Victoria University of Bangladesh
Name: Md. Ziaul Hoque "Sohel"
Student ID: 2221220031
Course Title: Software-Engineering-scaled
Course Code: CSI-321
Batch: 22nd (evening)
Semester: Summer-2023

Ans to the Que No 1(A)

Software Engineering:

Software engineering is a detailed study of engineering to the design, development and maintenance of software. Software engineering was introduced to address the issues of low-quality software projects. Problems arise when a software generally exceeds timelines, budgets, and reduced levels of quality. It ensures that the application is built consistently, correctly, on time and on budget and within requirements. The demand of software engineering also emerged to cater to the immense rate of change in user requirements and environment on which application is supposed to be working.

Ans to the Que No 1(B)

Waterfall Model:

The Waterfall Model was the first Process Model to be introduced. It is also referred to as a linear-sequential life cycle model. It is very simple to understand and use. In a waterfall model, each phase must be completed before the next phase can begin and there is no overlapping in the phases. The Waterfall model is the earliest SDLC approach that was used for software development. The waterfall Model illustrates the software development process in a linear sequential flow. This means that any phase in the development process begins only if the previous phase is complete. In this waterfall model, the phases do not overlap.

Validation Process:

Validation is the process of checking whether the software product is up to the mark or in other words product has high-level requirements. It is the process of checking the validation of the product i.e. it checks what we are developing is the right product. it is a validation of actual and expected products. Validation is simply known as Dynamic Testing.

Ans to the Que No 1(C)

Characteristics of Software Requirements Specification (SRS):

1. Complete: The SRS should include all the requirements for the software system, including both functional and non-functional requirements.
2. Consistent: The SRS should be consistent in its use of terminology and formatting, and should be free of contradictions.
3. Unambiguous: The SRS should be clear and specific, and should avoid using vague or imprecise language.
4. Traceable: The SRS should be traceable to other documents and artifacts, such as use cases and user stories, to ensure that all requirements are being met.
5. Verifiable: The SRS should be verifiable, which means that the requirements can be tested and validated to ensure that they are being met.
6. Modifiable: The SRS should be modifiable, so that it can be updated and changed as the software development process progresses.

7. Prioritized: The SRS should prioritize requirements, so that the most important requirements are addressed first.
8. Testable: The SRS should be written in a way that allows the requirements to be tested and validated.
9. High-level and low-level: The SRS should provide both high-level requirements (such as overall system objectives) and low-level requirements (such as detailed functional requirements).
10. Relevant: The SRS should be relevant to the software system that is being developed, and should not include unnecessary or irrelevant information.
11. Human-readable: The SRS should be written in a way that is easy for non-technical stakeholders to understand and review.
12. Aligned with business goals: The SRS should be aligned with the overall business goals and objectives of the organization, so that the software system meets the needs of the business.
13. Agile methodologies: Agile methodologies, such as Scrum and Kanban, provide an iterative approach to requirements capturing and validation, where requirements are captured and validated in small chunks of functionality and feedback is gathered from the customer.

Ans to the Que No 2(A)

Software maintenance:

There are four types of software maintenance:

- Corrective Software Maintenance
- Adaptive Software Maintenance
- Perfective Software Maintenance
- Preventive Software Maintenance

Corrective Software Maintenance: Corrective software maintenance is what one would typically associate with the maintenance of any kind. Correct software maintenance addresses the errors and faults within software applications that could impact various parts of your software, including the design, logic, and code.

Adaptive Software Maintenance: Adaptive software maintenance becomes important when the environment of your software changes. This can be brought on by changes to the operating system, hardware, software dependencies, Cloud storage, or even changes within the operating system. Sometimes, adaptive software maintenance reflects organizational policies or rules as well.

Perfective Software Maintenance: Perfective software maintenance focuses on the evolution of requirements and features that existing in your system. As users interact with your applications, they may notice things that you did not or suggest new features that they would like as part of the software, which could become future projects or enhancements.

Preventive Software Maintenance: Preventative Software Maintenance helps to make changes and adaptations to your software so that it can work for a longer period of time. The focus of the type of maintenance is to prevent the deterioration of your software as it continues to adapt and change. These services can include optimizing code and updating documentation as needed.

