is a BlueJ window screen showing a Java program, a class with 3 objects and an output screen.



```
[illegible]
```

Fig. 9.2

The name of the class is clCBSE. There are 3 objects – clCBSE1, clCBSE2, clCBSE3. The program has a parameterized constructor and 2 more methods and 3 member data. The output screen shows the execution of the methods of the class through the objects.

Q. 7. What is user defined data and how is it different from built in data?

Ans. Built in data are those which are recognized by the compiler. They always represent the same kind of variable. They are also called *primitive data.*

User defined data are those which are created in the program by the programmer. They are not recognized by the compiler. They are directly or indirectly made up of built in data members. Their specification may vary from one program to another. They are also called *composite data.*

Q. 8. What is memory allocation?

Ans. Data items that are used in program require certain amount of memory space. The amount of memory space required varies from one data type to another. The process of allocating memory to a data item in a program is called memory allocation.

There are two types of memory allocation – static and dynamic.

Static Memory Allocation : In this type of memory allocation, the amount of memory required by a data is fixed. For example, the built in data types allocated same amount of memory space every time they are declared.

Variable Type	Description	Size
byte	Byte length integer	1 byte
short	Short integer whole numbers	2 bytes
int	Integer whole numbers	4 bytes
long	Bigger integer whole numbers	8 bytes
float	Fractional numbers	4 bytes
double	Bigger fractional numbers	8 bytes
char	Keyboard characters	2 bytes
boolean	Values of type True or False	1 byte

Dynamic Memory Allocation : In this type of memory allocation, the amount of memory required by a data is not fixed. It is mainly applicable on user defined data types such as arrays and objects.

Keyword new is used for dynamic allocation of memory.

```
Example : int na[ ] = new int [10];
          double da [ ] = new double [10];
          Student s1 = new Student(); // Student is a user defined data
```

Q. 9. What is the use of keyword 'new'? What happens when a new object is created?

Ans. Keyword new is used to allocate memory. In case of composite data types, memory requirement is not fixed as in case of primitive data type. Size of composite data types can vary from one another. The compiler therefore cannot allot a predefined memory to objects of these kinds. Keyword new is used to allocate memory area depending on the present size of an object/array.

Example 1

int ar[ ] = new int [5];

Example 2

Student sm = new Student(); // Student is a user defined data, whose members are not
// recognized by the compiler

Q. 10. What is the use of keyword 'static'? [Application shown in school objects]

Ans. A class can have multiple objects, where the objects can invoke the members of the class individually. But some members of the class are such which are to be shared by all the objects. Those members are called "static" members.

Objects cannot make a personal copy of the "static" members. If any object changes the value of a static data member, the change is reflected to all other objects.

Static members of a class are called **Class Members** and non-static members of a class are called **Instance Members.**

Q. 11. What is a Constructor ? What is its use ?

Ans. A constructor is a special method of a class that gets executed as soon as an object of that class is created. It indicates that some instructions has to be executed before the actual work starts. [For e.g., value of full marks is set even before the exam begins and the student's paper is evaluated]

So we can say, a constructor is a special method of a class that has following features,

- A constructor has the same name as that of the class.
- It cannot be called as other functions. It gets automatically executed whenever an object of that class is created.
- It does not have any return type, not even void.
- It may or may not take arguments.
- A constructor can be overloaded.
- If any code is written under a constructor, it will get executed as soon an object of that class is created.
- A constructor is used to initialize the member data of a class to their initial values. In general, the member data are initialized to null.

Q. 12. What happens in case of a parameterized constructor?

Ans. A parameterized constructor takes values for its parameters when an object is created. The parameters' can be used in the constructor for initiating values of member data of the class.

Note :

- *A Constructor cannot be made static. It is because a constructor gets activated after an object is created.*

Real World programming examples in terms of classes and objects – [only class definition]

Example 1 : Calculator

Member Data : double operand1, operand2 ; int counter ; String operator

Member Method : addition, subtraction, multiplication, division, square_root, median, tan,

The class in program form with partial details would be like

```
class Calculator
{
        double op1, op2;        int counter;
        public double addition(int p1, int p2);
        public double subtraction (int p1, int p2);
        public double multiplication (int p1, int p2);
        public double division (int p1, int p2);
        public double median (int p1, int p2, int p3, int p4);
        public double square_root (int p1);
        public void graphic_presentation( );
        // detail of the methods can be written here
}
```

Various objects of class Calculator can be created. Shown below are some instances. They can exist and invoke different methods simultaneously.

