Class Structure

# Before Class

1. From the course textbook, available in your local repository, read:
   1. Chapter 4 (Methods and Testing) ok
   2. Chapter 5 (Conditionals and Logic) ok
   3. Chapter 6 (Loops and Strings) ok
   4. Chapter 7 (Arrays and References) yy
   5. Chapter 11 (sections dealing with Constructors and toString() method) yy
2. Watch the video ‘Static vs Non-Static Variables and Methods in Java’:

<https://youtu.be/-Y67pdWHr9Y?feature=shared>

static jak dotyczy całej klasy non-static jak obiektów

1. Explain the concept “method overloading”. Search the Internet for any explanation. Then, watch the video ‘Method overloading in Java’:

<https://youtu.be/km8xnlSRtWg?feature=shared>

Jest to koncepcja w programowaniu obiektowym, która pozwala na zdefiniowanie wielu metod o tej samej nazwie, ale o różnych zestawach parametrów. Celem jest umożliwienie programiście korzystania z tej samej nazwy metody w różnych kontekstach, co sprawia, że kod jest bardziej czytelny i elastyczny.

Method overloading następuje, gdy w obrębie danej klasy istnieje więcej niż jedna metoda o tej samej nazwie, ale różniąca się lista parametrów, tzn. różna liczba parametrów, różne typy parametrów lub obie te rzeczy. W momencie wywołania metody kompilator rozpoznaje, która wersja metody powinna zostać użyta na podstawie ilości i typów przekazanych argumentów.

Muszą być różne parametry!

# During Class

## Static class members

1. Define a class SurfaceArea that contains static methods to calculate the surface area for: circle, rectangle and triangle. Then, write a program that calculates and displays:
   1. area of a rectangle with sides 4 and 5
   2. area of a circle with a radius of 3
   3. area of a triangle with base 3 and height 4

public class SurfaceArea {  
 static float rectangle(float a, float b){  
 return a\*b;  
 }  
}

public class SurfaceAreaTest {  
 public static void main(String[] args) {  
 System.out.print("Rectangle area (5,2) is ");  
 System.out.println(rectangle(5,2));  
 }  
}

1. The class Math contains numerous static methods for performing basic numeric operations such as the elementary exponential, logarithm, square root, and trigonometric functions. You can use the methods in your programs. Search the Internet for a list of methods that this class contains. Then, using the Math class, write a program that calculates and displays:
   1. The larger of two numbers: 34 and 49
   2. PI number
   3. Absolute value of -17
   4. Square root of the expression: nine divided by two
   5. Random number in the range <0.0,1.0)
   6. Integer random number in the range <0,10>
   7. Sine of 90 degrees
2. Define a class MyCalendar with three static integer fields: year, month and day. Then, create:
   1. Method myDate() that returns the date as a string, in the format yyyy-mm-dd
   2. Method days() that returns the number of days passed from the beginning of year.
   3. Method monthName() the returns a string with the month name

Finally, display calculated values. Sample result.

Date: 2023-04-09  
Days from the beginning of year: …  
Month name: April

public class MyCalendar {  
    static int year = 2023;  
    static int month = 4;  
    static int day = 9;  
  
    /\*  
     \* Returns date  
     \* in the form yyyy-mm-dd  
     \*/  
    static String myDate(){  
        return "";  
    }  
  
    /\*  
     \* Returns the number of days  
     \* from the beginning of year  
     \*/  
  
    /\*  
     \* Returns month name  
     \*/  
}

1. Define a class CinemaTicket to describe cinema tickets. The ticket attributes are: cinema name, film title, row, seat, and price. Since the name of cinema is the same for all tickets, use a static field. Then, define a method to display a ticket. Use toString() method. Finally, try to create two tickets and display ticket details.

public class CinemaTicket {  
 static String cinemaName = "...";  
}

## Constructor

1. Modify the class describing cinema tickets. Add a constructor that creates a ticket for the specified movie, with row and seat numbers. Assume that in the first two rows the ticket price is PLN 10, and in the remaining rows, PLN 25. Then write a program that, in the Morning Star Cinema, creates two tickets for the movie “Gladiator”, the first one is in row 2 and the second one in row 7. Print the created tickets.
2. Define a class Person for person’s record. The class should contain the attributes: name, weight (kg) and height (cm). Add constructors:
   1. Person(name)
   2. Person(name,weight,height)

Then add methods that represents the object's behaviors:

* 1. setWeightAndHeight(weight,height)
  2. calculateBMI() – calculates Body Mass Index
  3. displayRecord() – displays full info (name, weight, height, and BMI)

The correct BMI value is 18.5 to 24.9. When displaying a person’s record, if the BMI is too low, display the message: "BMI too low". When the index is too high, display the message: "BMI too high". Write a program, create three persons and display their records.

