

Course Name:
INTRODUCTION TO JAVA

Course code: 18IS6IEJVA
Semester: 6 A

OOPs Concepts in Java- Basics with Examples

1. Object-Oriented Programming System (OOPs)

OOPs is a programming concept that works on the principles of abstraction, encapsulation, inheritance, and polymorphism. It allows users to create objects they want and create methods to handle those objects. The basic concept of OOPs is to create objects, re-use them throughout the program, and manipulate these objects to get results.

OOP meaning "Object Oriented Programming" is a popularly known and widely used concept in modern programming languages like Java.

Following are the list of OOPs Concepts in Java with Examples

The following are general OOPs concepts in Java:

- **Class**

The class is one of the Basic concepts of OOPs which is a group of similar entities. It is only a logical component and not the physical entity. Lets understand this one of the OOPs Concepts with example, if you had a class called “Expensive Cars” it could have objects like Mercedes, BMW, Toyota, etc. Its properties(data) can be price or speed of these cars. While the methods may be performed with these cars are driving, reverse, braking etc.

- **Object**

An object can be defined as an instance of a class, and there can be multiple instances of a class in a program. An Object is one of the Java OOPs concepts which contains both the data and the function, which operates on the data. For example - chair, bike, marker, pen, table, car, etc.

- **Inheritance**

Inheritance is one of the Basic Concepts of OOPs in which one object acquires the properties and behaviors of the parent object. It's creating a parent-child relationship between two classes. It offers robust and natural mechanism for organizing and structure of any software.

- **Polymorphism**

Polymorphism refers to one of the OOPs concepts in Java which is the ability of a variable, object or function to take on multiple forms. For example, in English, the verb run has a different meaning if you use it with a laptop, a foot race, and business. Here, we understand the meaning of run based on the other words used along with it. The same also applied to Polymorphism.

- **Abstraction**

Abstraction is one of the OOP Concepts in Java which is an act of representing essential features without including background details. It is a technique of creating a new data type that is suited for a specific application. Lets understand this one of the OOPs Concepts with example, while driving a car, you do not have to be concerned with its internal working. Here you just need to concern about parts like steering wheel, Gears, accelerator, etc.

- **Encapsulation**

Encapsulation is one of the best Java OOPs concepts of wrapping the data and code. In this OOPs concept, the variables of a class are always hidden from other classes. It can only be accessed using the methods of their current class. For example - in school, a student cannot exist without a class.

- **Association**

Association is a relationship between two objects. It is one of the OOP Concepts in Java which defines the diversity between objects. In this OOP concept, all object have their separate lifecycle, and there is no owner. For example, many students can associate with one teacher while one student can also associate with multiple teachers.

- **Aggregation**

In this technique, all objects have their separate lifecycle. However, there is ownership such that child object can't belong to another parent object. For example consider class/objects department and teacher. Here, a single teacher can't belong to multiple departments, but even if we delete the department, the teacher object will never be destroyed.

- **Composition**

Composition is a specialized form of Aggregation. It is also called "death" relationship. Child objects do not have their lifecycle so when parent object deletes all child object will also delete automatically. For that, let's take an example of House and rooms. Any house can have several rooms. One room can't become part of two different houses. So, if you delete the house room will also be deleted.

1.2 Advantages of OOPs (Object-Oriented Programming System):

Following are few advantages

- OOPs Concepts in Java offer easy to understand and a clear modular structure for programs.
- Objects created for Object-Oriented Programs can be reused in other programs. Thus it saves significant development cost.
- Large programs are difficult to write, but if the development and designing team follow OOPS concepts, then they can better design with minimum flaws.
- It enhances program modularity because every object exists independently.

1.3 Comparison of OOPS with other programming styles with help of an Example

Let's understand with example how Java OOPs Concepts are different than other programming approaches. Programming languages can be classified into 3 primary types

1. **Unstructured Programming Languages:** The most primitive of all programming languages having sequentially flow of control. Code is repeated throughout the program
2. **Structured Programming Languages:** Has non-sequentially flow of control. Use of functions allows for re-use of code.
3. **Object Oriented Programming Languages:** Combines Data & Action Together.

