

1

# What is Java?

Java Introduction

Java is a popular programming language, created in 1995.

It is owned by Oracle, and more than **3 billion** devices run Java. It is used for:

* Mobile applications (especially Android apps)
* Desktop applications
* Web applications
* Web servers and application servers
* Games
* Database connection
* And much, much more!

# Why Use Java?

Java works on different platforms (Windows, Mac, Linux, Raspberry Pi, etc.) It is one of the most popular programming languages in the world

It has a large demand in the current job market It is easy to learn and simple to use

It is open-source and free

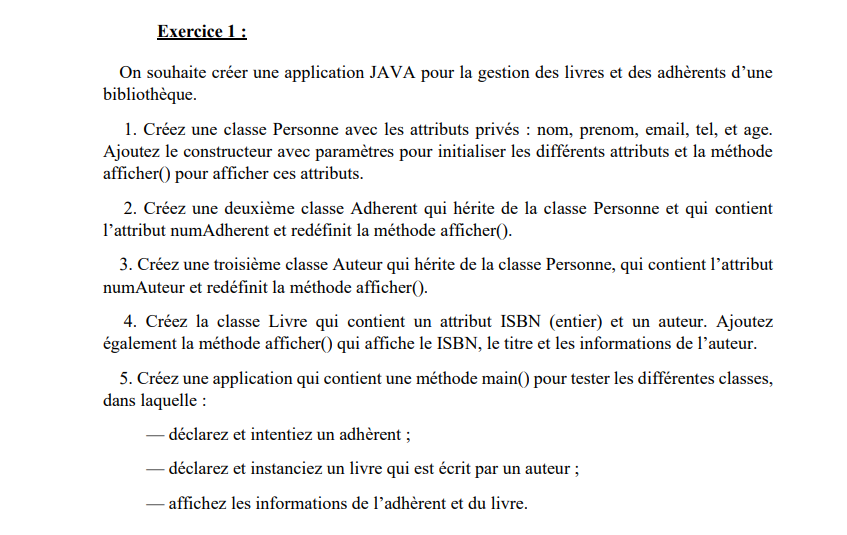
It is secure, fast and powerful

It has a huge community support (tens of millions of developers)

Java is an object-oriented language which gives a clear structure to programs and allows code to be reused, lowering development costs

As Java is close to [C++](https://www.w3schools.com/cpp/default.asp) and [C#](https://www.w3schools.com/cs/default.asp), it makes it easy for programmers to switch to Java or vice versa

# Exercise 1:



*Figure 1: Exercise 1*

* *In this exercise, we will develop a simple java application that manage books and authors*

