Exercises about classes and inheritance

- Solve them in Visual Studio.

Exercise 13.01

- The namespace of your project is “Persons”.
- You make a class “Person”.
  - 6 variables.
    - name (string).
    - birthDate (DateTime).
    - street (string).
    - houseNumber (string).
    - zipcode (string).
    - city (string).
  - 7 properties.
    - Name (string).
    - BirthDate (DateTime).
    - Street (string).
    - HouseNumber (string).
    - ZipCode (string).
    - City (string).
    - Address (string) (Dynamic property).
      - This consists of more than one variable.
      - street + space + houseNumber + jump to new line + zipcode + space + city.
  - You have to determine by yourself what needs a get and / or a set.
  - One constructor with 6 parameters.
    - name (string).
    - birthDate (DateTime).
    - street (string).

Notes

Notes

....

....

....

....

....

....

....

        - houseNumber (string).
        - zipcode (string).
        - city (string).
    - Method ShowInformation.
        - This shows all the information of a person.
- You make a class “Employee”.
    - This class inherits from Person.
    - You add 2 variables.
        - company (string).
        - wage (double).
    - You add 2 properties.
        - Company (string).
        - Wage (double).
    - You make a constructor with 8 parameters.
        - This constructor inherits from the constructor of Person.
    - Method ShowInformation.
        - This shows all the information of an employee.
        - Don’t repeat code.

	You will later add a property to the class Person. *Method ShowInformation should work without editing it in the class Employee.*

Notes

....

....

....

....

....

....

....

- You make a class “Student”.
  - This class inherits from Person.
  - You add 1 variable.
    - school (string).
  - You add 1 property.
    - School (string).
  - You make a constructor with 7 parameters.
    - This constructor inherits from the constructor of Person.
  - Method ShowInformation.
    - This shows all the information of a student.
    - Don’t repeat code.

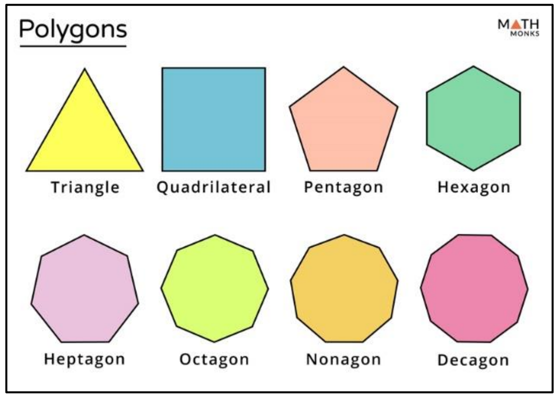
	You will later add a property to the class Person. *Method ShowInformation should work without editing it in the class Employee.*

- You make a testprogram to test if all works fine.

	For testing, you will add a property to the class Person. You can invent one. *This property will be added in the ShowInformation.* *The method ShowInformation should work without editing it in the class Employee and Student.* *If this works, than change the constructor of a Person, by adding the new property.* *What must change in Employee and Student?*

Exercise 13.02

- The namespace of your project is “Mathematics”.



- You make a class “Polygon”.
  - One variable “color”
  - One property “Color”.
  - One constructor that sets the color.
- You make a class “Rectangle”.
  - Inherits from Polygon.
  - Two extra variables / properties.
    - Width.
    - Height.
  - One constructor to create a rectangle with a Width and a Height, with a certain color.
  - Two methods.
    - Area (and its functionality to calculate the area).
      - Width * Height.
    - Circumference (and its functionality to calculate the circumference).
      - 2 * Width + 2 * Height.

Notes

....
....
....
....
....
....
....