Let's understand these 3 types with an example. Suppose we want to create a Banking Software with functions like

1. Deposit
2. Withdraw
3. Show Balance

1.3.1 Unstructured Programming Languages

The earliest of all programming language were unstructured programming language. A very elementary code of banking application in unstructured Programming language will have two variables of one account number and another for account balance

Example:

```
int account_number=20;  
int account_balance=100;
```

Suppose deposit of 100 dollars is made.

```
account_balance=account_balance+100
```

Next you need to display account balance.

```
printf("Account Number=%d,account_number)\n");\nprintf("Account Balance=%d,account_balance)\n");
```

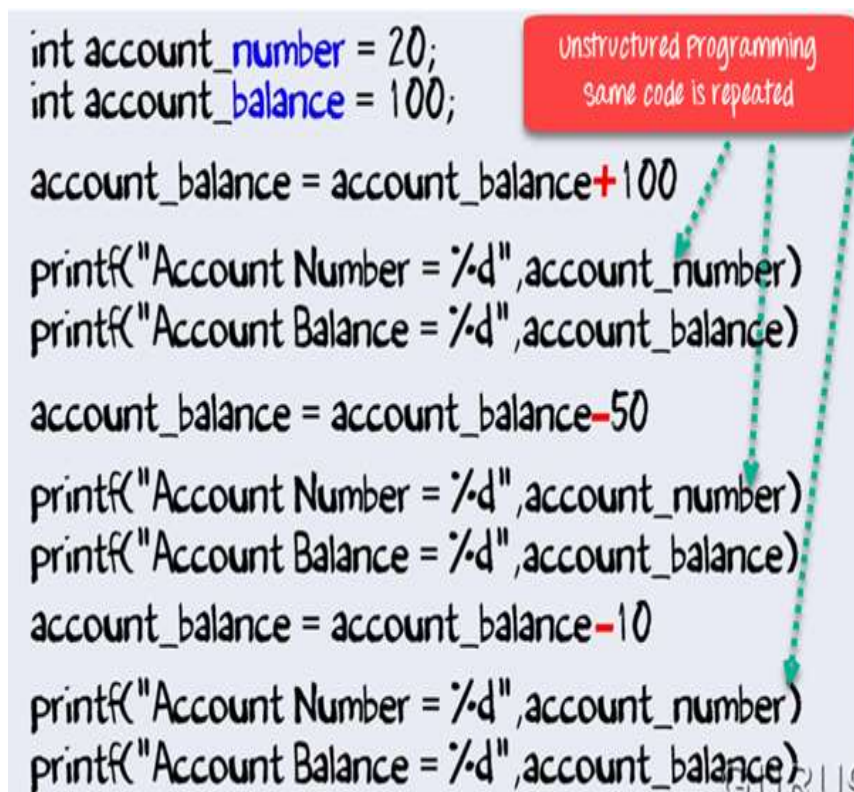
Now the amount of 50 dollars is withdrawn.

```
account_balance=account_balance-50
```

Again, you need to display the account balance.

```
printf("Account Number=%d,account_number)\n");\nprintf("Account Balance=%d,account_balance)\n");
```

The entire code combining all the lines of code mentioned is below



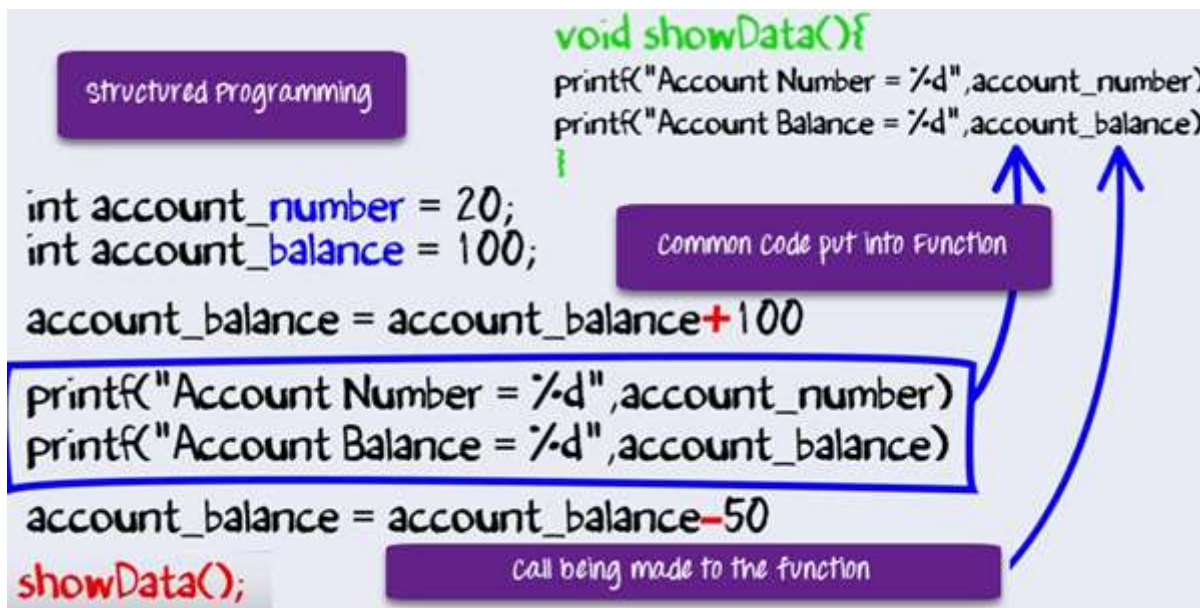
```
int account_number = 20;\nint account_balance = 100;\n\naccount_balance = account_balance+100\n\nprintf("Account Number = %d",account_number)\nprintf("Account Balance = %d",account_balance)\n\naccount_balance = account_balance-50\n\nprintf("Account Number = %d",account_number)\nprintf("Account Balance = %d",account_balance)\n\naccount_balance = account_balance-10\n\nprintf("Account Number = %d",account_number)\nprintf("Account Balance = %d",account_balance)
```

