Object-oriented Programming

It focuses on implementing real world objects using **Classes** to create variation of **Objects** that has attributes and purpose.

It helps us create much **flexible** and **efficient** code than **procedural programming.**

Classes

It is **created by** the programmer; it will act as a **blueprint** of an object that you want to implement in your program.

They **contain** all the **attributes** and **methods** that your desired **object** should have.

Objects

It is **created** by instantiating a **class**. It is anything that has an **attribute** and a **purpose**.

Example: Person, Furniture, and Food

Attributes

These are the **global variables** declared inside the **class** of our **object**. It is used to **create variations** of an **object** using only one class.

Class Creation

```
Modifiers className class{
     // Attributes
    // Methods or Purpose
}
```

Person

- First Name
- Last Name
- Sex
- Age

```
public Person class{
      // Attributes
      String firstName;
      String lastName;
      char sex;
      int age;

     // Methods or Purpose
}
```

Class Instantiation

The process of creating an **Object** using a **class** so we can use it in our program.

ClassName identifier = new **ClassName**();

Person p = new **Person**();

Accessing Attributes

ClassName identifier = new **ClassName**();

WRITING Attributes

identifier.**attribute** = value;

READING Attributes

System.out.println(*identifier.**attribute***);

Constructors

It is the **method** called when you **instantiate a class/ create an object**. It is used to **initialize** the **attributes** of an **object** or **run** a **block of code** when an **object is created**.

CREATING Constructors

Constructor methods are named after their **Class Name**.

```
modifiers className class{
    className(){
        // Constructor
    }
}
```

```
public Product class{
    Product(){
          System.out.println(“Product Created”);
    }
}
```

Constructors are used to initialize attributes.

```
public Product class{

      String name;
      float price;

      Product(String name, float price){
            this.name = name;
            this.price = price;
```

```
        }
    }
```

THIS Keyword

The **this** keyword refers to the **class** itself.

The **this keyword** will **enables** you to **access global variables** inside the **class** if you have the **same variable names** in a **parameter**.

USING Constructors

```
ClassName identifier = new ClassName(parameters);

Product p1 = new Product("Milk", 150.0f);
Product p2 = new Product("Noodles", 15.25f);
Product p3 = new Product("Softdrinks", 12.50f);
```

USER INPUT Object Creation

Create a class of your choice then create an object from that class using user input.

Object Methods

Are **methods** declared inside an **Object Class. Object Methods** are **considered** as the **Object's purpose**.

CREATING Object Methods

Object Methods are the same as the Methods we talked about.

```
modifiers className class{

    modifiers returntype methodName(arguments){
        // Do Anything Here
```

```
      }
 }

public Character class{
          String name, dialog;
          int hp, mp, lvl;

          void introduce(){

         System.out.printn("I' am " + name);
          }
}
```

CALLING Object Methods

Object Methods are the same as the Methods we talked about.

SYNTAX

```
ClassName cn = new ClassName(constructor);
cn.methodName(arguments);
```

EXAMPLE

```
Character c = new Character (constructor);
c.introduce();
```