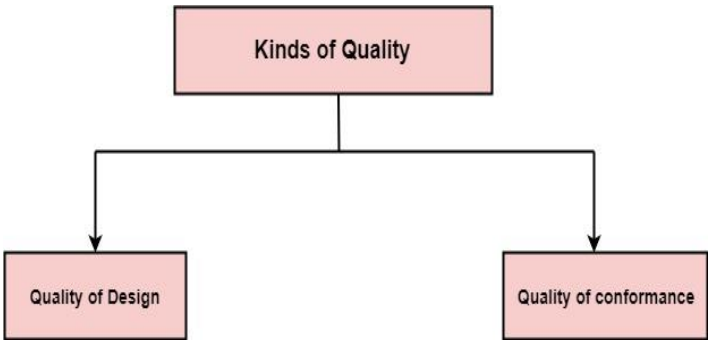
Ans to the Que No 2(B)

Difference between Black box & white box testing:

Black Box Testing	White Box Testing
It is a way of software testing in which the internal structure or the program or the code is hidden and nothing is known about it.	It is a way of testing the software in which the tester has knowledge about the internal structure or the code or the program of the software.
Implementation of code is not needed for black box testing.	Code implementation is necessary for white box testing.
It is mostly done by software testers.	It is mostly done by software developers.
No knowledge of implementation is needed.	Knowledge of implementation is required.
It can be referred to as outer or external software testing.	It is the inner or the internal software testing.
It is a functional test of the software.	It is a structural test of the software.
This testing can be initiated based on the requirement specifications document.	This type of testing of software is started after a detail design document.
No knowledge of programming is required.	It is mandatory to have knowledge of programming.

Ans to the Que No 2(C)

Software Quality Assurance (SQA):



There are two kinds of Quality:
Quality of Design: Quality of Design refers to the characteristics that designers specify for an item. The grade of materials, tolerances, and performance specifications that all contribute to the quality of design.

Quality of conformance: Quality of conformance is the degree to which the design specifications are followed during manufacturing. Greater the degree of conformance, the higher is the level of quality of conformance.

Software Quality: Software Quality is defined as the conformance to explicitly state functional and performance requirements, explicitly documented development standards, and inherent characteristics that are expected of all professionally developed software.

Quality Control: Quality Control involves a series of inspections, reviews, and tests used throughout the software process to ensure each work product meets the requirements place upon it. Quality control includes a feedback loop to the process that created the work product.
Quality Assurance: Quality Assurance is the preventive set of activities that provide greater confidence that the project will be completed successfully.

Quality Assurance focuses on how the engineering and management activity will be done?
As anyone is interested in the quality of the final product, it should be assured that we are building the right product.

Ans to the Que No 3(A)

Software Project Management:

Software project management is an art and discipline of planning and supervising software projects. It is a sub-discipline of software project management in which software projects planned, implemented, monitored and controlled. It is a procedure of managing, allocating and timing resources to develop computer software that fulfills requirements.

In software Project Management, the client and the developers need to know the length, period and cost of the project.

Ans to the Que No 3(B)

Project Manager role and responsibilities:

Without any sound and fury, here is a rundown of all the fundamental, mission-critical tasks that fall within a project manager's purview:

- Planning Everything from Execution to Delivery

A manager with effective project management skills will always have a plan ready to maximize output while minimizing input. Project managers are tasked with determining the most efficient means of achieving the desired outcomes for their clients and other stakeholders as soon as possible. The project manager should choose the method, such as Agile, Waterfall, kanban etc., whichever would be suitable for this task.

- Oversee the Software Development Team

A project manager's job affords them unusual insight into the many moving parts of a team's task forces. The project manager may collaborate with following members of a team:

- Business analysts
- Web designers
- Software developers
- Content creators
- Graphic designers
- Sales and advertising teams
- Marketing teams

Not to mention a plethora of others! However, it's possible that these departments won't be able to communicate with one another or have a clear view of the project's final product. Relationships between these groups must be managed, updates must be sent to the project owner, and data must be shared throughout departments and teams.

