NYIT

**Fall 2021**

Homework No: 04

Title: UML 2 Java

**Name:** Patade, Yash Arun

**Class ID#:**  11

**School ID#:** 1284979

**Course:** Java Networking

**Course ID:** CSCI 725

**Date:** 18/10/2021

Contents

[1) Java 9 installation. 4](#_Toc87718667)

[ UML to Java Mapping. 9](#_Toc87718668)

[Mapping Representation: Notes. 9](#_Toc87718669)

[- Each UML class may be assigned a note with the proposed PSM (Platform specific – Java partial or complete code ideas) 9](#_Toc87718670)

[Visibility for Attributes and Operations. 10](#_Toc87718671)

[- UML note symbol has a tab in the upper right corner. 10](#_Toc87718672)

[Class Scope Attributes and Operations. 11](#_Toc87718673)

[- UML model is Platform Independent Model or PIM and Java program is Platform Specific Model or PSM. 11](#_Toc87718674)

[Example: Utility Class Stereotype. 11](#_Toc87718675)

[- A grouping of global attributes and operations. 11](#_Toc87718676)

[ Designing Relationships - Aggregation (OOA-OOD-OOP) 12](#_Toc87718677)

[Composition – Strong Aggregation. 12](#_Toc87718678)

[- Composition and Aggregation are the form of association between two objects, but there is a subtle difference. 12](#_Toc87718679)

[Example: Composition. 12](#_Toc87718680)

[- Example: Human class is a composition of body parts; Hand, Leg and Heart. 12](#_Toc87718681)

[- When human object dies, all it's body part 12](#_Toc87718682)

[Example: Composition. 13](#_Toc87718683)

[- Another example of Composition is Car and it's part e.g. engines, wheels etc. Individual parts of the car cannot function when a car is destroyed. 13](#_Toc87718684)

[- When you destroy Car Windshield is destroyed too. 13](#_Toc87718685)

[Composition Property. 13](#_Toc87718686)

[- When composed (Strongly aggregated) object moves all of its parts are moving too. 13](#_Toc87718687)

[- Example: Body and organs. 13](#_Toc87718688)

[- Example: Vehicle and parts 13](#_Toc87718689)

[Aggregation. 14](#_Toc87718690)

[- Aggregation or weak aggregation allows including object to exists without a container (without being part of the main object). 14](#_Toc87718691)

[- Aggregation is denoted by an empty diamond, which shows their obvious difference in terms of strength of the relationship. 14](#_Toc87718692)

[Example: Aggregation. 14](#_Toc87718693)

[- Team and Player which is part of a Team, can exist without a team and can become part of other teams as well. 14](#_Toc87718694)

[Example: Aggregation. 15](#_Toc87718695)

[- Student in School class, when School closed, Student still exist and then can join another School or so. 15](#_Toc87718696)

[Relationships. 15](#_Toc87718697)

[- Relationship between classes differ in their strength, where composition represents strongest form of relationship and association being the most general (the weakest) form. 15](#_Toc87718698)

[Example: Aggregation. 16](#_Toc87718699)

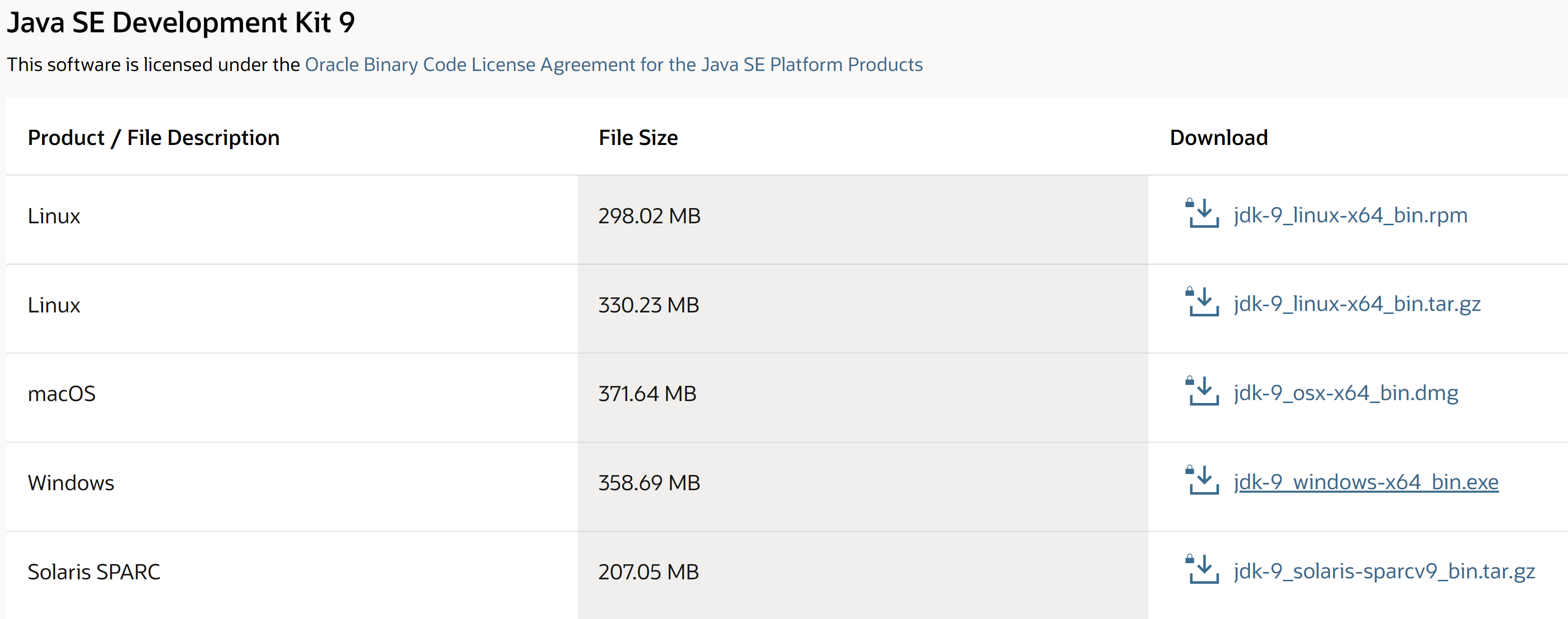
[- Student class and Address (or Apartment) class. 16](#_Toc87718700)

