AF-IAST
Skilling Pakistan

PAK-AUSTRIA FACHHOCHSCHULE:
INSTITUTE OF APPLIED SCIENCES AND TECHNOLOGY

IT & Computer Science

Instructor Name: Zubaria Noureen

Name: Abuzar Khan **Reg #:** B22F1053SE23

Section: SE (GREEN) **Date:** 28/09/2025

Software Re-Engineering

Assignment 01

Answers

Answer 01:

Exception: (c) Perfective maintenance.

Justification (concise and clear):

- Corrective maintenance fixes defects so the system conforms to the existing functional spec it changes code/design but not the spec itself.

- Adaptive maintenance modifies the system to run in a changed environment (OS, platform, regulations) without altering its functional requirements.

- Preventive maintenance is refactoring, restructuring, or other changes to reduce future faults or improve maintainability functionality remains the same.

- Perfective maintenance, however, is intended to improve or add functionality (enhancements requested by users or to extend capabilities). That changes the functional specification (new/changed behaviour) so it is the one that cannot be described as "leaves the functional specification unchanged."

Example:
Adding a new "export to XLSX" feature is perfective it changes what the system does and thus the functional spec. Fixing a bug that prevented export from working is corrective the spec stays the same.

Answer 02:

Classification: This feature that complex changes can introduce new defects should be considered a risk (commonly called *regression risk*).

Why (short):

- A risk is a potential adverse event: here, a change may introduce defects (probability) and those defects have an impact (time/cost/reliability).

- It's not a benefit. While introduced defects might later produce learning or improvements, they are fundamentally negative possibilities.

- It has a cost component (if defects occur they cause rework/cost), but primary classification for planning and decision-making is risk. Treat it as a risk that also implies possible additional cost if realized.

Brief mitigation suggestions (what your answer should mention):

- Perform impact analysis and traceability before making the change.
- Write/extend automated regression tests and run them in CI.
- Use incremental/controlled rollout (feature toggles, canary releases) and have a rollback plan.
- Code review, static analysis, and high-test coverage to reduce probability.
- Estimate the risk (probability × impact) and include it in the change decision.

END!