Seminar 02
Classes and separate compilation

1. Separate compilation.

- Why do we need it?
  - Separation of the program's logic, readability and future-proofing.

- How does it work?
  - Compiling each .cpp file and linking the object (.o) files.

- Header files. (.h/.hpp)
  - Contain ONLY **declarations** of structs, classes, functions, etc.

- Source files (.cpp)
  - Contain **definitions** of the declared structs, classes, functions, etc.

- Include guards
  - Prevents multiple inclusion of header files *(multiple definitions error)*
  - Standard include guard:

```
#ifndef __HEADER_INCLUDED__
#define __HEADER_INCLUDED__
    // ..code..
#endif
```

  - Non-standard, but widely accepted:

```
#pragma once
    // ..code..
```

2. Classes.

- Why do we need them?
  - Abstraction, reusability, single encapsulated objects with interface.

- Methods and **this** pointer.
  - Methods are functions inside of a class' declaration. They all have access to the **this** pointer.
  - **this** is a pointer referring to the object that's running the method.

- Constructors
  - Methods called when an object of a specific class is being created.
  - Constructors don't have a return type.
  - Default, parameterized and *copy* constructors.
    More on the copy constructors and destructors - in the next lesson.

- Access modifiers.
  - **public:**
    Everything after this modifier is visible by the outside world.
  - ***protected:*** *(more on this modifier when we learn about **inheritance**)*
    Everything after this modifier is visible by the children of the class.
  - **private:**
    Everything after this modifier is NOT visible by the outside world.

- Differences with structs.
  - In **C++** almost none. Structs have public access modifier by default. Classes - private by default.
  - In **C** structs **can't** have methods, static members, access modifiers and more.

3. Examples.

Rectangle.h

```
#pragma once

class Rectangle
{
public:
    Rectangle(double width,
              double height);
    double calcArea();

private:
    double width;
    double height;
};
```

Rectangle.cpp

```
#include "Rectangle.h"

Rectangle::Rectangle(double width
                     double height)
{
    this->width = width;
    this->height = height;
}

double Rectangle::calcArea()
{
    return width * height;
}
```

Source.cpp

```
#include <iostream>
#include "Rectangle.h"

int main()
{
    Rectangle rect(4, 6);          // ⇔ Rectangle rect{4, 6};
    std::cout << rect.calcArea();

    return 0;
}
```