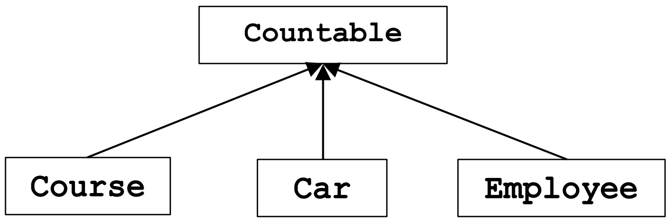
Lesson 5

If you find any open questions in some of the presented exercises then feel free to make your own decisions. However, your decisions should be reasonable and must not contradict any of the conditions explicitly written for the exercise. Please, write comments in the programs that clarify your assumptions/decisions, if any.

Exercise 1

Recall the exercise of dugga 3. In this dugga, you are asked to implement a class named **`Countable`** that represents a class that counts the number of its existing instances. A possible implementation for this class is given in the file **`ex1.cpp`**.

It is easy to see that many hierarchies of classes, like **`Course`**, **`Car`**, **`Employee`**, can be made **`Countable`**. In other words, **`Countable`** can be made the base class of these hierarchies, as the figure below illustrates.



a) Describe what problems the proposed implementation for class **`Countable`** imply taken into account the scenario of the figure above.

b) Modify the class **`Countable`** implementation to address those problems.

Exercise 2

Write a template function, named **`my_copy`**, that copies a given subsection of a container into an array **`V`** of integers. The function should return the number of values copied into **`V`**. For some examples of how to call the function see the main in **`ex2.cpp`**.

Add your definition of the template function to **`ex2.cpp`** and test your code with the given **`main`** function.

Exercise 3

Recall the class **`Set`** of Lesson 2, exercise 1. Add to this class a member function named **`add`** that adds the values in a given subsection of a container (e.g. a vector, a list, or an array) to the set. Then, test your code with the **`main`** given in **`test_set.cpp`**.

Exercise 4

Write a program that reads two text files and displays a list of the words that occur in both files.

Your program should read the words of in the files in two respective sets (**`std::set`**). Then, used the set intersection algorithm (**`std::set_intersection`**) to find the common words and store them in a vector. Finally, display the words in the vector.

Use **`STL`** to solve this exercise. Use standard algorithms, instead of hand-written loops (such as **`for`**-loop, **`while`**-loop, or **`do`**-loop). You program should not have any of these hand-written loops.

Two text files are given to test your program, **`ex4_1.txt`** and **`ex4_2.txt`**.

Lycka till!!