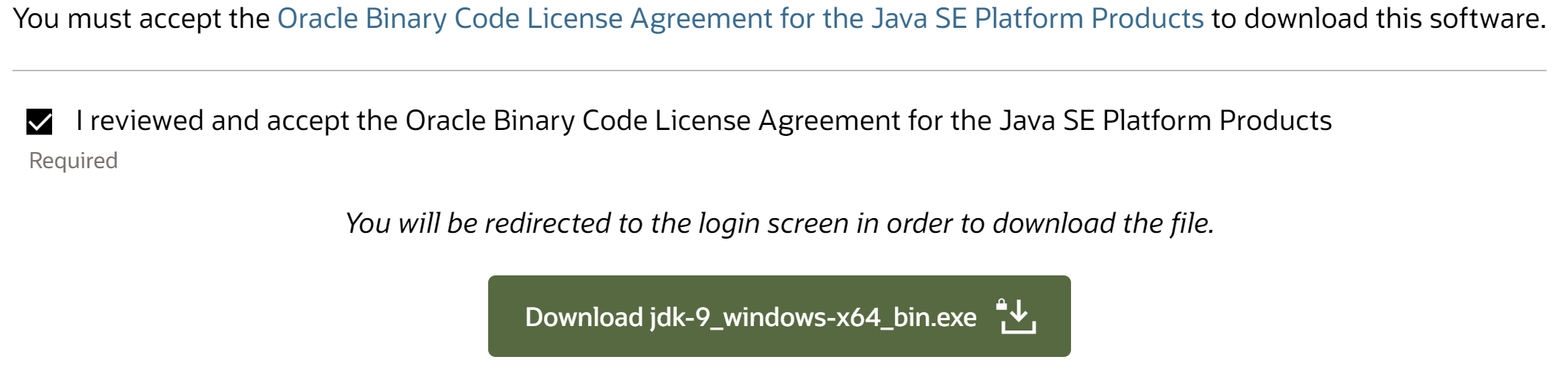
[Nested Classes. 17](#_Toc87718701)

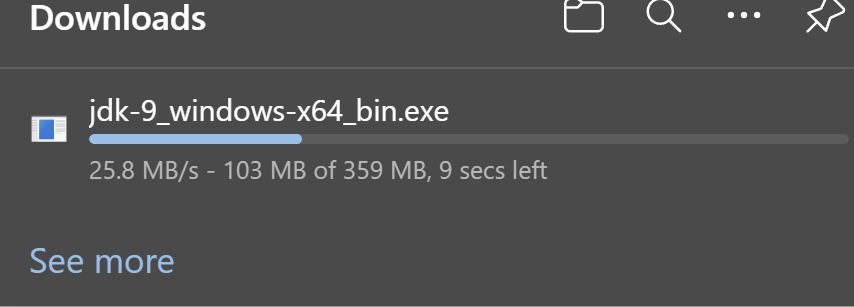
Work from [Microsoft PowerPoint - 725\_04\_Lecture\_00\_JDK9\_Install\_.pptx (tfbor.com)](http://tfbor.com/02_725/04_UML2Java/725_04_Lecture_00_JDK9_Install_.pdf)

