Test with no date
(https://drive.google.com/drive/folders/1bZAFZsWGN6cseigeaaXAS9GwVvA1kb_1?fbclid=IwAR0zfslNu4QhPrzVSwaD9RrsQF3bPrRVH37RJyv_D_XegkB_sSOF65u3DoA)

1. Building software has particular characteristics that pose new challenges, making typical approaches of traditional engineering disciplines hard to apply to software systems.
Examples of such software characteristics are:

a)
Software is mainly made up of abstractions,
written in concrete languages.
Sometimes is hard to comprehend Software
developed by others, because of how abstract it is.

Software Engineering:
Invisible constructions (logical product)
1. Temporal nature of software is complex and hard to
understand (discrete systems)
2. Big needs to change and evolve along lifespan
3. Software must be very easy to adapt; each case is a case
4. Underlying technologies evolve very fast
5. Short history (since 1960?)
6. No â▫▫nature lawsâ▫▫ or â▫▫phyical lawsâ▫▫ to conform to...

2. RUP is heavily supported by UML. In your opinion, in what phase of the process is most UML developed? Elaborate you choice by describing who is responsible for elaborating the UML and what diagrams are most often adopted.

b)
RUP is heavily supported by UML. In my opinion, the elaboration phase is where most UML is developed.
In this stage the developers take a closer look at the project to determine its architecture foundation and to evaluate the architecture in relation to the project.

The UML is the most widely used too to represent all major components, users and their interaction (Model visually). The types of UML's most used to model the architecture visually are:
    1. Use-Case Diagram, Class Diagram, State Machine Diagram, Component Diagram, Deployment Diagram: they show a static view of the project.
    2. Communication Diagram, Sequence Diagram: they show the dynamic behavior of the project.

The system analyst is usually responsible for the development of the UML, and the project manager contributes/reviews it.

5. Requirements engineering refers to the process of defining, documenting and maintaining requirements in the software emginiring process.
Other important characteristics of the process of requirements engineering are:

c)
Requirements engineering is very valued in waterfall-like processes, being always the first phase of the development process. More recent, processes assume the importance of Requirements engineering along all the process, even at usability studies phases.

The first two stages of a waterfall-like process are dedicated to requirements engineering (system and Software requirements). In the case of more recent processes like agile methods, they usually use incremental requirements engineering
and may express requirements as user stories.

The A option is wrong. The flow of requirements engineering is:
1. Requirements elicitation; 2. Requirements analysis and negotiation; 3. Requirements specification; 4. Requirements validation and 5: Requirements management. So the last step is management not validation.

The B option is wrong because both system and user requirements are functional.

The D option is wrong because engineering requirements are not ignore in agile methods. Rather the system requirements are considered a waste of time since they
change so quickly. Software requirements are considered very important and are usually expressed via user stories through the development.

6. Software is complex to build. If we were to build all our applications from scratch, software engineering would be an inefficient discipline. Fortunately, software can be composed and reused, enabling us to take on existing software as a scaffold upon which we can build our own.
When building and running the program below in C, which level of software reuse is being applied?

d)
It is object level because we are directly using objects from a library: we are using the "printf" object directly from the Std library.

7. It is common knowledge in Software Engineering that "software must evolve or it will die". Evolving software is thus a key practice to ensure software longevity. What do you think is a major factor of entropy to evolving a system?

(entropy = chaos = change)

b) The change proposals.

The Software is always evolving: always changing in response to changing customer needs. This stage will cause a major entropy to an evolving system since requirements are in constant change and the project and its team must adapt frequently to (sometimes drastic) changes.

Definition of Software entropy:
The second law of thermodynamics, in principle, states that a closed system's disorder cannot be reduced, it can only remain unchanged or increase. A measure of this disorder is entropy. This law also seems plausible for software systems; as a system is modified, its disorder, or entropy, tends to increase. This is known as software entropy.

1. A computer program that is used will be modified
2. When a program is modified, its complexity will increase, provided that one does not actively work against this.

9. Software project management is concerned with activities to ensure that software is delivered on time, on budget and in accordance with the requirements of the organizations developing and procuring software. Poor people management is an important contributor to project failur.

Which of the following do you consider the worst practice to manage a Software team?

c) no team leaders, all team elements are equal

It is very important that a project has team leaders. Team leaders are responsible
for creating a positive working environment through effective communication, team
building activities, and reflective listening. team leaders must be effective at

identifying and resolving team problems in order for the project to succeed. To do this, team leaders must have an understanding of the team dynamic for decision-making and must be able to manage conflict among personality differences and barriers. A good team leader will make the development of a project run smoothly.

10. ---