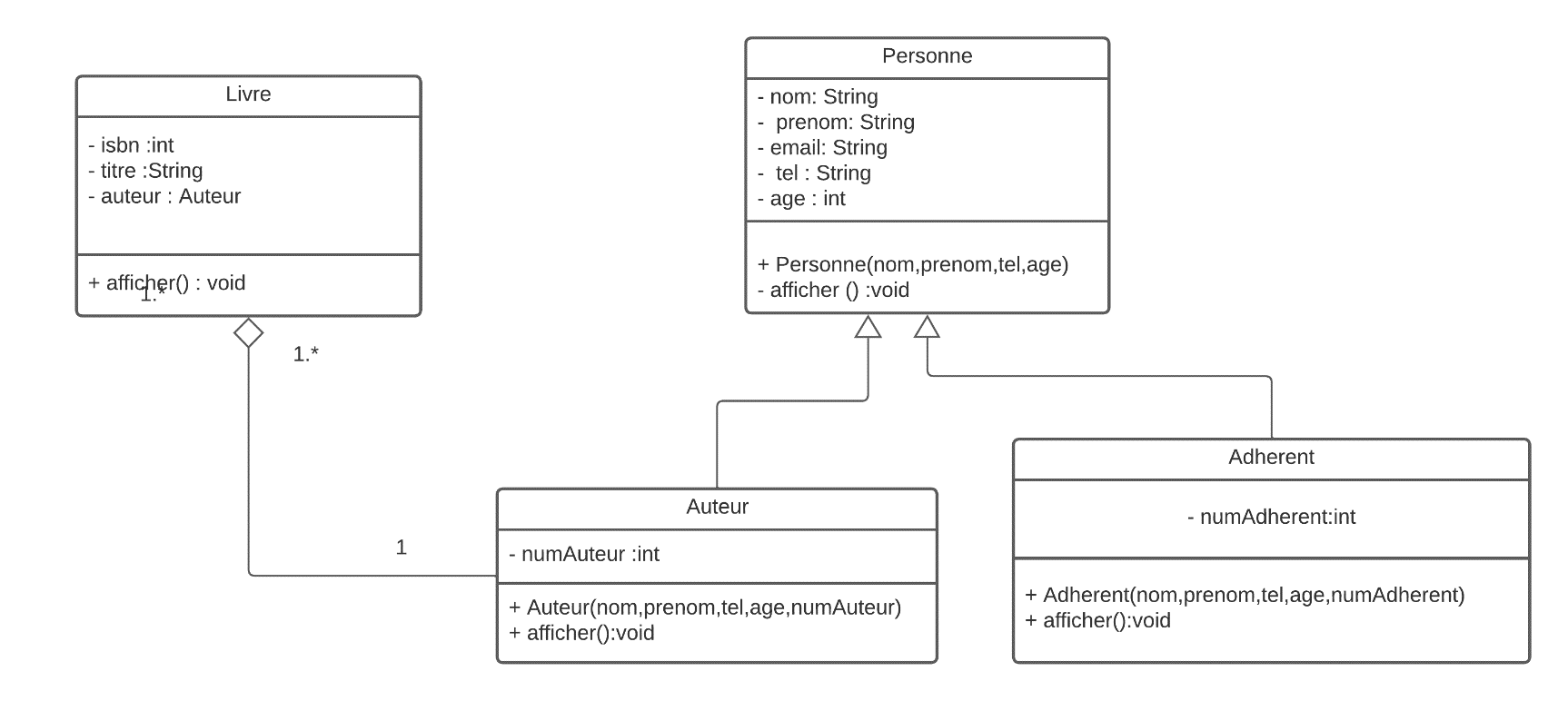
*Of a library.*

***Step 1: Conception***

*The*[*UML*](https://en.wikipedia.org/wiki/Unified_Modeling_Language)*Class diagram is a graphical notation used to construct and visualize object-oriented systems. A class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes*

*the structure of a system by showing the system's:*

* *classes,*
* *their attributes,*
* *operations (or methods),*
* *and the relationships among objects*

**

*Figure 2: UML Class Diagram*

*As Shown in the picture above there are four classes. Person class is the parent of two class (Adherent – Author), So the connection between Person class and the other class we call this inheritance.*

