1 C#

1.1 Generics

Compare the following two methods:

```
int GreaterCount<T, U>(IEnumerable<T> items, T x)
    where T : IComparable<T>;

int GreaterCount<T, U>(IEnumerable<T> items, T x)
    where T : U
    where U : IComparable<U>;
```

Both methods returns the amount of elements in items which are greater than x.

1.1.1 Explain in your own words what the type constraints mean for both methods.

The first method have a interface constraint that the type(T) must implement IComparable<T>. items must be comparable to each other.

The second method have a naked type constraint as well, meaning that the type of T must match the type of U, and the types of U must be comparable to each other.

1.2 Github Repository

Remaining C# solutions in this repository: https://github.com/A-Guldborg/Assignment01

2 Software Engineering

2.1 Exercise 1

Nouns = blue
Verbs = red

I want a version control system that records changes to a file or set of files over time so that I can recall specific versions later. This system should work on any kind of files may they contain source code, configuration data, diagrams, binaries, etc.

I want to use such a system to be able to revert selected files back to a previous state, revert the entire project back to a previous state, to compare changes over time, to see who last modified something that might be causing a problem, who introduced an issue and when, etc.

2.1.1 Noun/verb

Explain in which domain nouns and verbs that you identified are located.

The verbs are identified by having a certain behavior, that are connected to the noun. Each verb is split into a certain noun and with the object noun that it refers to.
The nouns are characterized by being some sort of object that could be a class.

Noun	Verbs
Version Control System	records(changes to files) recall(versions) revertFiles(previous state) revertProject(previous state) compare(changes)
File	contain(source code) contain(configuration data) contain(diagrams) contain(binaries) cause(Problems)
(Implicit) user/person	use(version control system) modified(file) introduce(issue)

2.1.2 libgit2sharp

The implementation in libgit2sharp does neither contain a class File nor a class State. Explain how that can be when libgit2sharp is an implementation of Git which is certainly a version control system as described above.
libgit2sharp does not contain a class file since each commit is a binary file (blob). Each commit points to a blob.

2.2 Exercise 2

In class we discussed so far the Coronapas App and Git as cases for software systems.

2.2.1 Types of applications

Categorize each of the two systems into Sommervilles types of applications. Note, the systems may not fall cleanly into a single category.

Coronapas: Information System

Git: A support environment

2.2.2 Arguments

Argue for why you choose certain categories for each system.

Coronapas, Information System: Its primary usage is to inform users of Corona vaccinations, by connecting information from different databases and to secure the persons data. It also has traits of being a support system, supporting the Danish government in conducting and controlling covid-tests in Denmark.

Git, Support System: The primary application for Git is to support developers in programming. However, it also functions as a database where different users can access the same information which indicates that Git also leans against the Information System category by maintaining data integrity.

2.3 Exercise 3

Sommerville describes that there are two kinds of software products. Describe for the Coronapas App and Git what kind of software product they are and provide arguments for your believes.

2.3.1 Git (Generic)

Easy for developers to download and start using. No specific software is created for each user or company.

2.3.2 Coronapas (Customized)

Is a software product made specific to Digitaliseringsstyrelsen to aid in the Corona crisis.

2.4 Exercise 4

Sommerville discusses quality of professional software, non-functional quality attributes, or product characteristics. Compare the Coronapas App, Git, and the Insulin pump control system (see SE chap 1.3.1) with respect to the quality attributes dependability, security, efficiency, and maintainability.

Do they all share the same characteristics with regards to these quality attributes or are these of varying importance to the three systems? Give examples for each of the three systems with regards to each if the quality attributes above.

2.5 Exercise 5

Inspect the implementations Gitlet and Git a bit more thoroughly than in class.