Notes

- You make a class "Square".
  - Inherits from Rectangle.
    - Attention: A square is a special rectangle.
  - One constructor to create a Square with only one size (Width) and a certain color.
    - This sets the Width and Height for the rectangle.
- You make a class "RightTriangle".
  - Inherits from Polygon.
  - Two extra properties.
    - Base. (one side of the right angle)
    - Height. (other side of the right angle)
  - One constructor to create a triangle with a Base and a Height, with a certain color.
  - Two methods.
    - Area (and its functionality to calculate the area).
      - (Base * Height) / 2.
    - Circumference (and its functionality to calculate the circumference).

Do you have enough information on how to calculate this?

Look it up with internet. Think Pythagoras.

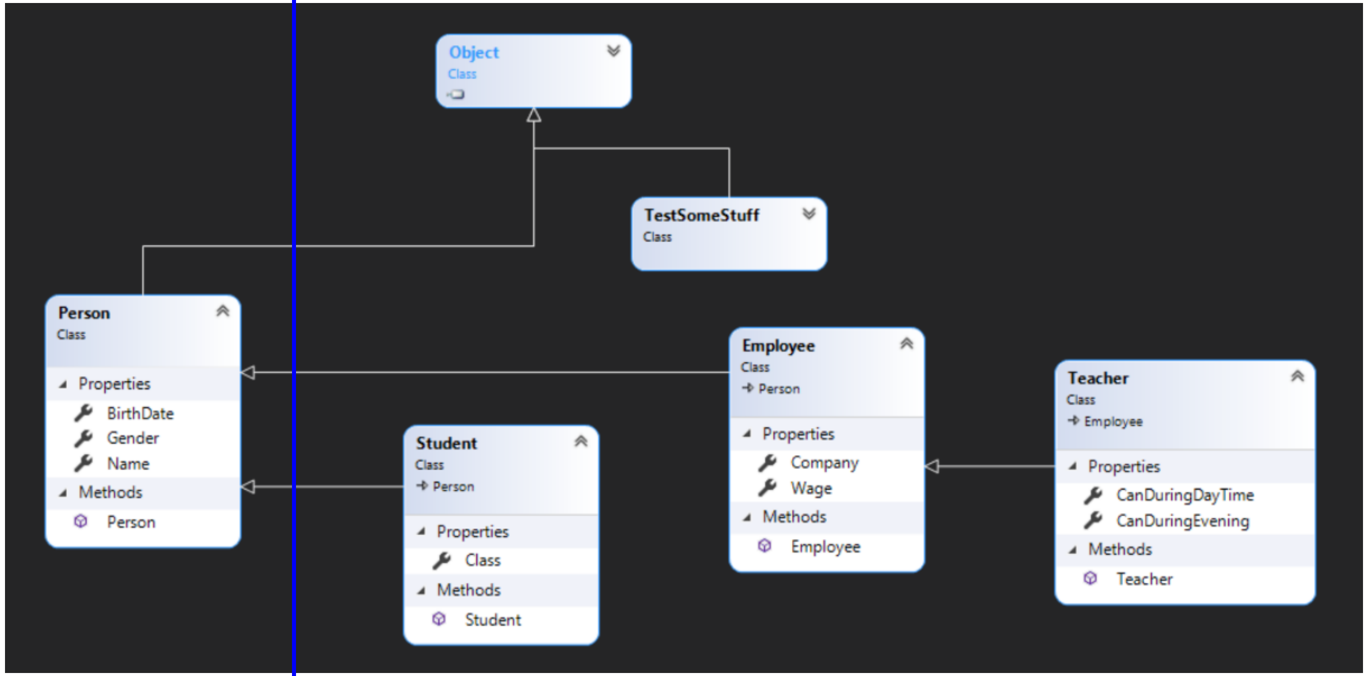
$$a^2 + b^2 = c^2$$

Variant 1

- Can you add a Circle to it?

Exercise 13.03

- Create the classes thru a class diagram.
    - You don't type them by yourself.
- The picture is shown below.
    - Your classes will be generated by the tool.