*Author-----<> Livre: we call this association aggregation*

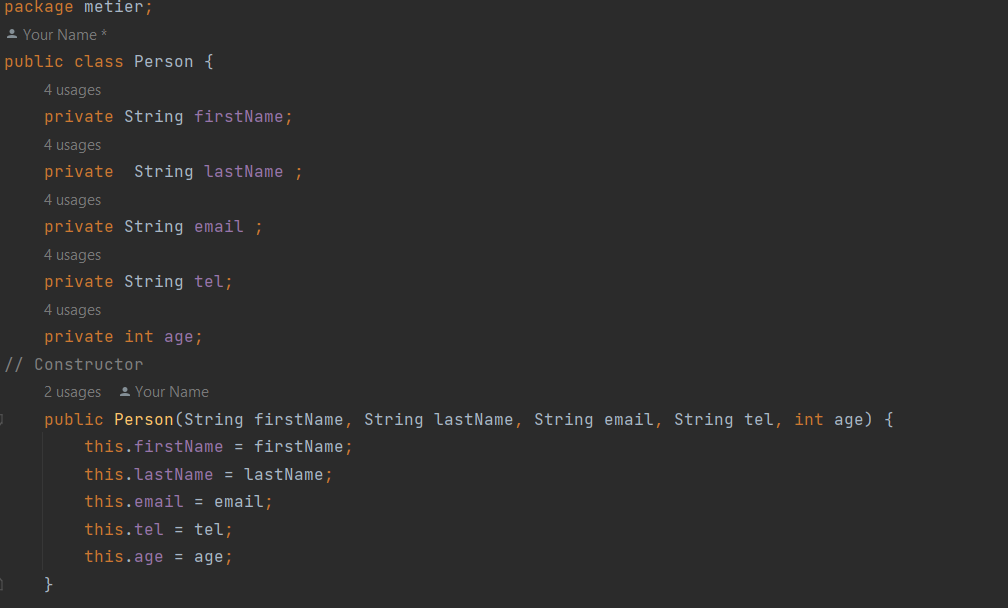
***Step 2: Development***

1. ***Person class:***

*In this class we automatically generate the following methods:*

***Constructors****: A constructor in Java is a****special method****that is used to initialize objects. The constructor is called when an object of a class is created. It can be used to set initial values for object attributes.*

