FullName: Mariam omar dirie
Student_ID: HU0003699

Assignment: Software Quality Assurance

Question 1: Understanding Software (4 Marks)

a. Definition of Software and Difference from Hardware

Software is a collection of data, programs, and procedures that instruct a computer on how to perform specific tasks. Unlike hardware, which is the physical components of a computer (e.g., CPU, memory, keyboard), software is intangible and consists of code written in programming languages.

Example: Microsoft Word is software, while the laptop it runs on is hardware.

b. Main Categories of Software

1. **System Software** – Manages hardware and provides a platform for running applications. Exemple: Windows, Linux.
2. **Application Software** – Designed for end-users to perform specific tasks. Example: Microsoft Excel, Photoshop.
3. **Embedded Software** – Built into devices to control their functions. Example: Software in washing machines or automotive control systems.

c. Software Quality vs Physical Product Quality

Ensuring software quality is harder because software is intangible, complex, and often changes during development. Physical products can be visually inspected or measured, while software defects are logical and may appear only under certain conditions. Moreover, user interactions and environments make software testing unpredictable.

Question 2: Causes of Software Errors (4 Marks)

a. Classification of Software Errors

1) **Human Errors** – Mistakes made by developers or users, such as logic errors or incorrect coding.
2) **Process Errors** – Flaws in the software development process, e.g., poor requirements or inadequate reviews.
3) **Tool Errors** – Issues caused by software tools, compilers, or libraries.
4) **Environmental Errors** – Failures due to external factors like hardware faults, power failures, or network issues.

b. Examples

Error Type	Example 1	Example2
Human	Coder writes wrong formula	Forgot to check user input
Process	Missing requirements	No code review done
Tool	Software tool has a bug	Old library causes error
Environmental	Power outage stops system	Slow internet breaks app

c. Preventive Measures

- Code reviews and peer programming
- Clear requirement documentation
- Automated testing and continuous integration
- Using reliable, updated tools
- System monitoring and backup plans

Question 3: Definition and Objectives of SQA (4 Marks)

a. Definition of SQA

Software Quality Assurance (SQA) is a systematic process that ensures software meets defined quality standards and performs reliably throughout its lifecycle.

b. Objectives of SQA

- Ensure compliance with standards and procedures
- Detect and prevent defects early
- Improve development efficiency
- Maintain customer satisfaction
- Provide continuous improvement feedback loops

c. SQA vs Testing

Testing is one phase within SQA focused on finding bugs in a product. SQA encompasses the entire development process — including planning, process audits, documentation, and post-release evaluations.

Example:

- **Testing:** Checking login functionality works correctly.
- **SQA:** Verifying that the overall development process ensures security, usability, and compliance.

Question 4: Software Quality Factors (4 Marks)

a. Key Quality Factors

1. **Correctness** – Accuracy of output against requirements.
2. **Reliability** – Ability to operate without failure over time.
3. **Usability** – Ease of use and user experience.

4. **Efficiency** – Optimal resource utilization and performance.
5. **Maintainability** – Ease of modifying and fixing software.
6. **Portability** – Ability to function across environments.

b. Example Assessment: E-Commerce Website (e.g., Amazon)

- ***Correctness****:* Displays accurate prices and order details.
- ***Reliability****:* Rarely experiences downtime.
- ***Usability****:* User-friendly design with clear navigation.
- ***Efficiency****:* Fast page loading and optimized search.
- ***Maintainability****:* Regular updates and bug fixes.
- ***Portability****:* Works across browsers and mobile devices.

c. Most Critical Factor in Modern Systems

Reliability is the most critical, as users expect systems — particularly financial, healthcare, or cloud services — to operate continuously without failures that could cause data loss or service disruption.

Question 5: Factors Affecting Intensity of Quality Assurance (4 Marks)

a. Key Factors Influencing SQA Intensity

1. **Project Size** – Larger projects need more testing and documentation.
2. **Complexity** – Complex algorithms or integrations increase QA effort.
3. **Tools and Technology** – Use of automated tools can reduce manual workload.
4. **Team Skills** – Skilled developers and testers produce higher-quality outputs.
5. **Risk Level** – High-risk systems demand more rigorous assurance.

b. Comparison: Critical vs General Applications

- *Critical Systems (e.g., Healthcare, Banking):* Require formal verification, redundancy, and compliance with standards (e.g., ISO 9001).
- *General Applications (e.g., Mobile Games):* Focus more on usability and quick iteration rather than formal testing.

c. Balancing Cost and Quality

- Implement risk-based testing — allocate effort based on impact.
- Automate repetitive tests.
- Adopt Agile and DevOps practices for continuous integration.
- Maintain minimum viable documentation for lean efficiency.

Question 6: Case Study – Ariane 5 Rocket Failure (5 Marks)

a. Category of Error

The failure was primarily a **process error** — the reuse of Ariane 4's software without adequate adaptation or testing for Ariane 5's different flight dynamics. It also included a **human error** in overlooking this incompatibility.

b. Role of SQA in Prevention

Proper SQA reviews, simulation testing, and validation against new requirements could have revealed the integer overflow. Independent verification and validation (IV&V) would have detected the mismatch during pre-launch analysis.

c. Quality Factors Compromised

- **Reliability:** System failed catastrophically under real conditions.
- **Maintainability:** Lack of adaptable design prevented safe reuse.

d. Lessons Learned

- Conduct system-specific testing even for reused software.
- Implement strong SQA review gates before deployment.
- Prioritize fail-safe mechanisms and defensive programming.
- Treat software as a critical component of safety systems, not a secondary concern.

🔗 Suggested Links / Resources

Reference's	Possible Links or Ressource
Pressman, R. S., & Maxim, B. R. (2020). *Software engineering: A practitioner's approach* (9th ed.). McGowan-Hill.	The publisher page for this edition: *Software Engineering: A Practitioner's Approach* on McGowan-Hill site McGraw Hill+1
European Space Agence (ESA). (1996). *Ariane 5 Flight 501 failure report*	ESA press release / inquiry board summary: *Ariane 501 – Presentation of Inquiry Board report* European Space Agency Also, full versions of the report are mirrored (e.g. via MIT / NASA forums) sunnyday.mit.edu+1
Gérard Le Lann. (1996). *The Ariane 5 Flight 501 Failure — A Case Study in System Engineering for Computing Systems*	INRIA / research report versions available online (via ResearchGate, INRIA archives) ResearchGate+2ResearchGate+2