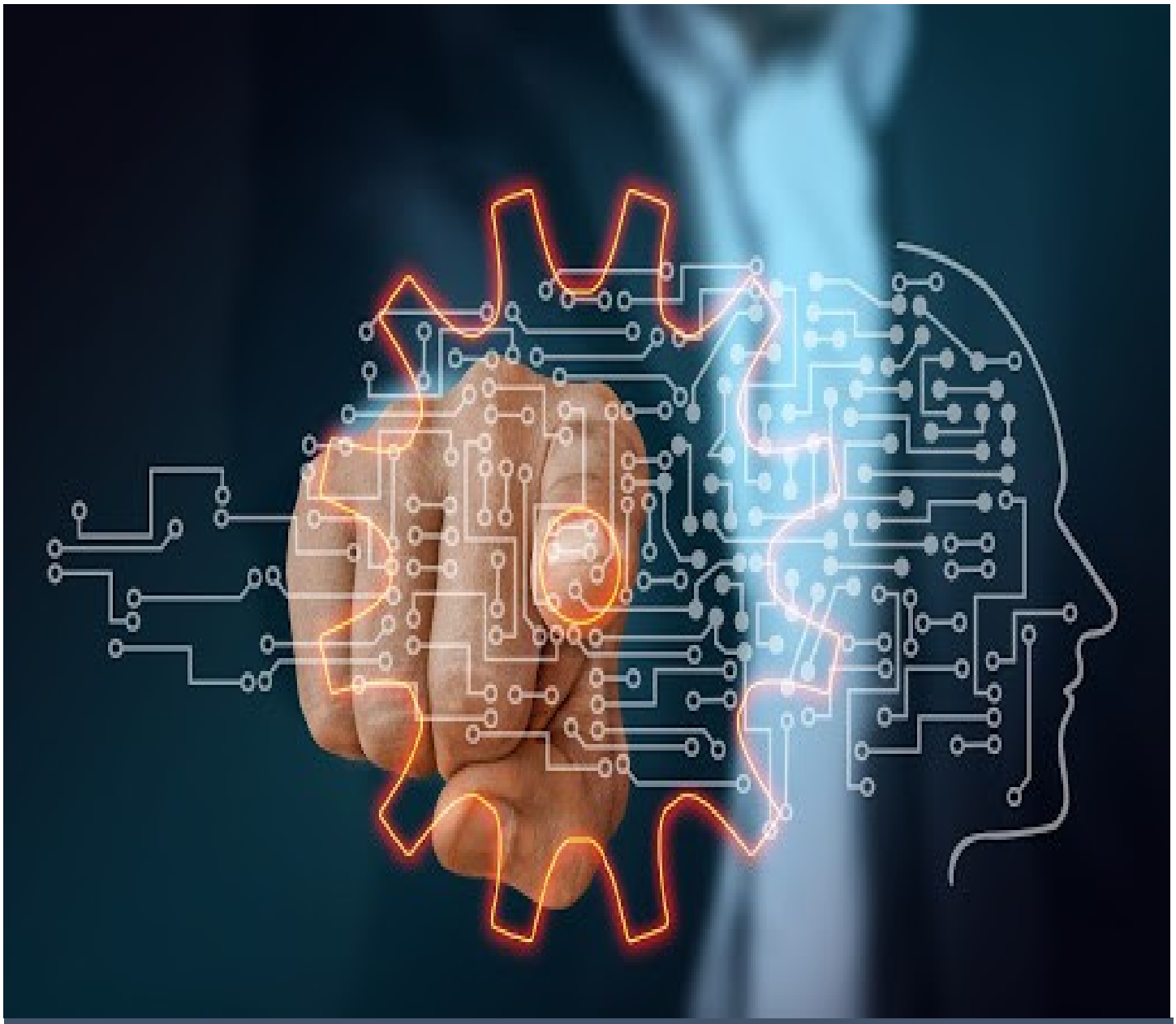

QUESTION 1

Q1.1

- **Requirements Analysis**
  Understand and examine the necessities of the software program.
- **Test Planning**
  Create an in depth check plan which incorporates scope, goals, sources, timetable, and risks.
- **Test Design**
  Develop particular test cases based on the requirements and look at plan.
- **Test Environment Setup**
  Establish the important hardware and software configurations for testing.
- **Test Execution**
  Perform the actual checking out by executing the organized test cases.
- **Defect Reporting**
  Document any defects or discrepancies discovered during trying out.
- **Defect Tracking And Management**
  Ensure that new code modifications do not negatively affect current functionalities.
- **Regression Testing**
  Manage defects by means of assigning priority, repute, and tracking their resolution.
- **Acceptance Testing**
  Conduct consumer popularity testing (UAT) to validate that the software program meets the person's necessities.

Q.1.2

- **Verification:**

Purpose:

The main aim of verification is to make certain that the software program conforms to its specs and necessities. It pursuits to reply the question, "Are we constructing the product proper?".