***afficher() :*** *This method returns all information about an actor(name-phone-email).*

**

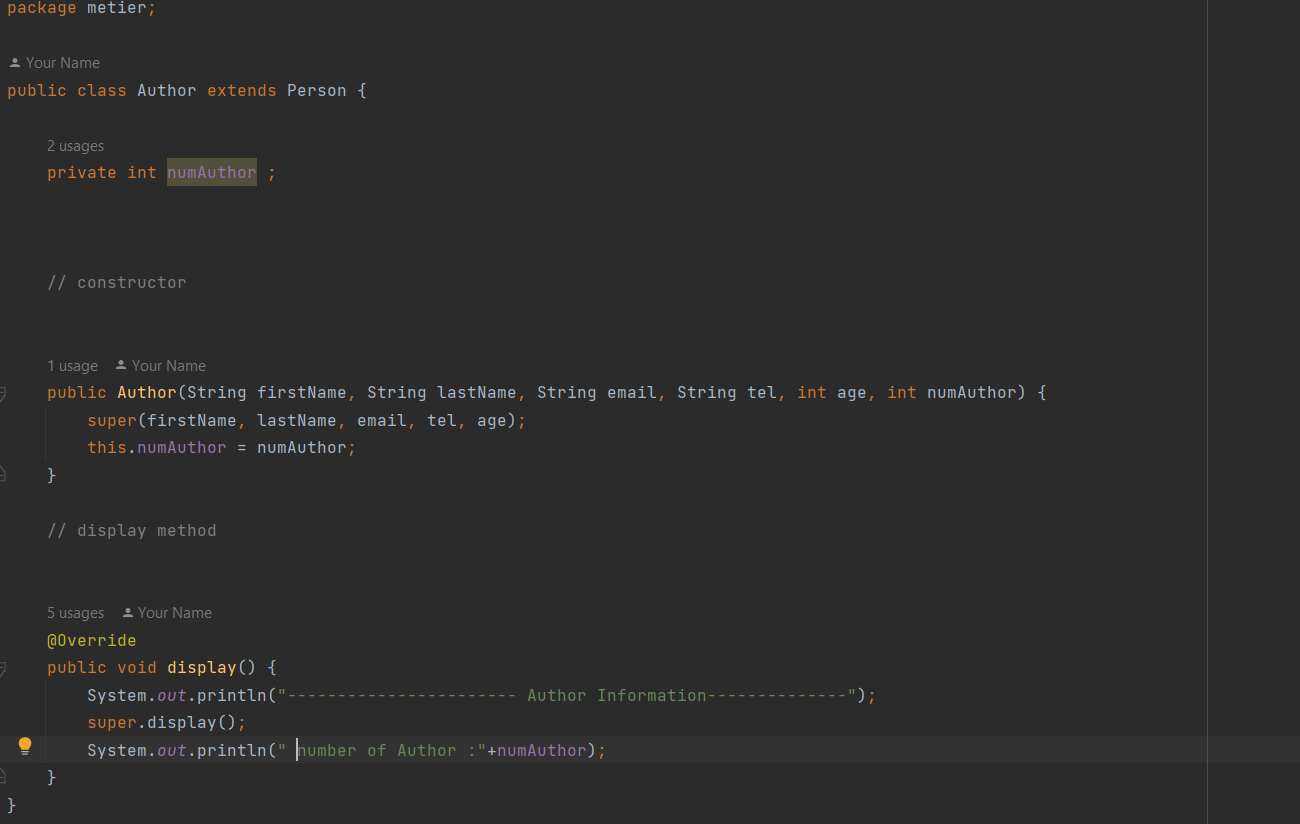
*Text

Description automatically generated*

*Figure 3: Person Class*

*This class represent the parent of Author and Adherent class.*

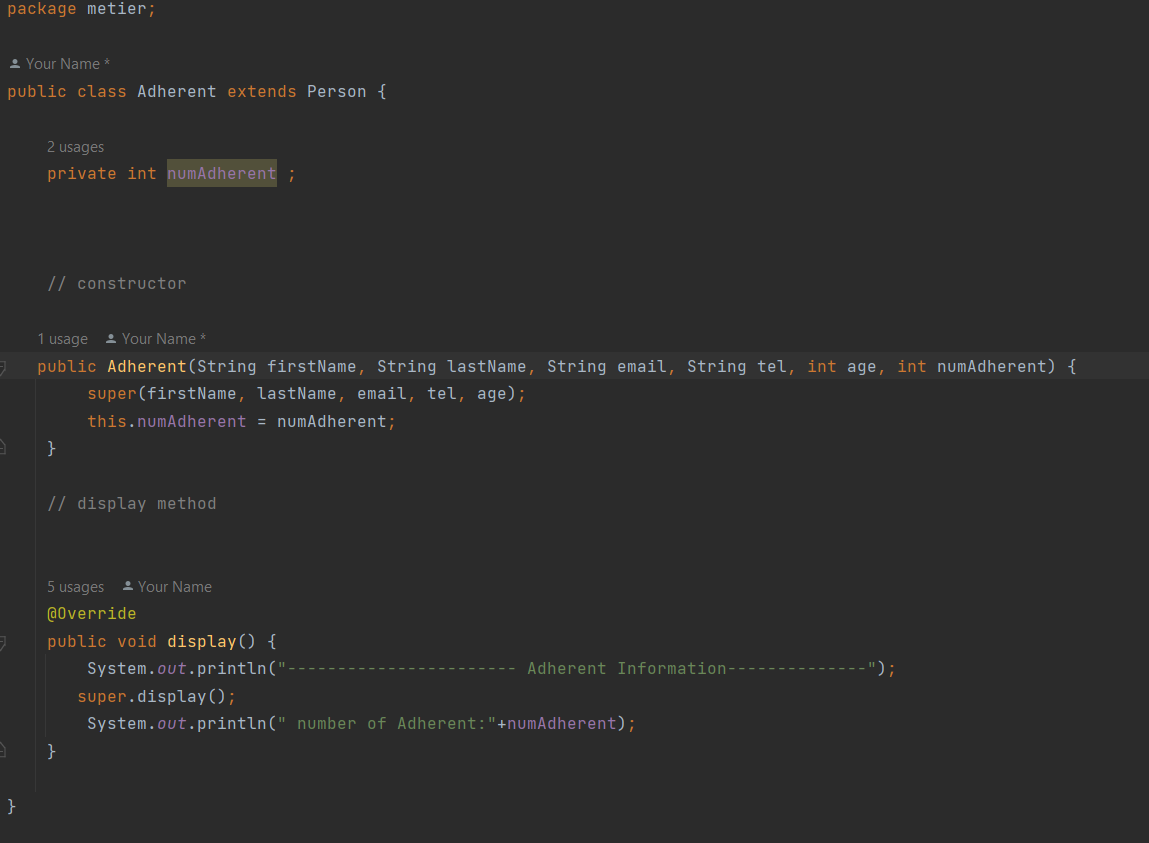
