# BDSA Assignment 02

tcla, olfw, hast

September 2022

## Repository

https://github.com/Lignio/assignment-02

## Types

### Class

A class is reference type meaning you can have multiple variables pointing at the same object. They come with a default constructor. Classes can use inheritance. If your objects aren't solely for storing data in properties you want to use class.

### Struct

A struct is value type like primitive data types meaning you make copies when assigning the "same" struct to different variables. If you want to assign multiple values to a variable quickly without taking up too much space you can use struct.

### Record Class

A record class is reference type like class but it comes with overwritten equals() and toString() methods that compares and prints based on the properties and their values. If you want a class who's main purpose is to model a data type with specific properties to maybe compared with other objects of same type you can use a record class.

### Record Struct

A record struct is value type like struct but comes with the option for positional parameters and more fancy functionalities like a toString() method. If you want a struct-like variable but with more functionality, perhaps positional parameters rather than named parameters you can use a record struct.

## Extension methods

Solve the following questions with extension methods (one-liners):

Flatten the numbers in xs:

xs.SelectMany( x ⇒ x);

The SelectMany keyword flattens the input, resulting in an output of x for every x.

Select numbers in ys which are divisible by 7 and greater than 42:

ys.Where(x ⇒ x%7 == 0 && x>42);

The Where keyword checks every value in the input for the following condition(s), in this case if the input is divisible by 7 and if it is greater than 42.

Select numbers in ys which are leap years:

ys.Where (x ⇒ ($x$%400 == 0||($x$%4 == 0&&$x$%100! = 0))&&$x$ > 1582);

Here the conditions are that the input is divisible by 400 or 4 and not 100 and that it is greater than 1582. It is the rules for leapyears which were used in assignment 00.

## Delegates / Anonymous methods

Implement the following anonymous functions using the built-in delegates and lambda expressions: A method which takes a string and prints the content in reverse order (by character):

Action<string> reverse = str ⇒ Console.WriteLine(new string(str.Reverse().ToArray()));

A method which takes two decimals and returns the product:

Func<double, double, double> product = (number1, number2) ⇒ number1 * number2;

A method which takes a whole number and a string and returns true if they are numerically equal. Note that the string ” 0042” should return true if the number is 42:

Func<string, int, bool> isEqual = (str, number) ⇒ int.Parse(str) == number;

# Software Engineering

## Exercise 1

**Describe the difference between a scenario and a use case. Describe for each of the two concepts when and for what they are used.**

A scenario is a way of finding out how people handle different situations. This can include a description of a work-flow and the reasoning behind the specific work-flow. Or a story about how something went wrong and was handled.

A use-case is on the other hand is a systematic way of describing interactions users have with a system. It is a system of mapping out a problem using graphic symbols and text.

## Exercise 2

**Identify and briefly describe four types of requirements that may be defined for a computer-based system.**

One type of requirement is a functional requirement. A functional requirement is a requirement that is a explicit statement of what a system should be able to do or shouldn't do.

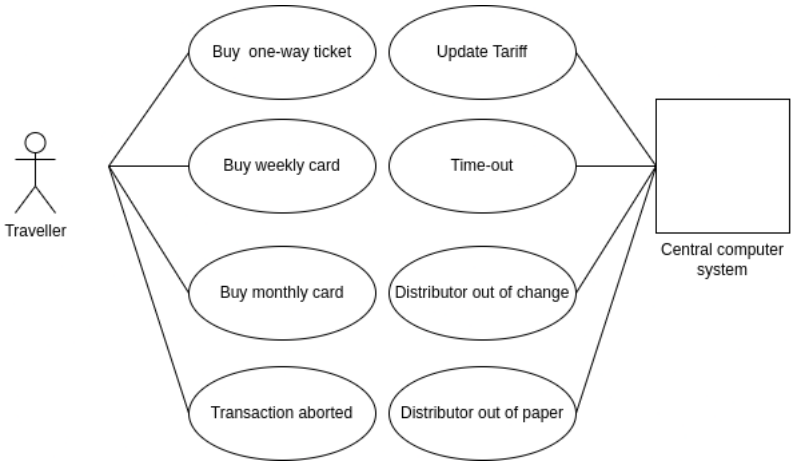
Another type of requirement is a non-functional requirement. A non-functional requirement is a constraint to how the system can be developed. This could be a resource constraint such as a timing constraint or a workforce constraint.