# 1) Java 9 installation.

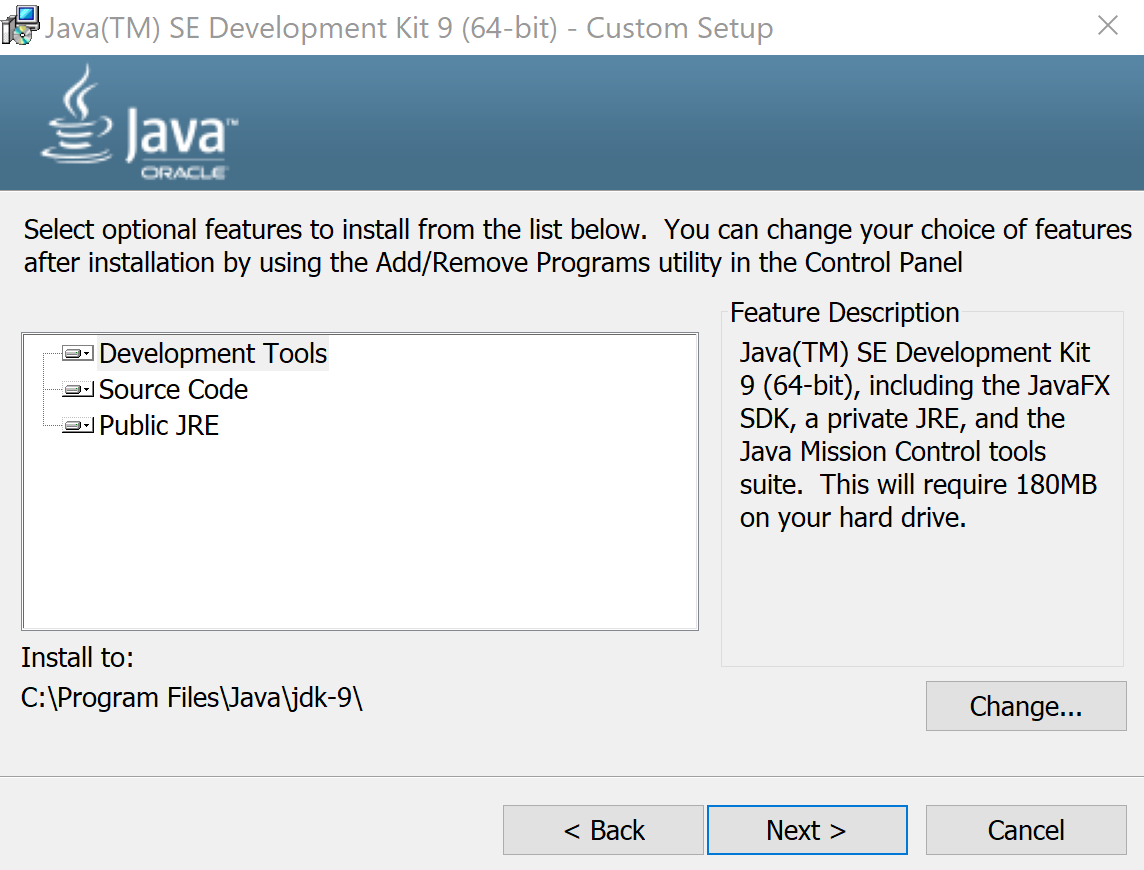


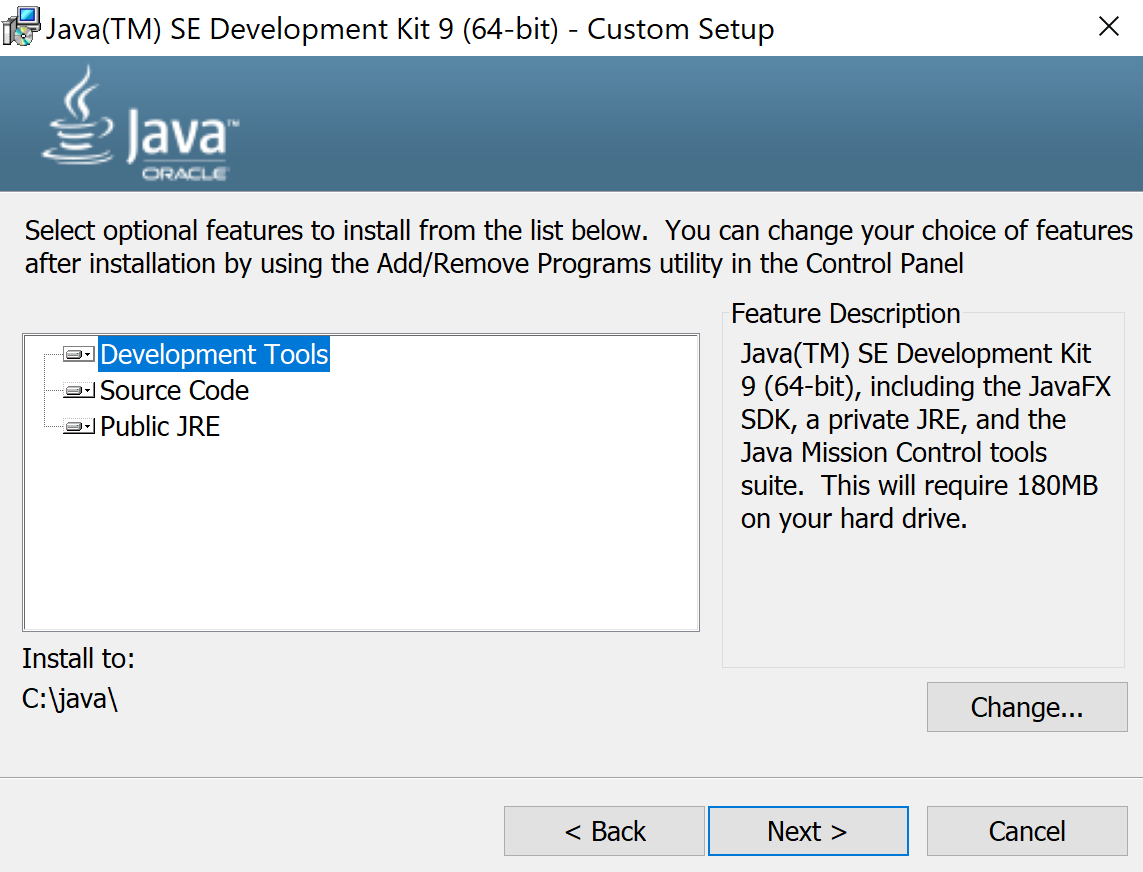


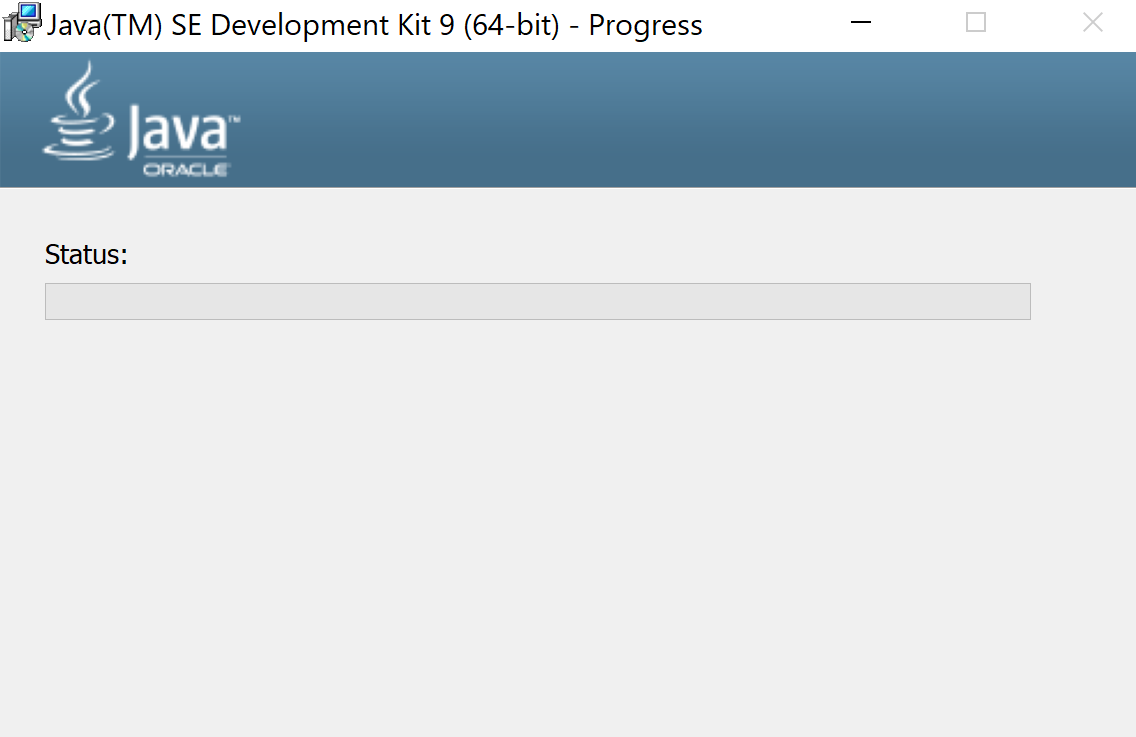




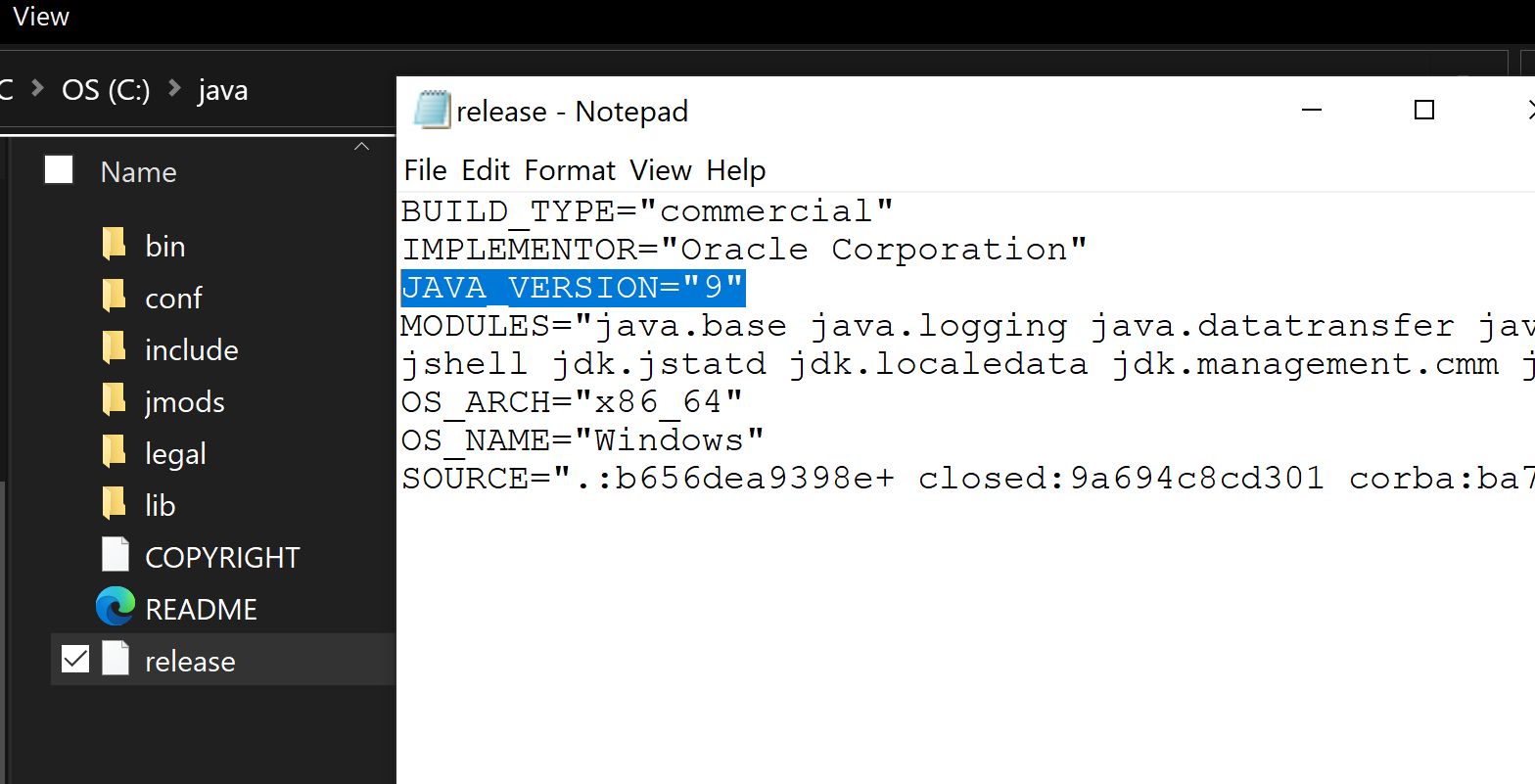


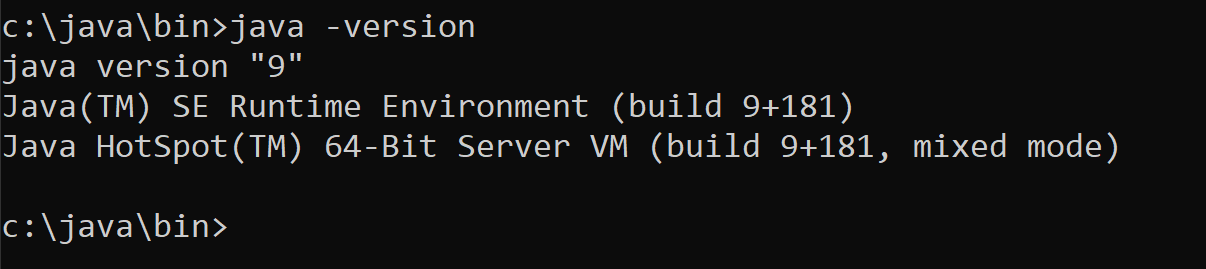