1. ***Author class:***

******

*Figure 4: Author Class*

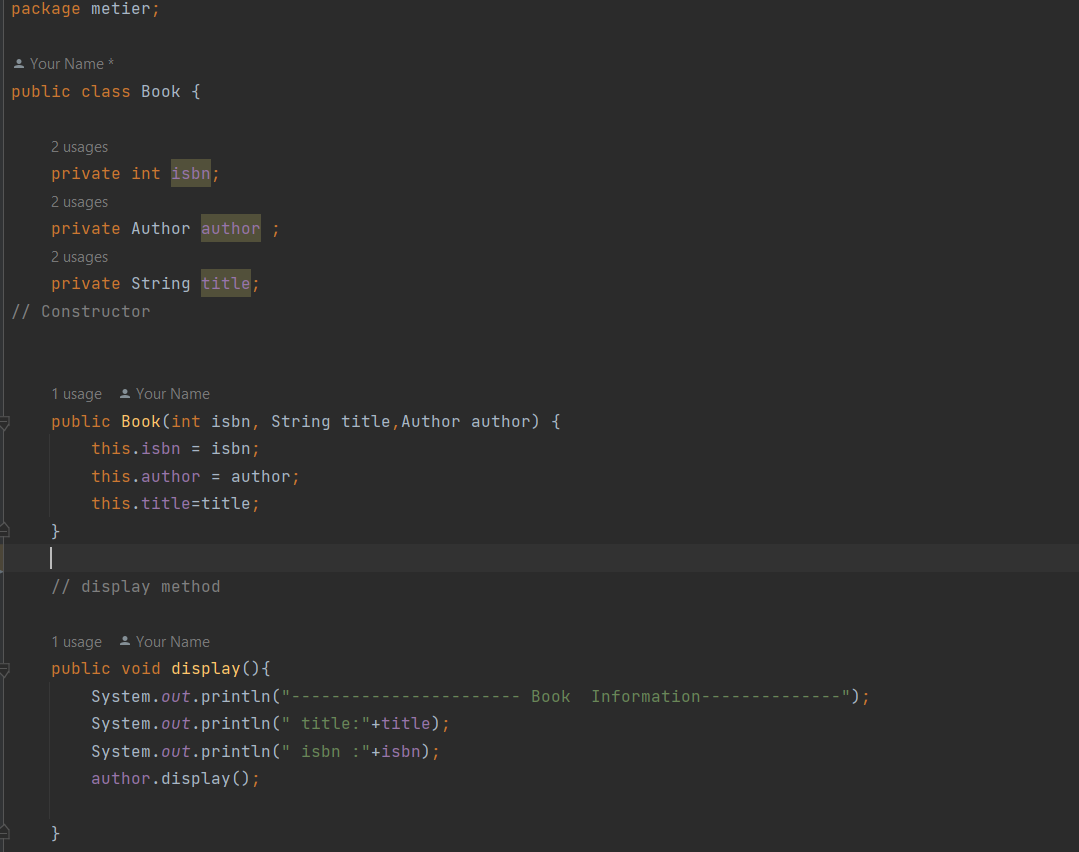
***Extends:*** *indicates that a class is inherited from another class.*

1. ***Adherent class:***

**

*Figure 5: Adherent Class*

*Adherent is a subclass of Person (Adherent inherit all classes and attributes of Person class)*