2.5.1 Explain why is there likely no architecture for Gitlet.

Gitlet is created to understand and teach Git, and is written in JavaScript. Gitlet is not intended to be used as a version control system (source), and is in layman terms a translation

	Git	Coronapas	Insulin Pump
Dependability Security	High priority for Git. It is important that the Software that is being worked on is not corrupted by some outside source. In the same manner, it is important that the code is the same as last time it was accessed.	Data integrity must be high for this app to work, otherwise the app might be rejected from the public	Since the software controls insulin for users with diabetes, of all the quality requirements, the software must provide a reliable system for the the insulin pump to avoid critical medical situations.
Efficiency	The commands should be fast and responsive, when many costumers use the system	If loading a passport takes a few seconds due to inefficiency or costs more in data storage, it is okay as long as the other quality aspects are kept prioritised. However since it is public money, it should not be overly wasteful either.	The insulin control system must also be efficient and be able to deliver insulin when required
Maintainability	New features should be able to be implemented and improved	Found bugs should be very easy to fix, and they are very likely to happen in an app that had a short development deadline and a huge number of users from day one.	While the purpose of the system should not change, the system should be maintainable when developing improvements with e.g. better collection of data

of git to JavaScript, so it can be used with JavaScript. A git command is thus a function calling an already existing function from Git, and the decisions for how the Git backend works has thus already been made at the creation of Git.

2.5.2 How could you infer the architecture of Git that was depicted in class without any more documentation, i.e., only the available source code?

You can look for the structure of directories and files, the documentation and comments contained in the source code.

2.5.3 Gitlet has a particular design that mimics the architecture of Git but that is implemented differently. Can you describe it in words?

As described in 5.1, and from the Gitlet Documentation, Gitlet is another version of Git based on JavaScript.

While Git is more backend related, with a frontend system as a Command Line Interface (CLI), Github Desktop or integrated in IDEs, Gitlet could be considered a JavaScript frontend version of Git.

2.5.4 Git and Gitlet are designed with respect to different quality attributes (product characteristics). Name some of the most prominent quality attributes that influence the design of each of the two systems.

Gitlet was made with the purpose of learning the internals of git. Everyone can contribute as it is Open source, so more emphazise is made on maintainability than security. Gitlet focuses on expressing Git in an easy-to-understand way, paying attention to acceptability. Because Gitlet is written in JavaScript, it's purpose is to make an interpretation of Git, to make it more understandable.

2.6 Exercise 6

Look at the following two cases of issues with health care software systems:

- "Softwareproblemer skadede mere end 100 patienter på amerikansk hospital"
- "Kodefejl i Sundhedsplatformen: Fem patienter har fået forkert dosis medicin"

2.6.1 Based on the articles, describe the reasons that caused the respective issues.

The people using the hospitals information system were not aware of the "unknown queueffeature, which might be caused by either a lack of training from the hospital management or a lack in the requirements specifications.

For the wrong dosage issue, it seems that the issue is caused by a change in the code where the math used to calculate medicine dosages has been changed where it should not have been.

2.6.2 Describe potential solutions to the problem.

One solution for the unknown queue issue at the American hospital would be, that whenever an item, in this case an order for an examination, is inserted in a queue, the actor (doctor) is notified in which queue this item has been placed.

Another solution would be to only enable the ”save”/”send” button once the item is ready to be inserted in a queue that is not the unknown catch-all queue.

For the wrong dosage issue, a solution could be to develop a greater test suite that would disallow source code changes unless they pass a grand number of tests.

Another solution could be to, like for the American hospital issue, provide feedback for the actors in the system, such that whenever a dosage was inserted in the system, it would create a prompt that had to be accepted that echoes the result of the dosage calculation.

2.6.3 Compare your solutions to the proposed solutions in the articles, which one would you as a software engineer recommend? Argue for your recommendations.

Our solution to the American Hospital is similar to the proposed solution in regards to more user confirmation and alerts, but without the suggestion of automatic processes.

For Sundhedsplatformen, we suggest that the user confirms medical doses, while they suggest more throughout testing when connecting their different systems.

2.6.4 Discuss ethical dilemmas in case you were developing either of these health care systems.

When developing such systems, a software solution compared to a paper journal, can always come with bugs, flaws or fail to deliver at some point. It is a ethical dilemma, to allow such possibilities of failure when using a software system that involves real patients.