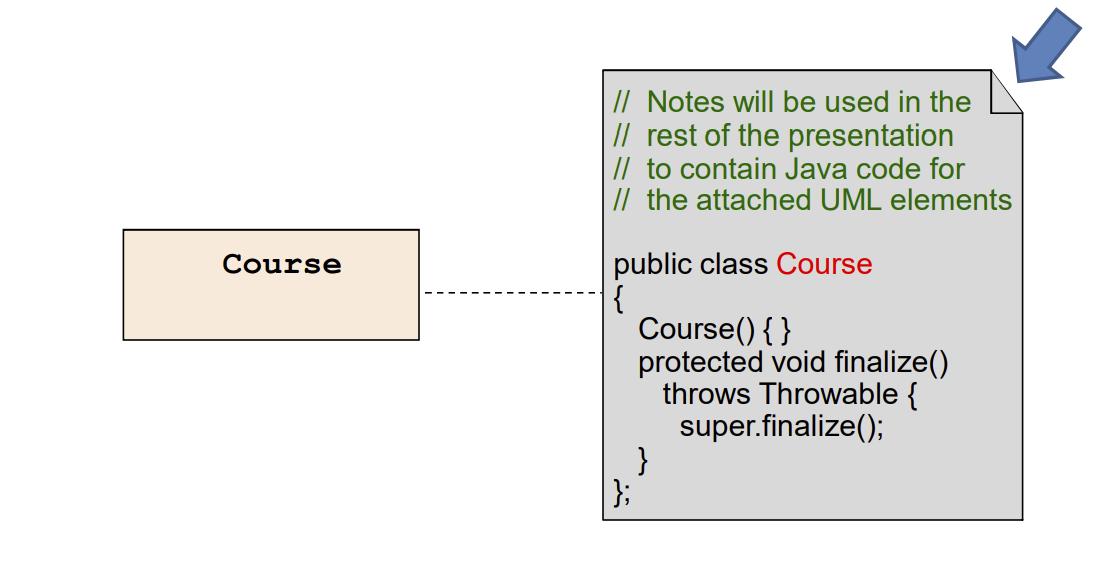


Work from [Microsoft PowerPoint - 725\_04\_Lecture\_01\_UMLtoJava\_v1.pptx (tfbor.com)](http://tfbor.com/02_725/04_UML2Java/725_04_Lecture_01_UMLtoJava_v1.pdf)