**

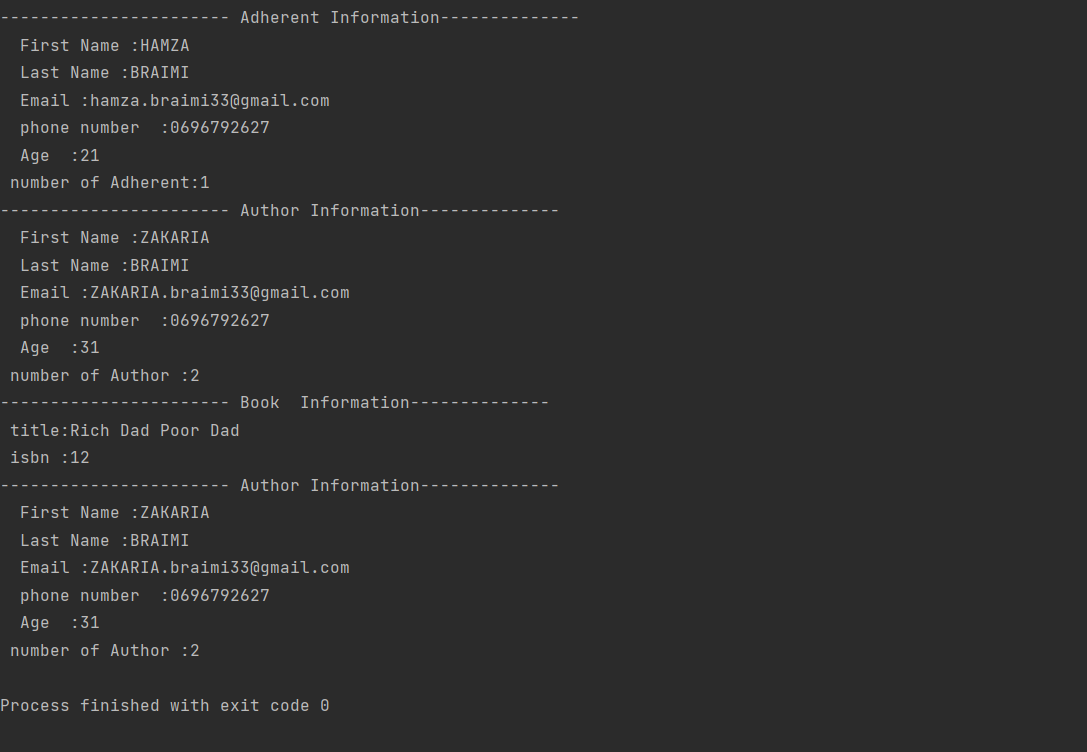
*Figure 5: Book Class*

1. ***Main class:***

*Text

Description automatically generated*

***The Results of compilation:***

******

# Exercise 2:

# Text Description automatically generated

# 

* *In this exercise, we will develop a simple java application that manage books and authors*

*Of a library.*

***Step 1: Conception***

***Diagram

Description automatically generated***

*Figure 1: UML Class Diagram*

*As Shown in the picture above there are three classes. Employee abstract class is the parent of two class (Manager – Engineer), So the connection between Employee class and the other class we call this inheritance.*

