Classes And Objects

# Before Class

1. Watch the video Programming paradigms:

<https://youtu.be/Wt4FPjkCNaU>

[***Paradygmat***](https://pl.wikipedia.org/wiki/Paradygmat)***programowania definiuje sposób patrzenia***[***programisty***](https://pl.wikipedia.org/wiki/Programista)***na***[***przepływ sterowania***](https://pl.wikipedia.org/wiki/Przep%C5%82yw_sterowania)***i wykonywanie***[***programu komputerowego***](https://pl.wikipedia.org/wiki/Program_komputerowy)***. Przykładowo, w***[***programowaniu obiektowym***](https://pl.wikipedia.org/wiki/Programowanie_obiektowe)***jest on traktowany jako zbiór współpracujących ze sobą obiektów, podczas gdy w***[***programowaniu funkcyjnym***](https://pl.wikipedia.org/wiki/Programowanie_funkcyjne)***definiujemy, co trzeba wykonać, a nie w jaki sposób.***

Then list the main features of the following programming paradigms (you can search the Internet for some details):

* 1. Imperative programming - ***Programowanie imperatywne –***[***paradygmat programowania***](https://pl.wikipedia.org/wiki/Paradygmat_programowania)***, który opisuje proces wykonywania jako sekwencję instrukcji zmieniających stan***[***programu***](https://pl.wikipedia.org/wiki/Program_komputerowy)***.***
  2. Procedural programming - ***Programowanie proceduralne –***[***paradygmat programowania***](https://pl.wikipedia.org/wiki/Paradygmat_programowania)***zalecający dzielenie kodu na***[***procedury***](https://pl.wikipedia.org/wiki/Podprogram)***, czyli fragmenty wykonujące ściśle określone operacje. Procedury nie powinny korzystać ze***[***zmiennych globalnych***](https://pl.wikipedia.org/wiki/Zmienna_globalna)***(w miarę możliwości), lecz pobierać i przekazywać wszystkie***[***dane***](https://pl.wikipedia.org/wiki/Dane)***(czy też***[***wskaźniki***](https://pl.wikipedia.org/wiki/Wska%C5%BAnik_(typ_danych))***do nich) jako parametry wywołania.***
  3. Structured programming - ***Programowanie strukturalne –***[***paradygmat programowania***](https://pl.wikipedia.org/wiki/Paradygmat_programowania)***opierający się na podziale***[***kodu źródłowego***](https://pl.wikipedia.org/wiki/Kod_%C5%BAr%C3%B3d%C5%82owy)[***programu***](https://pl.wikipedia.org/wiki/Program_komputerowy)***na procedury i hierarchicznie ułożone bloki z wykorzystaniem struktur kontrolnych w postaci instrukcji wyboru i pętli. Rozwijał się w opozycji do programowania wykorzystującego proste instrukcje warunkowe i***[***skoki***](https://pl.wikipedia.org/wiki/Instrukcja_skoku)***. Programowanie strukturalne zwiększa czytelność i ułatwia analizę programów, co stanowi znaczącą poprawę w stosunku do trudnego w utrzymaniu „***[***spaghetti code***](https://pl.wikipedia.org/wiki/Spaghetti_code)***” często wynikającego z użycia instrukcji [goto](https://pl.wikipedia.org/wiki/Instrukcja_skoku" \o "Instrukcja skoku). W programowaniu strukturalnym unika się skoków bezwarunkowych (takich jak goto), co ma na celu zwiększenie czytelności i przewidywalności kodu.***

1. Watch the video ‘Fundamental Concepts of Object Oriented Programming’

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

***Programowanie obiektowe (***[***ang.***](https://pl.wikipedia.org/wiki/J%C4%99zyk_angielski)***object-oriented programming, OOP) –***[***paradygmat programowania***](https://pl.wikipedia.org/wiki/Paradygmat_programowania)***, w którym programy definiuje się za pomocą***[***obiektów***](https://pl.wikipedia.org/wiki/Obiekt_(programowanie_obiektowe))***– elementów łączących stan (czyli***[***dane***](https://pl.wikipedia.org/wiki/Dane)***, nazywane najczęściej***[***atrybutami***](https://pl.wikipedia.org/wiki/Atrybut_(programowanie))***) i zachowanie (czyli procedury, tu:***[***metody***](https://pl.wikipedia.org/wiki/Metoda_(programowanie_obiektowe))***). Obiektowy program komputerowy wyrażony jest jako zbiór takich obiektów, komunikujących się pomiędzy sobą w celu wykonywania zadań. Podejście to różni się od tradycyjnego***[***programowania proceduralnego***](https://pl.wikipedia.org/wiki/Programowanie_proceduralne)***, gdzie dane i***[***procedury***](https://pl.wikipedia.org/wiki/Podprogram)***nie są ze sobą bezpośrednio związane. Programowanie obiektowe ma ułatwić pisanie, konserwację i wielokrotne użycie programów lub ich fragmentów.***