# UML to Java Mapping.

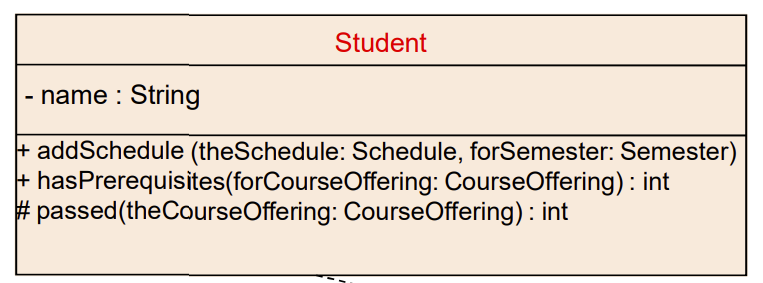
# Mapping Representation: Notes.

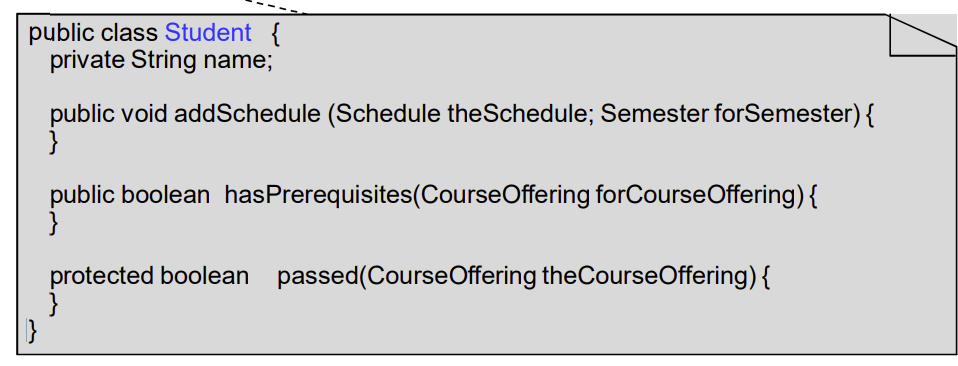
## - Each UML class may be assigned a note with the proposed PSM (Platform specific – Java partial or complete code ideas)