- Delegating Work Effectively

It is crucial to intelligently allocate work to teams when dealing with complex scenarios like large projects or several jobs inside a project. Every project manager must practice and master this form of leadership, and doing so successfully is ultimately the manager's job as a part of risk management. A manager shouldn't exploit their position to make their employees feel bad about themselves. Team members can be more efficient and productive if they are given higher-priority

duties. Managers should assess their staff' abilities and assign work accordingly. If someone wants to inspire trust in their team, they need to demonstrate effective leadership skills through delegating responsibly.

- ❖ Monitoring Progress and Tracking Roadblocks

For the most part, a software project manager's duties concentrate on keeping tabs on ongoing endeavors. A project manager's responsibility after the project has begun is to monitor progress and ensure that work is proceeding as planned. During the project's midsection, progress is accomplished via a variety of methods, including periodic reviews, briefings, and spontaneous updates. If the project managers pick the right management system, they'll have less work to do.

- ❖ Managing the Deployment Deliverables

Delivery of deliverables on schedule and within budget is another key responsibility of the project manager. It's part of their work to question things like:

- Can you tell me about the alterations being made to the company?
- Exactly what is the group up to at the moment?
- Is there a reason for this action?
- What sort of business opportunity or threat exists?
- How do you propose we go about doing this?
- What are the most well-liked methods for managing projects?
- Who exactly performs what?
- Where can we find the project's files and paperwork?
- When and where will meetings be held? What are the requirements?
- What time frame are we looking at for these activities?

Ans to the Que No 3(C)

Iterative Model advantage:

- Some working functionality can be developed quickly and early in the life cycle.
- Results are obtained early and periodically.
- Parallel development can be planned.
- Progress can be measured.
- Less costly to change the scope/requirements.
- Testing and debugging during smaller iteration is easy.
- Risks are identified and resolved during iteration; and each iteration is an easily managed milestone.
- Easier to manage risk - High risk part is done first.
- With every increment, operational product is delivered.
- Issues, challenges and risks identified from each increment can be utilized/applied to the next increment.
- Risk analysis is better.
- It supports changing requirements.
- Initial Operating time is less.
- Better suited for large and mission-critical projects.

Iterative Model disadvantage:

- More resources may be required.
- Although cost of change is lesser, but it is not very suitable for changing requirements.
- More management attention is required.
- System architecture or design issues may arise because not all requirements are gathered in the beginning of the entire life cycle.
- Defining increments may require definition of the complete system.
- Not suitable for smaller projects.
- Management complexity is more.
- End of project may not be known which is a risk.
- Highly skilled resources are required for risk analysis.
- Projects progress is highly dependent upon the risk analysis phase.

Ans to the Que No 4(A)

Quality:

Software quality product is defined in term of its fitness of purpose. That is, a quality product does precisely what the users want it to do. For software products, the fitness of use is generally explained in terms of satisfaction of the requirements laid down in the SRS document. Although "fitness of purpose" is a satisfactory interpretation of quality for many devices such as a car, a table fan, a grinding machine, etc. For software products, "fitness of purpose" is not a wholly satisfactory definition of quality.

Importance of Quality:

Maintaining software quality helps reduce problems and errors in the final product. It also allows a company to meet the expectations and requirements of customers. This makes it possible for companies to build high-quality products that, in turn, increase the trust and loyalty of consumers. By following the standards and procedures defined by a QA program, teams can uncover issues before they emerge.

Ans to the Que No 4(B)

Software Quality Assurance (SQA) Activities:

1. SQA Management Plan:
   Make a plan for how you will carry out the sqa throughout the project. Think about which set of software engineering activities are the best for project. check level of sqa team skills.

2. Set The Check Points:
   SQA team should set checkpoints. Evaluate the performance of the project on the basis of collected data on different check points.

3. Multi testing Strategy:
   Do not depend on a single testing approach. When you have a lot of testing approaches available use them.

4. Measure Change Impact:
   The changes for making the correction of an error sometimes re introduces more errors

keep the measure of impact of change on project. Reset the new change to change check the compatibility of this fix with whole project.

5. Manage Good Relations:
   In the working environment managing good relations with other teams involved in the project development is mandatory. Bad relation of sqa team with programmers' team will impact directly and badly on project. Don't play politics.

END