***Klasy w programowaniu są jak szablony lub opisy określonego typu rzeczy. One mówią nam, jakie informacje można trzymać oraz jakie operacje można wykonywać na tych informacjach.***

***Obiekty to konkretna reprezentacja tych klas. Możemy myśleć o nich jak o rzeczywistych obiektach lub instancjach tych szablonów. Na przykład, jeśli mamy klasę "Samochód", to obiektem tej klasy może być konkretny samochód, jak na przykład Ford Focus.***

***Klasy mogą zawierać zarówno dane (czyli informacje, np. marka samochodu) jak i procedury (znane jako metody), które mówią nam, co możemy zrobić z tymi danymi (np. uruchomić silnik).***

***W skrócie, klasy są jak opisy, co coś może robić i jakie informacje może przechowywać. Obiekty są konkretnymi instancjami tych opisów, reprezentującymi rzeczywiste obiekty lub byty.***

***Założenia:***

***Abstrakcja - W programowaniu obiektowym, abstrakcja oznacza, że skupiamy się na tym, co obiekt robi i jakie ma właściwości, a nie na tym, jak dokładnie jest zaimplementowany. To pozwala programistom korzystać z obiektów i korzystać z ich funkcjonalności bez konieczności analizowania ich wewnętrznych mechanizmów.***

***Enkapsulacja - pozwala na ukrycie wewnętrznych szczegółów obiektu, co sprawia, że jest on bardziej intuicyjny i łatwiejszy do użycia. Dzięki enkapsulacji, można kontrolować sposób, w jaki inne części programu mogą interagować z obiektem, co może pomóc w unikaniu nieprawidłowego lub niebezpiecznego użycia danych.***

***Dziedziczenie (inheritance***) ***w programowaniu oznacza zdolność klasy do przejmowania właściwości i zachowań (danych i metod) z innej klasy, zwaniej klasą nadrzędną lub nadklasą. Dzięki temu pozwala się na ponowne wykorzystanie i rozszerzanie funkcjonalności klasy nadrzędnej. (base class- person, super class-klasa wyżej niż sub, subclass-employee, customer)***

***Polimorfizm - pozwala różnym obiektom zachowywać się inaczej w odpowiedzi na te same komunikaty. Na przykład, jeśli mamy różne typy zwierząt, to każde z nich może wydać dźwięk po wywołaniu metody "wydajDźwięk()", ale dźwięki będą różne w zależności od rodzaju zwierzęcia. Dzięki polimorfizmowi, programy mogą być bardziej elastyczne i bardziej zdolne do obsługi różnych typów danych czy obiektów, co przekłada się na większą uniwersalność i łatwiejsze zarządzanie kodem.***

1. From the Java tutorial:

<https://www.w3schools.com/java/default.asp>

familiarize yourself with all topics, from the first one until the Java Booleans.

1. Check what data types and operators are available in Java.

***Data types are divided into two groups:***

***Primitive data types - includes byte, short, int, long, float, double, boolean and char***

***Non-primitive data types - such as***[***String***](https://www.w3schools.com/java/java_strings.asp)***,***[***Arrays***](https://www.w3schools.com/java/java_arrays.asp)***and***[***Classes***](https://www.w3schools.com/java/java_classes.asp)***(you will learn more about these in a later chapter)***

1. From the course textbook, available in your local repository, read Chapter 1 (Computer Programming) and Chapter 2 (Variables and Operators).
2. Every Java program starts with the main() method. Look in your course textbook for the syntax of the main() method.

***public static void main(String[] args)***

1. Familiarise yourself with the CamelCase – a naming convention for variables, constants, methods and classes in Java. Search the Internet for some details.

***camelCase – system***[***notacji***](https://pl.wikipedia.org/wiki/Notacja)***ciągów tekstowych, w którym kolejne wyrazy pisane są łącznie, rozpoczynając każdy następny wielką literą (prócz pierwszego). Na przykład: foreColor, setConnection, isPaymentPosted. Zapis camelCase różni się od notacji [PascalCase](https://pl.wikipedia.org/wiki/PascalCase" \o "PascalCase), tym że rozpoczyna się małą literą.***

1. From the Chapter 1, complete all exercises in the Exercises section.
   * 1. ***Statement - Part of a program that specifies one step of an algorithm, comment - A part of a program that contains information about the program but has no effect when the program runs.***
     2. ***portable: The ability of a program to run on more than one kind of computer.***
     3. ***compile: To translate a program in a high-level language into a low-level language, all at once, in preparation for later execution.***
     4. ***executable: Another name for object code (The output of the compiler, after translating the program) that is ready to run on specific hardware***
2. Write a program that contains variables with your data: name, surname, age, student id, whether you are a first-year student or not, field of study and the name of the university where you study. The program displays your data. Then, compile and run the program:
   1. In a terminal window (use javac and java commands)
   2. In the VSCode

# During Class

1. Watch the film:
   1. What is object-oriented language

<https://youtu.be/SS-9y0H3Si8>

* 1. Object Oriented Programming

<https://youtu.be/0yEBBCfaIZk>

Then answer the questions:

* 1. What is a class?
  2. What is an object?
  3. What are object components?