User requirements is yet another type of requirement. A user requirement is a high-level abstract requirement to a system. In short, it is the intended resulting service in terms anyone can understand. A user requirement can also include a drawing/diagram.

A fourth type of requirement is a system requirement, which is a detailed description of what a system will do in order to have the same function as the user requirement.

## Exercise 3

**Draw a use case diagram for a ticket distributor for a train system.**




## Exercise 4

**Discover formulations in the requirement that are ambiguous**

There are several ambiguous formulations in the requirement. Eg. "... Derfor er det vigtigt, at store dele af løsningen kan afvikles på mobile enheder.", which leaves it open to interpretation how much a large part is, or "Det gælder både i forhold til at effektivisere tunge arbejdsgange via genbrug af data, deling af data og ikke mindst, at løsningen er medvirkende til god lovmedholdelighed i sagsbehandlingen" which isn't very clear about what exactly that entains, besides being able to handle some type of data in certain ways.

**Is there any information missing in the requirement?**

Information about the more specific needs and contraints of the product/solution in terms of user friendlyness is somewhat lacking, considering their formulations are generally speaking fairly vague, especially concerning what they consider to be adequate support from a digital solution.

**The requirement specifies a set of non-functional requirements. What is problematic about there formulation?**

Their formulation includes phrases and requirements that can be difficult to test such as "og derved skal den digitale understøttelse være medvirkende til, at kommunen er et attraktivt sted at arbejde"

**Rewrite the requirement according to what you identified as problematic in the three bullet points above.**

(Without knowing a lot about the actual product we have given examples of

what could have made sense to write. Changes are marked with blue)

Medarbejdere skal digitalt understøttes i udførelsen af deres kerneydelser, og derved skal den digitale understøttelse være medvirkende til at skabe mere effektivitet og overblik hos den enkelte medarbejder.

Kommunen har forskellige faglige målgrupper, som møder borgerne forskellige steder, og som arbejder, hvor borgerne er. Det kan være på gaden, i borgerens eget hjem, men også på dag- og døgninstitutioner uden for kontoret. Derfor er det vigtigt, at løsningens hovedfunktioner kan afvikles på mobile enheder.

Det er altafgørende, at medarbejderne føler sig godt understøttet af den nye digitale løsning. Det gælder både i forhold til at effektivisere tunge arbejdsgange via brug af caching og effektiv deling af <specifik data> overholdt inden for lovens rammer.

Et andet og vigtigt parameter er, at kommunen gerne vil fastholde sine dygtige medarbejdere med en løsning, der understøtter og guider dem i deres daglige opgaveløsning. En stabil og driftssikker løsning er en medvirkende faktor til større tilfredshed med ansættelsen i kommunen.

## Exercise 5

**Based on the video identify actors that interact with a music tracker software system. formulate three use cases in structured language that a software music tracker system has to support and express three non-functional requirements for a music tracker software system.**

The actor is the music producer/sound engineer. Another actor could be someone who wants to create a modification to the software.

A use case could be a beginner needs to learn and the system needs to give them a crash course. Another use case could be a musician trying to make a new song. A third could be a technician that needs to fix the system and the system needs to provide trouble-shooting options.

One non-functional requirement is that the software should be optimized and limited to music and not for creating other sounds. Another non-functional requirement for a music tracker could be the fact that the market is quite small, so the system will have to cater to that specific market. Because the market is small, there is probably also a limited amount of money to develop the software. One suggestion is to make software that is modifiable. This could be by either by the user or by the development team.

## Exercise 6

**Based on your observations and notes, write down a use case in structured language that a canteen payment system has to support. For this use case write down three requirements and categorize them as functional or non-functional.**

A use case for this situation could be one that involves the customer. The customer has to select what food they would like to eat and then choose a payment type. Then the customer has to pay for the meal and get a confirmation of the transaction in some way.

Requirement 1
Functional

One requirement for this use case is to have a list of all the food sold at the canteen. Such a list needs to be updatable. The pricing of the food should be included and also updatable.

Requirement 2
Non-functional

The system must have various payment options including mobile, card and cash.

Requirement 3
Functional

The system must provide proper feedback on which state it is in so the costumer knows when to do what (pay etc.).