Notes

Exercise 13.04

- Create a class “Animal”.
  - Do the necessary stuff for making this.
  - It must have a method “Eat”, that shows “Animal is eating”.
- Create a class “Mammal”.
  - Do the necessary stuff for making this.
  - It inherits from Animal.
  - It must have a method “Eat”, that shows “Mammal is eating”.
- Create a class “Cat”.
  - Do the necessary stuff for making this.
  - It inherits from Mammal.
  - It must have a method “Eat”, that shows “Cat is eating mouse”.
- Create a class “Mouse”.
  - Do the necessary stuff for making this.
  - It inherits from Mammal.
  - It must have a method “Eat”, that shows “Mouse is eating cheese”.

Notes

Exercise 13.05

- Create an abstract class “Shape”.
  - It has an empty constructor.
  - It has an abstract property “Name”.
  - It has a method “Area” that returns 0.
  - It has a method “Volume” that returns 0.
- Create a class “Point” that inherits from “Shape”.
  - It has 2 properties X and Y (the coordinates of a point).
  - The property Name returns “Point”.
  - It has a constructor with no parameters, where X and Y coordinates are 0.
  - It has a constructor with 2 parameters (X and Y coordinates).
  - You override the method “ToString” to show information about the point.
    - Show it in this format “[X, Y]”.
- Create a class “Circle” that inherits form “Point”.
  - It has 2 fields. The center and the radius.
  - It has 3 properties “Center” (this is a point), Radius and Name.
    - Radius can’t be negative. A negative value becomes 0.
    - The property Name returns “Circle”.
  - It has 3 constructors.
    - One with no parameters. This is a point on coordinates 0 and 0 with radius 0.
    - One with 3 parameters.
      - X coordinate.
      - Y coordinate.
      - Radius.
    - One with 2 parameters.
      - A point.
      - Radius.
  - You override the method Area that returns the Area of a circle.
  - You create a method that calculates the circumference.

Notes

Notes

- You create a method that calculates the diameter.
- You override the method "ToString" to show information about the circle.
  - You show the center, the radius, the Area and the diameter.

- Create a class "Cylinder" that inherits form "Circle".
  - It has 2 fields. The base and the height.
  - It has 3 properties "Base" (this is a circle), Height and Name.
    - Height can't be negative. A negative value becomes 0.
    - The property Name returns "Cylinder".
  - It has 4 constructors.
    - One with no parameters. This is a circle on coordinates 0 and 0 with radius 0 and the height is 0.
    - One with 4 parameters.
      - X coordinate.
      - Y coordinate.
      - Radius.
      - Height.
    - One with 3 parameters.
      - A point.
      - Radius.
      - Height.
    - One with 2 parameters.
      - A circle.
      - Height.
  - You override the method Area that returns the Area of a cylinder.
  - You create a method that calculates the volume.
  - You create a method that calculates the area.
  - You override the method "ToString" to show information about the cylinder.
    - You show the center, the radius, the Area and the diameter and the volume.

Exercise 13.06

	This is a very difficult exercise. *I must receive 2 things:* - *A document in how to use the classes you have created. This document can contains example code in how to use the classes.* - *The classes itself.*

- You must invent and create an exercise by yourself.
- There must be inheritance in classes at least 3 levels deep.
  - Parent – Child – Child of child.
  - Inheritance must be implemented somewhere.
    - this and base must be used somewhere.
- Your namespace is “StudentTryout”.
- Your code will be treated as a Black Box.
  - Meaning.
    - I will create a testprogram using your classes.
    - I will not look inside the classes to create the test program.
  - I will do this:
    - “using StudentTryout”.
    - And see what is there and I will try to use it.

	Whatever I do, if the program fails it must be failing in my program, because I'm doing something stupid. *It never fails in your code. Everything that goes outside your classes and inside your classes behaves correctly like you have sent me the documentation.*

- After that. I will look inside your code just for trying stuff out, because a lot is possible, and see if I could have broken your code.

Notes

....

....

....

....

....

....

....