# Visibility for Attributes and Operations.

## - UML note symbol has a tab in the upper right corner.

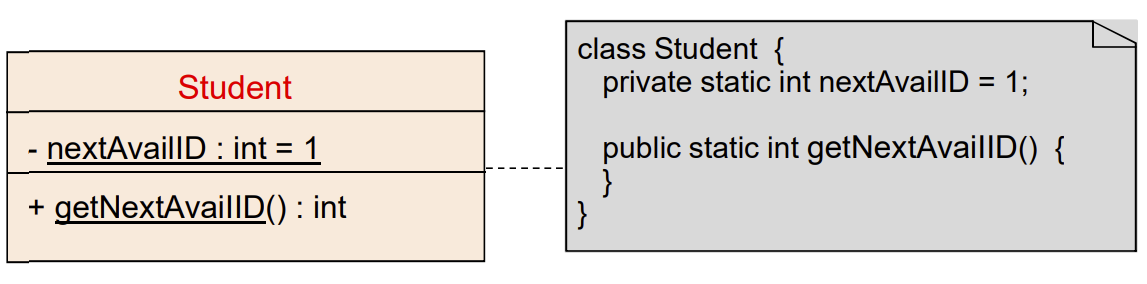




# Class Scope Attributes and Operations.

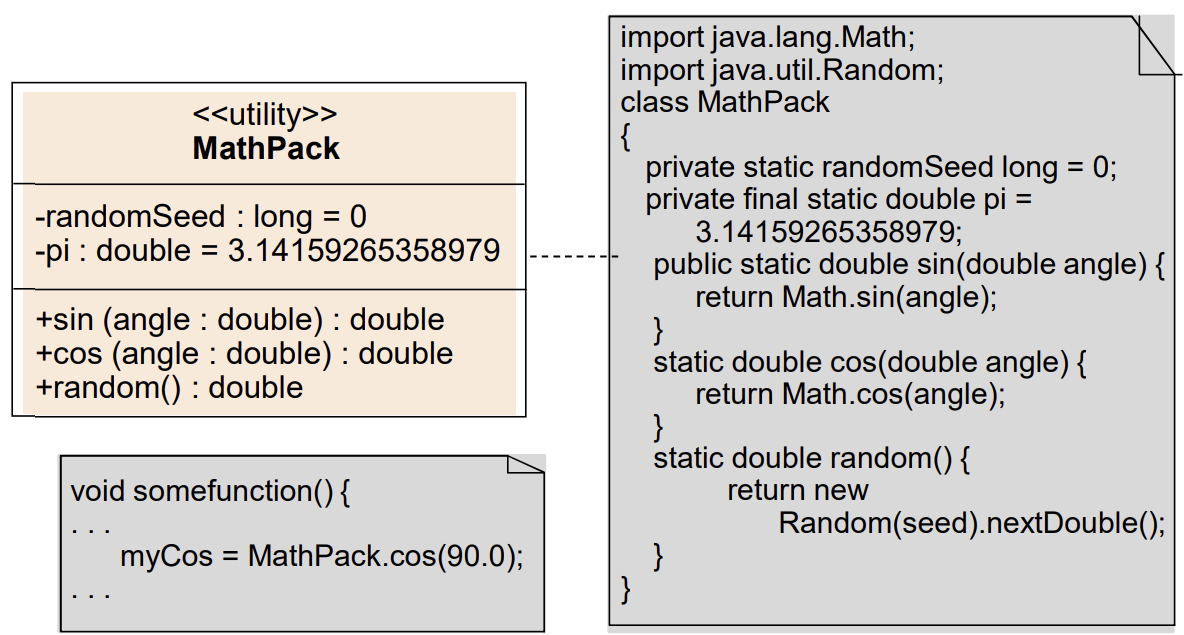
## - UML model is Platform Independent Model or PIM and Java program is Platform Specific Model or PSM.

- Two models are related via transformation or translation mechanism.



# Example: Utility Class Stereotype.

## - A grouping of global attributes and operations.



Work from [Microsoft PowerPoint - 725\_04\_Lecture\_02\_Aggregation\_UML\_Java\_v1.pptx (tfbor.com)](http://tfbor.com/02_725/04_UML2Java/725_04_Lecture_02_Aggregation_UML_Java_.pdf)

# Designing Relationships - Aggregation (OOA-OOD-OOP)

# Composition – Strong Aggregation.

## - Composition and Aggregation are the form of association between two objects, but there is a subtle difference.

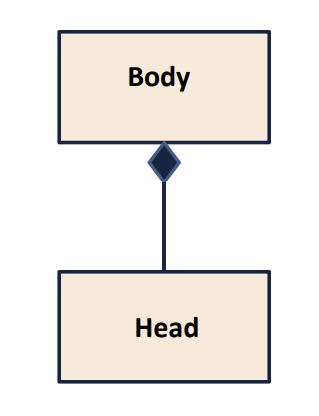
- With Composition, one class owns other class and other class cannot meaningfully exist, when its owner is destroyed.

– In UML notation, a composition is denoted by a filled diamond.

# Example: Composition.

## - Example: Human class is a composition of body parts; Hand, Leg and Heart.

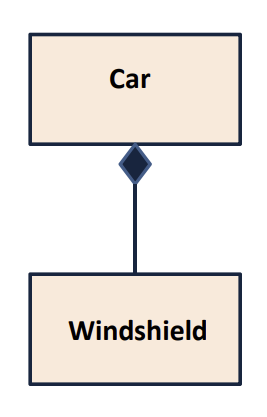
### - When human object dies, all it's body part



# Example: Composition.

## - Another example of Composition is Car and it's part e.g. engines, wheels etc. Individual parts of the car cannot function when a car is destroyed.