# After Class

1. In the class SurfaceArea, add methods to calculate the surface area of the following solids: cuboid, sphere, cone. In the VSCode, display a list with all defined methods (OUTLINE panel). Then, complete the program to calculate and display the surface area of solids with sample dimensions.
2. Create a library of methods for converting temperatures between Celsius, Kelvin, and Fahrenheit. You can name the methods e.g. CelsiusToKelvin(), KelvinToCelsius(), ect. Then create a program that calculates and displays the temperature:
   1. 25 degrees Celsius in Kelvin and Fahrenheit
   2. 100 degrees Fahrenheit in Kelvin and Celsius
   3. 0 degrees Kelvin in Celsius and Fahrenheit
3. Create a library of methods for the calculation of basic statistics:
   1. Number of items within the specified range <x,y>
   2. Sum of numbers in the given range <x,y>
   3. Arithmetic mean of the numbers in the given range <x,y>

Then create a program that calculates and displays the basic statistics for integers in the range <5,10>

1. Define a MyArray class with some static methods:
   1. even(int[] array) that returns the number of even values in the array
   2. positiveOdd(int[] array) that returns the number of positive odd numbers in the array
   3. reverse(int[] array) that returns the array in reverse order
   4. compare(int[] array1, int[] array2) that returns ‘ture’ if both arrays are identical or ‘false’ otherwise
   5. different(int[] array1, int[] array2) that returns numbers from the first array that do not appear in the second array
   6. exist(int number, int[] array) that returns ‘true’ if the number is present in the array or ‘false’ otherwise
   7. secondMax(int[] array) that returns the second largest number in the array
   8. lastColumn(int[][] array) that returns the sum of values in the last column
   9. swap(int[][] array) that returns the array with the first and last column swapped
   10. two2one(int[][] array) that returns the 2D array converted to 1D array

Then, write a program in which use all defined methods. Display results.

1. Define a class Alphabet that contains a static method isAlphabet(String t) that returns true if the letters in the text t are in alphabetical order or false otherwise. Sample result:

Alphabet.isAlphabet("abegsw") returns true  
Alphabet.isAlphabet("abcmhsw") returns false

1. There are many devices that can use the Internet: a personal computer, laptop, tablet, smartphone, internet radio or TV. Define an InternetDevice class to describe such devices. The class should contain attributes: name (device name), connected (whether a device is connected to the Internet), and connectedDevices (the number of connected devices - static field). Add a constructor InternetDevice(name) and methods for device manipulation: connect(), disconnect(), isConnected(), displayStatus(), displayConnections() (static method). Then write a program that creates five different internet devices, three of which are connected to the Internet. Display the status of each device and information about the number of devices connected to the Internet.
2. The following definition of the StudentGrades class allows the storage of a student's grades.

public class StudentGrades {  
  
 String studentName;  
 double[] grades;  
  
 StudentGrades(String name, double[] grades) {  
 this.studentName = name;  
 this.grades = grades;  
 }  
}

Complete the class with methods that:

* 1. calculates of the lowest grade
  2. calculates of the highest grade
  3. calculates of the number of grades
  4. calculates of the grade point average
  5. displays student’s record with student’s name, a list of the student's grades, number of grades, lowest grade, highest grade, and grade point average

Then, write a program that creates grades for two students:

* 1. Amanda: 3.5, 4.5, 4.0, 2.0, 5.0, 3.5, 3.5
  2. James: 2.0, 3.0, 2.0, 4.5, 4.5

Tip: pass student's grades to the constructor, as an array.

1. Add the constructor StudentGrades(String name, int numberOfGrades) to the StudentGrade class, which allows you to randomly create the given number of student grades. Tip: use the random number generator, which is available in the Random class. See the manual for more information.
2. Add the constructor StudentGrades(String name) that allows you to enter student’s grades from the keyboard. Tip: use the Scanner class. See the course textbook for more information.
3. Define a class Clock that represents a digital clock. It has two attributes: hour and minute. The hour value ranges from 0 to 23. The minute value ranges from 0 to 59. Next, define the following constructors:
   1. Clock() – initialises clock to 00:00
   2. Clock(hour,minute) – initialises clock to hour:minute

Then, define methods:

* 1. setClock(hour,minute)
  2. setClock() - reset clock to 00:00
  3. displayTime()
  4. addOneMinute()

After that, write a program that:

* 1. creates a clock with starting time 12:47
  2. displays time
  3. sets a clock to 18:14
  4. displays time
  5. sets a clock to 09:03
  6. displays time
  7. sets a clock to 23:58
  8. displays time
  9. adds one minute
  10. displays time
  11. adds one minute
  12. displays time

1. Add an alarm function to the created digital clock. Complete the class definition with the attributes: alarmHour, alarmMinute. Add methods: setAlarm(hour, minute) and runAlarm(), which displays the "beep-beep-beep-beep !!" alarm sound. Turn on the alarm when the alarm time is the same as the clock time(compare times in the addOneMinute() method).
2. Define a class Room that represents a hotel room. The class should contain the attributes: number, beds, occupied, and guestName. Add constructors: Room(number) (creates a room with two beds) and Room(number,beds). Next, define methods for room manipulation: checkin(guestName), checkout(), isOccupied(), and displayStatus(). Then write a program in which create three rooms with two beds, two rooms with three beds and one room with one bed. Store all room objects in an array:

Room[] rooms = new Room[6];

Then follow the steps below:

* 1. Display status of a single room (room number, number of beds, occupied, guest name). Display data in a single row.
  2. Display a report with a list of rooms (room number, number of beds, occupied, guest name)
  3. Display a report with a list of rooms, limited to rooms with the specified number of beds
  4. Display a report on how many rooms are vacant and how many are occupied
  5. Display a report on how many vacant beds are available

Create a separate static method for each report. Pass a list of rooms (an array) to the defined static methods.

1. Modify the Room class. Define toString() method that returns information about the room status. Replace the displayStatus() method with the defined toString() method. Tip: find out in the course textbook how to define and use the toString() method.