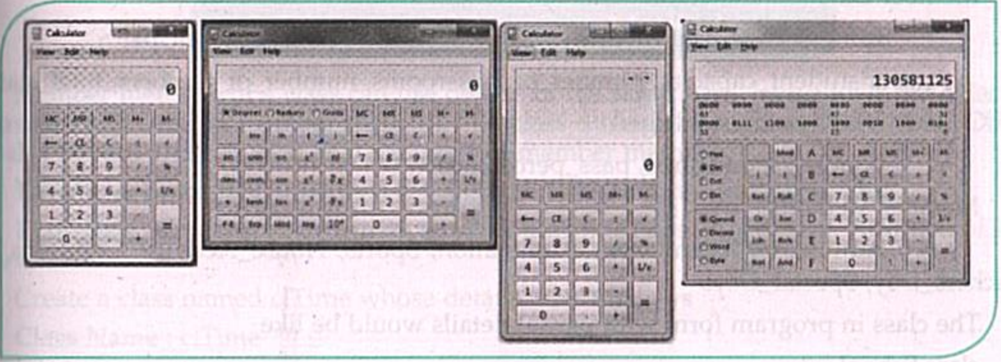


Fig. 9.3

The first calculator may perform addition of 2 numbers, the second may perform square root of a number, the third may find the exponential values of 2 numbers and the fourth may find the logarithmic value of a number.

Thus, multiple objects can exist together and can invoke different methods of the class simultaneously.

Example 2 : Super Market

Member Data

String Name, Address, Item_List as array,
int capacity, Number_of_items, Number _of_counters
double amt_invested,

Member Methods

Sale, Stock_Verification, Stock_Purchase, Stock_Clearance, Store_Cleaning

The class in program form with partial details would be like

```
class Super_Market
{
        String Name, Address, Item_List as array;
        int capacity, Number_of_items, Number _of_counters, Number_of_Work
        double amt_invested ;
        public int Sale( );
        public void Stock_Verification ( );
        public void  Stock_Purchase ( );
        public void Stock_Clearance ( );
        public void Store_Cleaning ( );
        //detail of the methods can be written here
}
```

Fig. 9.4

The various supermarkets are objects of class Super_Market. Multiple objects can exi and can have different values for their members. Different objects can invoke differen methods simultaneously.

Example 3 : School

Member Data

String School_Name, Address, Board_Name, Name_of_Principal
intstudent_capacity, number_of_classrooms, number_of_teachers, total_staff,
year_established
double ground_area, pass_percentage

Member Methods

Admission, Attendance, Examination, Sports, House_Activity, Childrens_Day, Teachers_Day, Special_Days

The class in program form with partial details would be like

```
class School
{
      String School_Name, Address, Board_Name, Name_of_Principal;
      int student_capacity, number_of_classrooms, number_of_teachers, total_staff,
            year_established;
      double ground_area, pass_percentage ;
      public void Admission ( );
      public void  Attendance ( );
      public void Examination ( );
      public void Sports ( );
      public void House_Activity ( );
      public void  Childrens_Day ( );
      public void Teachers_Day ( );
      public void Special_Days ( );
      // detail of the methods can be written here
}
```

Fig. 9.5

The various schools are objects of class School. Each school is an object of class School. They all can have different values for the member data and can invoke the methods independently.

Note :

- *That only non-static members can be accessed by the objects independently. Static members are common to all objects. They are called class members. Static members can be accessed even without using objects.*

Application of static members in a class :

Through the class School, we can observe that some methods should be made static. For example, "Board_Examination". The schools will not be able to execute this method independently. Thus method "Board_Examiantion" will be considered a static method.

Another variable can be considered to be static, maximum student in a class. It can be assumed that the board sets a rule that no school should have more than a given number of students in a class. Then that value would be common for all the objects.

Objects and Member Methods

Member methods of a class can use objects of the same class. Java allows the objects to be created and used in member methods of a class during the creation of the class. Objects can also be sent and returned as parameters of member methods.

The above concept is illustrated through the following program examples.

Example 1 : Creating objects in methods of the same class. Program of Addition of Time Intervals.

Create a class named clTime whose details are as follows :

Class Name : clTime

Member Data : int hh, int mm

Member Methods :

(i) clTime() : Constructor to initialize member data to null

(ii) void fnInput() : to input values for member data

(iii) void fnShow() : to print member data as Time [7 hours : 45 min]

(iv) void fnAddTime () : to create two objects of clTime, fill value in them using fnInput(), add the time intervals and print the three time intervals using fnShow().

(v) void main() : the controlling function

```
import java.util.*;
public class clTtime
{
    int hh, mm;
    clTtime()
    {
        hh = 0; mm = 0;
    }
    void fnInputTime()
    {
```