Inheritance and Polymorphism

# Before Class

1. Watch the video on creating derived classes:

<https://youtu.be/Zs342ePFvRI?feature=shared>

1. Familiarise yourself with the concept of inheritance:
   1. <https://www.w3schools.com/java/java_inheritance.asp>
   2. <https://docs.oracle.com/javase/tutorial/java/IandI/subclasses.html>
   3. <https://www.tutorialspoint.com/java/java_inheritance.htm>
2. Answer the questions:
   1. What is class inheritance?

Dziedziczenie klas to mechanizm w programowaniu obiektowym, który pozwala na tworzenie nowych klas (nazywanych podklasami lub klasami pochodnymi) na podstawie istniejących klas (nazywanych klasami nadrzędnymi lub nadklasami). Podklasy dziedziczą atrybuty i metody swoich nadklas, co umożliwia ponowne użycie kodu i wprowadza hierarchię między klasami.

* 1. What are differences between superclass and subclass?

Klasa nadrzędna (nadklasa): Jest to klasa istniejąca przed utworzeniem podklasy. Zawiera ogólne cechy wspólne dla jednej lub więcej podklas.

Klasa podrzędna (podklasa): Jest to klasa utworzona na podstawie istniejącej klasy nadrzędnej. Dziedziczy atrybuty i metody klasy nadrzędnej, ale może również mieć swoje unikalne cechy.

* 1. What is the main purpose of class inheritance?

Głównym celem dziedziczenia klas jest umożliwienie ponownego użycia kodu poprzez przechowywanie wspólnych cech i zachowań w klasach nadrzędnych. To również wspiera koncept hierarchii obiektów, co ułatwia zrozumienie i organizację struktury programu.

* 1. How inheritance is implemented in a programming language?

Mechanizm dziedziczenia w językach programowania pozwala klasie nadrzędnej przekazywać swoje właściwości (pola i metody) do klas podrzędnych. W zależności od języka programowania, dziedziczenie może być jednokrotne (klasa podrzędna dziedziczy od jednej klasy nadrzędnej) lub wielokrotne (klasa podrzędna dziedziczy od więcej niż jednej klasy nadrzędnej). Implementacja może również obejmować specjalne klasy, takie jak interfejsy lub klasy abstrakcyjne, które definiują umowne lub niepełne zachowanie, które musi być zaimplementowane w klasach dziedziczących.

1. Explain how to call a constructor of the parent class.

Super()

1. Explain the concept ‘method overriding’.

Method overriding to koncepcja w programowaniu obiektowym, w której klasa podrzędna (subklasa) dostarcza własną implementację metody, która została zdefiniowana w klasie nadrzędnej (superklasa). Przesłanianie metod pozwala na nadpisanie zachowania metody z klasy nadrzędnej w klasie podrzędnej, co umożliwia dostosowanie funkcjonalności do specyficznych wymagań klasy podrzędnej.

Aby przesłonić metodę, należy w klasie podrzędnej zdefiniować metodę o takiej samej sygnaturze (tj. tej samej nazwie, z tymi samymi parametrami i typem zwracanym) jak w klasie nadrzędnej. Właściwa implementacja tej metody zostanie użyta, gdy obiekt klasy podrzędnej wywołuje tę metodę.

1. Watch the video explaining polymorphism – one of the paradigms of the Object Oriented Programming:

<https://youtu.be/Zs342ePFvRI?feature=shared>

1. Familiarise yourself with the concept of polymorphism.
   1. <https://www.w3schools.com/java/java_polymorphism.asp>
   2. <https://www.tutorialspoint.com/java/java_polymorphism.htm>

# During Class

1. Explain the concepts:
   1. Class inheritance
   2. Superclass and subclass
   3. Method overriding
   4. Polymorphism
2. Create a UML class diagram containing the classes: Book, Ebook, Audiobook, and Library. For each class, specify the fields and methods (see the following tasks in this section). Add relationships between class objects in the diagram. Finally, add the Main class to the diagram along with the main() method containing the program's starting point.
3. Create a Book class that describes books. Add book attributes: title and author. Create a constructor where you assign book attribute values. Add accessor and mutator methods for all book attributes. Add also a display() method to display information about the book. Then, write a program that creates and displays two books.
4. Add an Ebook class definition to your project that derives from the Book class (inherits the Book class attributes and methods). Add an attribute: file name. Create a constructor where you assign the ebook attribute values (title, author and file name). Add accessor and mutator methods for all ebook attributes. Also add a display() method to display information about the ebook. Then, write a program that creates and displays one book and two e-books.
5. Add an Audiobook definition to the project that derives from the Book class (inherits the Book class's attributes and methods). Add the following attributes in the class: minutes and seconds, which define the duration of the audiobook. Create a constructor where you assign audiobook attribute values (title, author, and audiobook duration). Add accessor and mutator methods for all the audiobook attributes. Also add a display() method to display information about the audiobook. Then write a program that creates and displays one book and two audiobooks.
6. Add a Library class definition to your project that contains an attribute that stores any books, ebooks, or audiobooks. Apply an array. Consider the correct data type for the array. Then write a program that creates and displays one book, two e-books and two audiobooks. Save all object in the library. Display the contents of the library. Notice which display() object methods are called.

# After Class

1. Add a Publisher class definition to your project that includes the name and city attributes. Add a constructor in the class where you assign initial values for the attributes. Also add accessor and mutator methods for all attributes.
2. Modify the Book class by adding the publisher attribute. Then make the necessary changes to the remaining classes to display the library content along with publisher information.
3. Modify the Book class. Add the book's year of publication attribute, as well as accessor and mutator methods for this attribute. Then modify other project classes and display the library contents.
4. Modify the UML class diagram by supplementing all the changes you made.
5. Define a Writer class in your project that describes the writer by his/her first name, last name and literary genre. Then, in the Book class, use an object of that class to describe the author of the book. Modify other project classes and display the library contents.
6. Modify the UML class diagram by supplementing the diagram with the Writer class and relations between other classes of the project.