CONSTRUCTORS

Fundamentals

- A **constructor** is used in the creation of an object that is an instance of a class using the `new` keyword.
  Ex. `Employee emp = new Employee();`
  - A constructor performs operations required to initialize the class before methods are invoked or fields are accessed.
    Ex.
    ```
    public Employee() {
        salary = 15000;
    }
    ```
    Explanation: Every `Employee` object created will have a default starting salary of 15000 per month.
  - Constructors are never inherited.
  - Constructor declarations use the name of the class and have no return type.
- If you do not include any constructors in a class, Java provides a **default constructor**, a constructor with an empty parameter list and body. This is invisibly added to the class. It is also known as a **no-argument constructor**.
  Ex. `public Employee() { }`
- The **`this`** keyword is used when the instance variable has the same name with the constructor's parameter.
  Example:
  ```
  public class Employee {
     private double salary;
     public Employee(double salary) {
       this.salary = salary;
     }
  }
  ```

Constructor Overloading

- **Constructor overloading** occurs when constructors have different type parameters.
  Example:
  ```
  public class Student {
     private String name;
     private int age;
     public Student() {
       name = "No name yet.";
       age = 0;
     }
     public Student(String name, int age) {
       this.name = name;
       this.age = age;
     }
     public static void main(String[] args) {
       Student s1 = new Student();
       System.out.println(s1.name + ", " + s1.age);
       s1.name = "Riven Reyes";
       s1.age = 17;
       System.out.println(s1.name + ", " + s1.age);
       Student s2 = new Student("Nika Pena", 18);
       System.out.println(s2.name + ", " s2.age);
     }
  }
  ```
  Explanation: Instances of the `Student` class can be created with or without arguments. `Student s1 = new Student();` calls the default constructor because it does not have arguments.
- When a constructor calls another constructor with a greater number of parameters, it is called **constructor chaining**. This is accomplished using the `this` keyword too. The `this()` call must be the first statement in the constructor.
  Example:
  ```
  public Student() {
       this("No name yet.");
  }
  public Student(String name) {
       this(name, 0);
  }
  public Student(String name, int age) {
       this.name = name;
       this.age = age;
  }
  ```

Reference:
Oracle Docs (n.d.). *Citing sources.* Retrieved from https://docs.oracle.com/javase/tutorial/java/javaOO/index.html