***Abstract class:*** *Abstract classes cannot be instantiated, but they can be subclassed.*

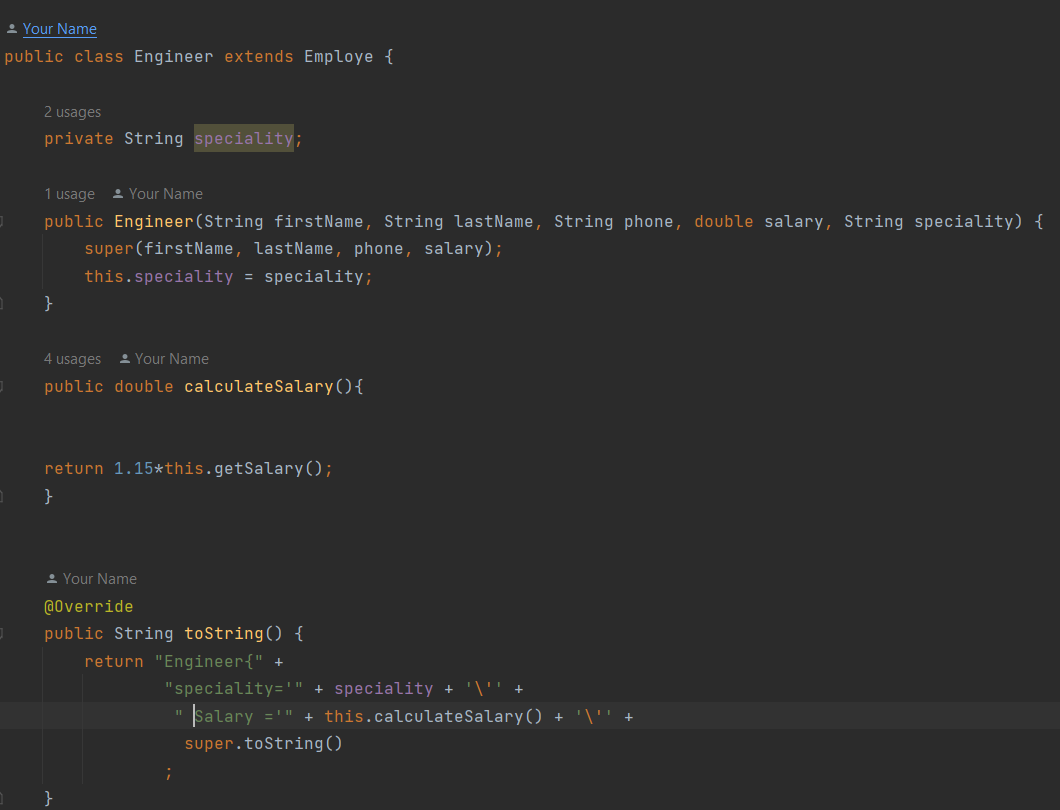
***Step 2: Development***

1. ***Employee class:***

# 

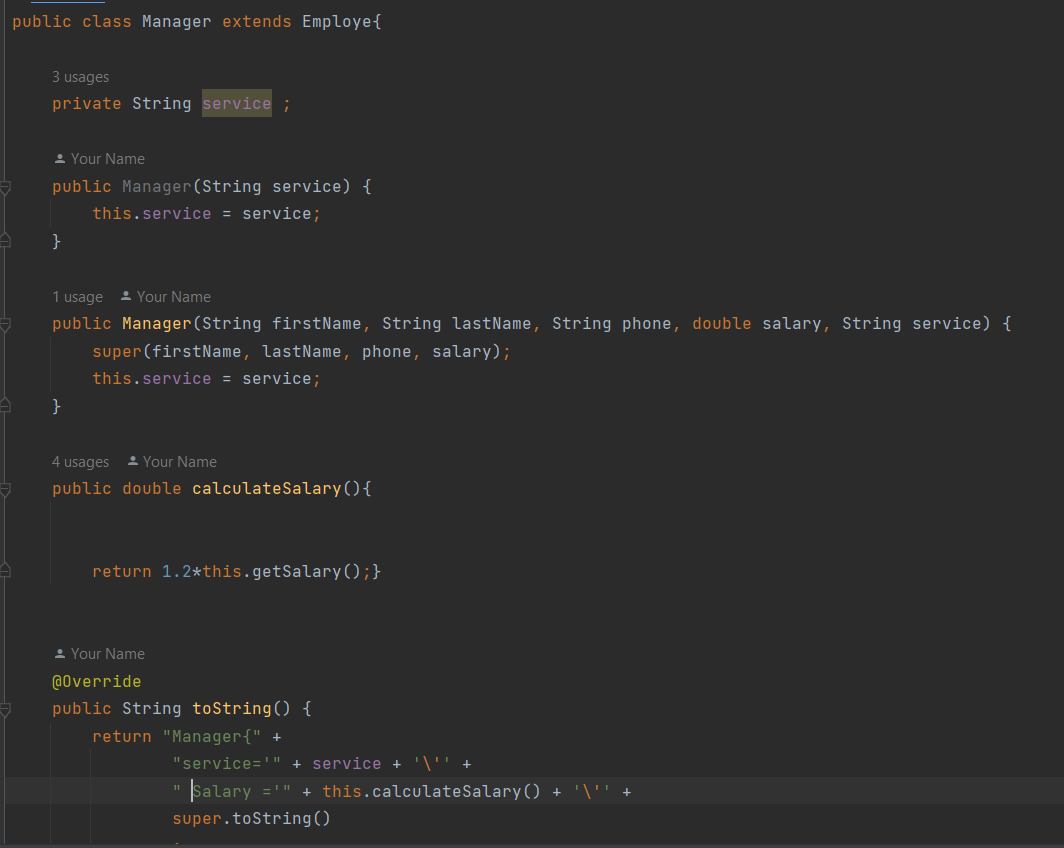
*Figure 2: Employee Class*

1. ***Engineer class:***

******

*Figure 3: Engineer Class*

1. ***Manager class:***

******

*Figure 4: Manager Class*

*Main class of this application:*

*Text

Description automatically generated*

*After RUNINNG THIS APP:*

*Text

Description automatically generated*

# ***Conclusion***