### - When you destroy Car Windshield is destroyed too.



# Composition Property.

## - When composed (Strongly aggregated) object moves all of its parts are moving too.

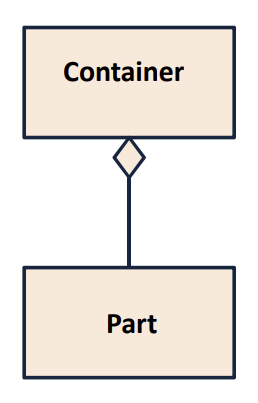
### - Example: Body and organs.

### - Example: Vehicle and parts

# Aggregation.

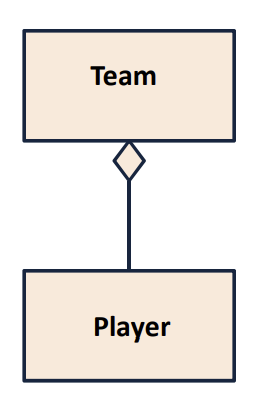
## - Aggregation or weak aggregation allows including object to exists without a container (without being part of the main object).

### - Aggregation is denoted by an empty diamond, which shows their obvious difference in terms of strength of the relationship.



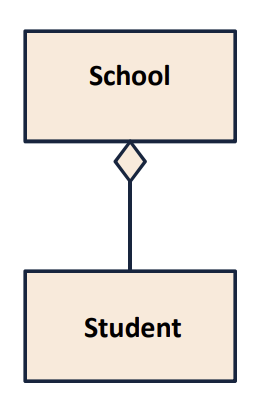
# Example: Aggregation.

## - Team and Player which is part of a Team, can exist without a team and can become part of other teams as well.



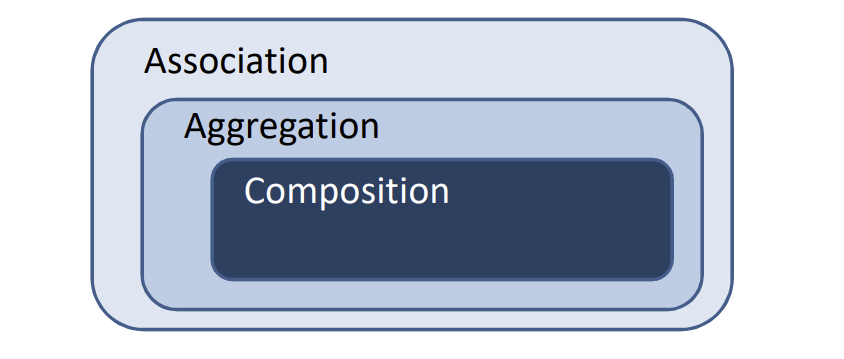
# Example: Aggregation.

## - Student in School class, when School closed, Student still exist and then can join another School or so.



# Relationships.

## - Relationship between classes differ in their strength, where composition represents strongest form of relationship and association being the most general (the weakest) form.



- Composition is stronger than Aggregation.

- Aggregation is a lighter form of Composition, where a sub-part object can meaningfully exist without main objects.

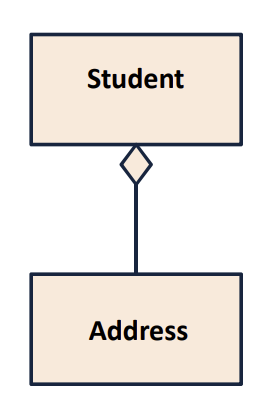
# Example: Aggregation.

## - Student class and Address (or Apartment) class.

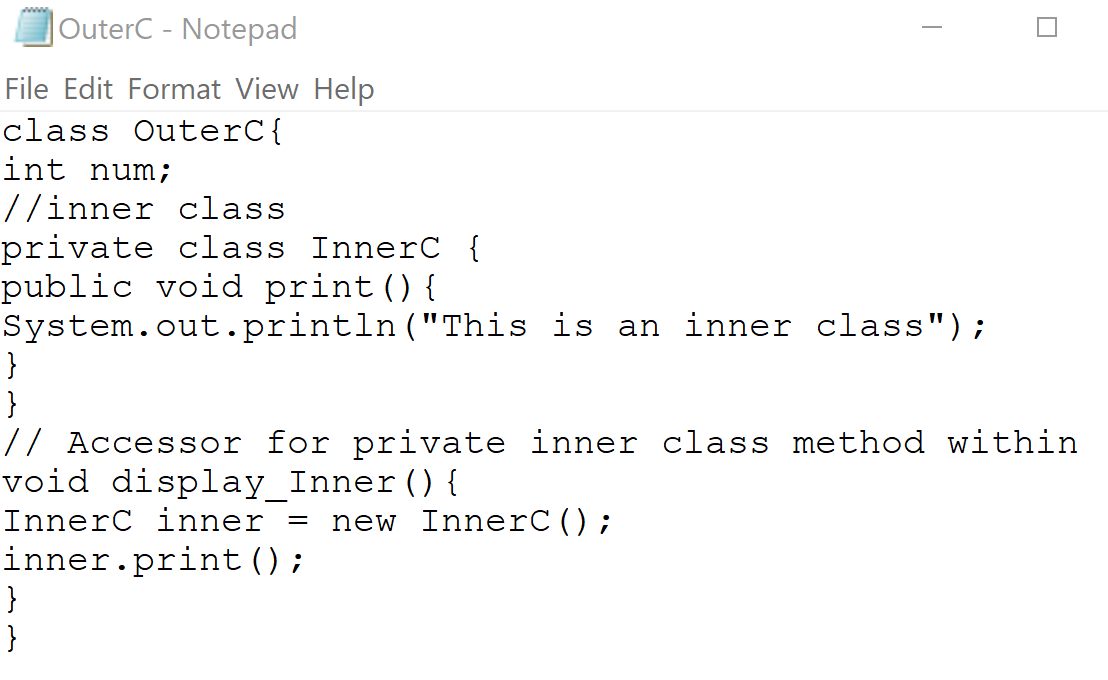
- Every student has an address so the relationship between student and address is a Has-A relationship.

