Repo links: CHES: https://github.com/Stender98/BDSA2021
ANMB: https://github.com/AMB-F/BDSA/tree/main/assignment01

Problem 1
What is meant by "knowledge acquisition is not sequential"? Provide a concrete example of knowledge acquisition that illustrates this.

Solution Knowledge acquisition was commonly thought to be linear, but new information can in practice invalidate current models and thereby challenge what was recognized as facts. This is especially valid in software engineering, which is a young and constantly evolving field of study, where known models are often challenged by new ideas. An example in history could be that the rain forest was commonly mistaken for being the lungs of the planet, since they consumed a lot of greenhouses gasses, specifically CO_2. While it was later found to be a self contained system, that is actually CO_2 neutral.

Problem 2
Specify which of the following decisions were made during requirements or system design:
"The ticket distributor is composed of a user interface subsystem, a subsystem for computing tariff, and a network subsystem managing communication with the central computer."
"The ticket distributor will use PowerPC processor chips."
"The ticket distributor provides the traveler with an on-line help."

Solution The first decision is made as a system design specification. The second is also a system design decision if made by the developers (else, it is a requirement). The third decision is a system requirement.

Problem 3
In the following description, explain when the term account is used as an application domain concept and when as a solution domain concept:
"Assume you are developing an online system for managing bank accounts for mobile customers. A major design issue is how to provide access to the accounts when the customer cannot establish an online connection. One proposal is that accounts are made available on the mobile computer, even if the server is not up. In this case, the accounts show the amounts from the last connected session."

Solution The first use of the term account is a application domain concept as it refers to the manifestation of bank accounts. The second usage of account, is dependent on the context of the issue. It could both mean accessing the online accounts in case of connection failures as well as access to the actual bank accounts, while the last two occurrences is solution domain concepts.

Problem 4
A passenger aircraft is composed of several millions of individual parts and requires thousands of persons to assemble. A four-lane highway bridge is another example of complexity. The first version of Word for Windows, a word processor released by Microsoft in November 1989, required 55 person-years, resulted into 249,000 lines of source code, and was delivered 4 years late. Aircraft and highway bridges are usually delivered on time and below budget, whereas software is often not. Discuss what are, in your opinion, the differences between developing an aircraft, a bridge, and a word processor, which would cause this situation.

Solution
Assembling a plane can in theory be done in the same way multiple times, but developing an aircraft, a bridge or a piece of software is usually an entirely new challenge, that requires previously unknown solutions, even though known patterns in design can be applied. Most aircraft has wings, different bridges can use similar support structures and different word processors might be very similar at the root of the source code.

Due to software engineering being a much younger field than both bridge designing and plane building, there is a huge data gap regarding "best practices". Due to this, the research and analysis phase might take very long time while still not resulting in the optimal product description resulting in faulty programs that

end up needing much redesign.

A new bridge or airplane could maybe easily be compared to existing bridges or planes (especially if the new airplane is simply a replica of an existing model) and as such help give a realistic time frame. Even at this point in history, most programs will be of a limited generation or might be first of its type, meaning that there is very limited understanding of time usage. This could then easily lead to projects taking much longer than planned, as the estimated times are based on limited cases, ending up pushing the deadlines significantly.

Both planes and bridges of course have a lot of freedom in their design, but also a long list of constraints, that might make it easier to chose a direction from the more limited list. In programming, the great freedom of choice is a huge advancement, but could maybe also be part of exceeding time use, as analyzing a problem for the best solution is truly an infinite process, when the possible solutions are endless.

There are still many similarities, between engineering in different fields, the idea of assembling a plane, a bridge or a word processor, still requires structured planning and a group of specialists, that each fill their own roles. It might be a common misconception in software engineering, that you just need a "programmer"/"IT-guy" to create a new piece of software, but to succeed at a larger scale, similar to building a bridge, requires different people that each has their own expertise.

Problem 5
Specify which of these statements are functional requirements and which are nonfunctional requirements:
"The TicketDistributor must enable a traveler to buy weekly passes."
"The TicketDistributor must be written in Java."
"The TicketDistributor must be easy to use."
"The TicketDistributor must always be available."
"The TicketDistributor must provide a phone number to call when it fails."

Solution The first and the last requirements are functional requirements. One and five describe specific functionality, that must be in one way or another, implemented as code. While requirement 2-4 are non-functional requirements, that describe indirectly requirements for the system, that are not necessarily tied to the functionality that was defined by the customer.

Problem 6
What is the purpose of modeling?

Solution A model is a higher level of abstraction, where we try to analyse the problem space and represent it in our solution space. It can be used as guidance, while specifying requirements, and can be a blue print for our product. It can also outlay a lot of the requirements for our system.'