1. Specify at least five attributes and three behaviors for objects that represent students.
2. In the Java programming language, define an empty Student class. Pay attention to the class name and the file name in which the class is defined.

public class Student {  
  
}

1. In the Student class, add name and age fields to represent the object's attributes. Pay attention to attribute types.

public class Student {  
 String name;  
 int age;  
}

1. In the VSCode, display the list of object attributes, defined in the Student class (OUTLINE panel).
2. Create a Student class object. To do this, create a separate StudentTest class with the main() method. Assign values to object attributes.

public class Student {  
 String name;  
 int age;  
}

public class StudentTest {   
 public static void main(String[] args) {  
   
 // object creation  
 Student s = new Student();  
   
 // object manipulation  
 s.name = "Peter";  
 s.age = 21;  
 System.out.println(s.name + " " + s.age);  
 }  
}

1. In the Student class, add the following methods that represent object behaviors:
   1. sayHello()
   2. displayName()
   3. displayAge()

Then, add a statement in each of the methods to display the appropriate information.

void sayHello(){  
        System.out.println("Hello from "+name);  
}

1. In the VSCode, display the list of object attributes and methods, defined in the Student class (OUTLINE panel).
2. Create two Student class objects. Assign values to the object fields. Then, call the methods.
3. Add the following fields and methods to the Student class to represent object attributes and behaviors:
   1. Object attributes: (1) student ID card, (2) whether the student ID card is valid or not, (3) semester number, (4) average grade
   2. Object behaviors: (1) display the student’s name, semester number and average grade, (2) change the status of student ID card (valid/invalid), (3) display the student’s name, ID card number and whether the ID card is valid (you can use ?: operator)

Then create two Student class objects, assign them some values and call methods.

1. In the VSCode, display the list of object attributes and methods, defined in the Student class (OUTLINE panel).

# After Class

1. A variable contains your height in cm. Write a program that calculates and displays your height both in cm and in feet and inches. Sample result:

I am 170cm tall, i.e. 5 feet and 7 inches

1. The speed of vehicles on a highway in Poland must be between 40 and 140 km/h. Write a program that checks whether the vehicle speed saved in a variable is correct. Sample result:

Vehicle speed: 158  
Speed is valid: False

1. A bank buys and sells Euro. Write a program that, based on the Euro buying and selling rates saved in variables, calculates the difference between the buying and selling rates (spread). Display result with 4 decimal places (you can use printf() method). Sample result:

Bank buys EUR: 4.5940  
Bank sells EUR: 4.6250  
Spread: 0.0310

1. Variables ‘hours’ and ‘minutes’ contain the number of hours and minutes that have passed since midnight. Write a program that displays the given time (in the format hh:mm – you can use printf() method) and calculates and displays the number of minutes and the number of seconds that have passed since midnight. Sample result:

hours = 14  
minutes = 27  
time: 14:27  
minutes from midnight: …  
seconds from midnight: …

1. Specify at least five attributes and three behaviors for the objects:
   1. Smartphone
   2. Bank account

Then, in separate files, create classes representing above objects. Define attributes and methods. Do not create a main method, just classes with their attributes and empty methods. After that, compile both classes to check whether their syntax is correct. Finally, display list of class attributes and methods in the VSCode (OUTLINE panel).

1. Define a class for describing books. Then, in the class, define fields and methods that represent the attributes and behaviors of book objects. Finally, create two books, assign them attribute values, and call defined methods.
2. Define a class for describing lamps. Then, define the following object attributes and behaviors:
   1. attributes: whether a lamp is on
   2. behaviors: (1) switch on the lamp, (2) switch off the lamp, (3) display information whether the lamp is on or off (use ?: operator)

After that, create two lamps, switch on the first one and switch off the second one. Display information whether the lamps are on or off.

1. Define a class for describing rectangles. Then create two objects that represents rectangles with dimensions of 3x4 and 2x7 respectively. Display their dimensions, perimeters and surface areas.
2. Design a simple counter. The initial value of the counter is 0. The counter can be increased or decreased by one, and increased and decreased by 10. It is possible to reset the counter to its initial value. Then, define a class for counters with five methods. Based on the class definition, create two counters. Finally, using the defined methods, set the value for the first counter to 23 and for the second one to -47. Display results.