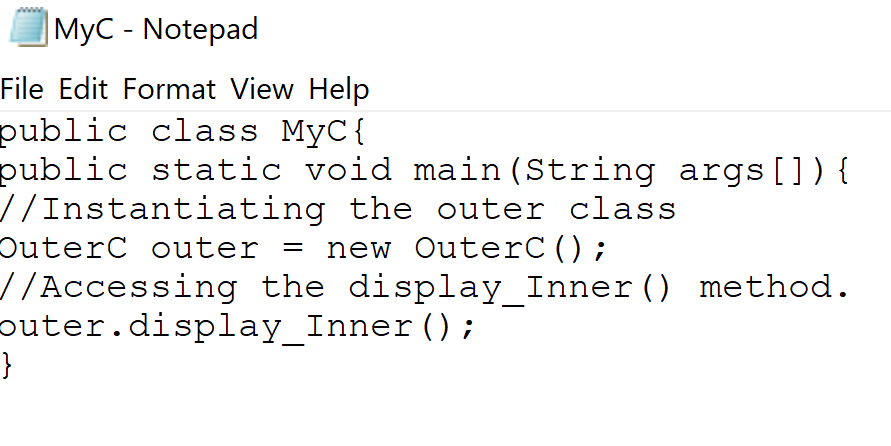
- But if you consider its vice versa then it would not make any sense as an Address doesn’t need to have a Student.

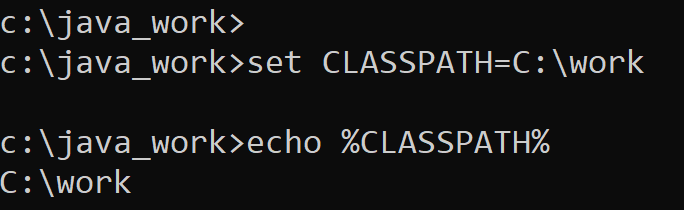
- Association may go both ways.

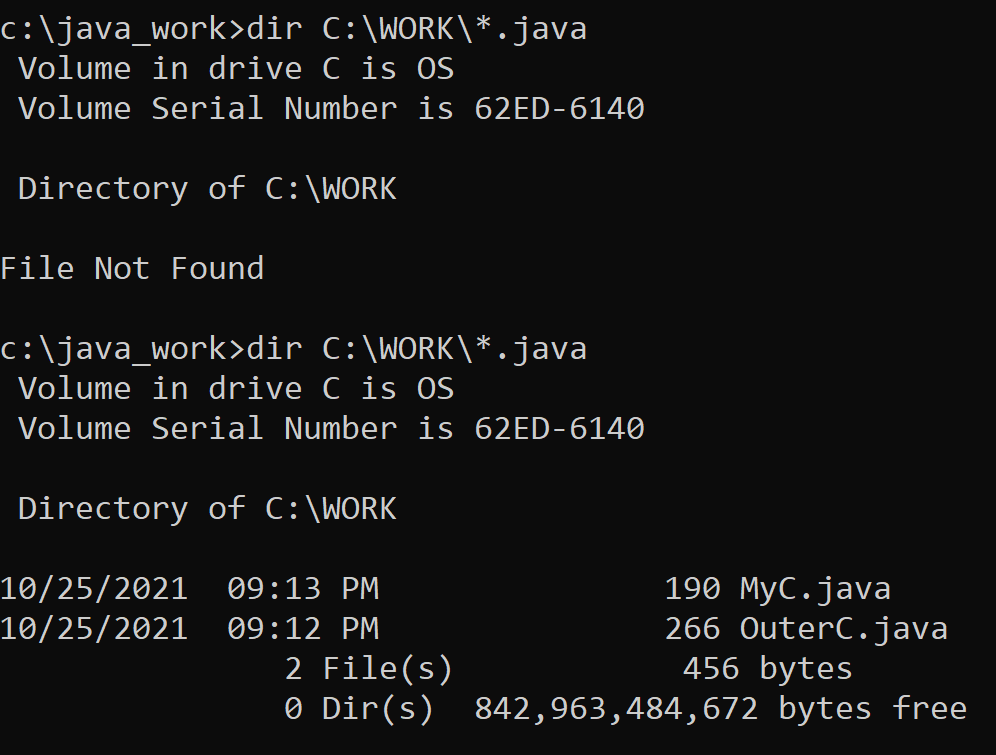


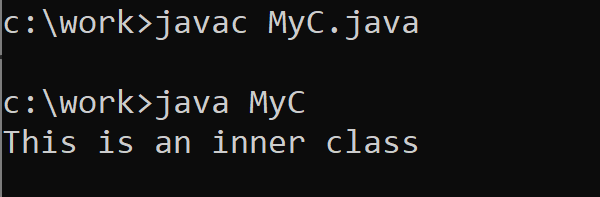
# Nested Classes.











# References.

[1] [Microsoft PowerPoint - 725\_04\_Lecture\_00\_JDK9\_Install\_.pptx (tfbor.com)](http://tfbor.com/02_725/04_UML2Java/725_04_Lecture_00_JDK9_Install_.pdf)

[2] [Microsoft PowerPoint - 725\_04\_Lecture\_01\_UMLtoJava\_v1.pptx (tfbor.com)](http://tfbor.com/02_725/04_UML2Java/725_04_Lecture_01_UMLtoJava_v1.pdf)

[3] [Microsoft PowerPoint - 725\_04\_Lecture\_03\_InnerClass\_a\_Java\_.pptx (tfbor.com)](http://tfbor.com/02_725/04_UML2Java/725_04_Lecture_03_InnerClass_UML_Java_.pdf)