Unstructured Programming
same code is repeated

For any further deposit or withdrawal operation – you will code repeat the same lines again and again.

1.3.2 Structured Programming

With the arrival of Structured programming repeated lines on the code were put into structures such as functions or methods. Whenever needed, a simple call to the function is made.



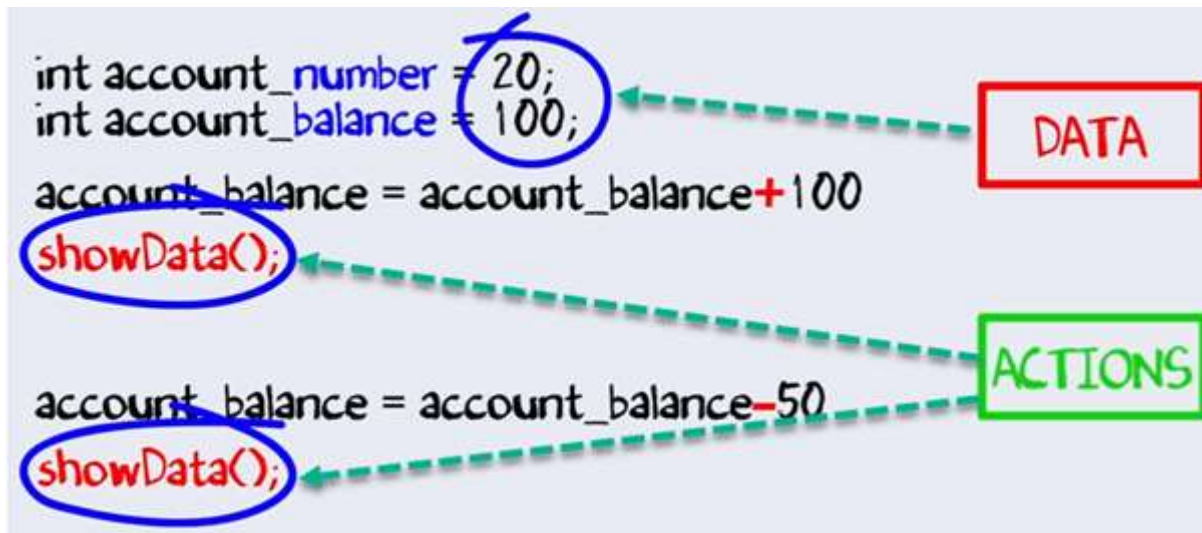
2. Object-Oriented Programming

In our program, we are dealing with data or performing specific operations on the data. In fact, having data and performing certain operation on that data is very basic characteristic in any software program. Experts in Software Programming thought of combining the Data and Operations. Therefore, the birth of Object Oriented Programming which is commonly called OOPS.

The same code in Object Oriented Programming languages will have same data and some action performed on that data.

Example:

```
Class Account{
    int account_number;
    int account_balance;
    public void showdata(){
        system.out.println("Account Number"+account_number)
        system.outprintln("Account Balance"+ account_balance)
    }
}
```



By combining data and action, we will get many advantages over structural programming viz,

- Abstraction
- Encapsulation
- Inheritance
- Polymorphism