When:

Verification is carried out for the duration of the improvement phase. It entails activities like code critiques, walkthroughs, and inspections.

Examples:

Requirements Analysis: This involves reviewing the necessities documentation to make sure that they are clean, whole, and properly defined. For instance, the verification process might perceive ambiguous or contradictory requirements.

Unit Testing: This is a shape of verification checking out wherein character devices or additives of the software program are examined in isolation to make certain they function as meant. For instance, checking out a characteristic that calculates the overall of a purchasing cart.

- **Validation**:

Purpose:

The primary aim of validation is to make sure that the software program application meets the customer's needs and expectancies. It addresses the query, "Are we constructing the right product?"

When:

Validation is finished after the software has been developed. It includes sports like character popularity trying out (UAT) and beta finding out.

Examples:

Beta Testing: This consists of releasing a limited model of the software program software to a pick company of clients outdoor of the improvement team. They use the software program in a real-international environment and provide remarks on any problems or improvements. For example, a social media platform may conduct beta trying out to build up consumer comments before a entire release.

Acceptance Criteria Verification: This entails confirming that the software program meets the predefined reputation criteria mentioned inside the necessities. For example, if a demand specifies that a net website online ought to load in plenty less than three seconds, validation guarantees that this criterion is met.

Q.1.3 **Focus:**

Load testing out specializes in checking out the device below normal or anticipated load conditions.

Stress testing out pushes the machine beyond its limits to recognize the way it behaves under intense conditions.

Goal:

Load testing goals to choose out basic overall performance bottlenecks and ensure the device performs acceptably underneath commonplace situations.

Stress testing goals to find the breaking factors or failure modes of the device.

Scenario:

Load testing simulates realistic patron behaviour and transaction patterns.

Stress testing simulates severe situations, which can be now not going to stand up in everyday utilization.

Outcome:

In **load testing**, the device is predicted to carry out properly inside its defined ability.

In **stress testing**, the machine may additionally moreover exhibit symptoms of pressure or maybe fail, but the goal is to apprehend its limitations.

Q1.4.1 Test Case 1 (age=60), the statement coverage achieved out is 100%. This means that every line of the code in the snippet was executed at least once during the test case.

Q.1.4.2 In this test case, we both branches of the selection point:

The route in which the condition age >= 60 is true (entered IF block).

The direction wherein the circumstance age >= 60 is false (skipped ELSE block).

Therefore, for Test Case 1, the decision coverage done is 100%. This way that both viable outcomes of the decision factor have been accomplished at some point of the test case.

Q.1.5 Component testing is an important practice that ensures the reliability, correctness, and maintainability of individual gadgets of code. It bureaucracy the inspiration upon which complex software structures are built and incorporated. Without thorough component testing, it becomes a challenge to broaden and preserve super software program.

QUESTION 2

Q.2.1 https://www.canva.com/design/DAFwGV2q68c/uQZBd7FUgVMXCthTH7c4oQ/edit?utm_content=DAFwGV2q68c&utm_campaign=designshare&utm_medium=link2&utm_source=sharebutton

Q.2.2

- **Incomplete or Inadequate Requirements:**

Risk: Testing primarily based on incomplete or vague requirements can result in ignored defects or wrong capability.

Example: Assume a requirement specifies that a login show have to have "appropriate safety capabilities". Without smooth definitions, the finding out team can also interpret this in a different way, possibly essential to safety vulnerabilities.

- **Changing Requirements:**

Risk: When requirements change at some degree inside the development manner, it may result in misalignment between the code and the take a look at times.

Example: If the requirement for a are trying to find capability modifications from a smooth key-phrase search to a complicated seek with filters, modern take a look at instances might not cover the brand new functionality.

- **Lack of Test Data:**

Risk: Inadequate or wrong test statistics can lead to insufficient check coverage or fake positives/negatives.

Example: In an e-exchange software, if the trying out group quality makes use of valid credit score rating card numbers for price attempting out, they might miss situations associated with invalid card numbers.

- **Time Constraints:**

Risk: Tight assignment timelines can lead to rushed trying out, probably lacking crucial defects.

Example: In a task with a good closing date, the checking out group may not have sufficient time to thoroughly take a look at all factors of the software, leading to a higher threat of undetected issues.

- **Dependency on External Systems:**

Risk: Testing that includes integration with outside structures (e.G., APIs, databases) can be affected by the supply and stability of those structures.

Example: If a fee gateway is down in some unspecified time in the future of testing, it can save you thorough testing of the price method.

- **Limited Testing Environments:**

Risk: Having confined or poorly configured checking out environments can motive incomplete or erroneous attempting out results.

Example: if the sorting out surroundings would not efficiently discover mirror the manufacturing surroundings, some defects might also satisfactory after deployment.

- **Insufficient Test Coverage:**

Risk: Failing to cover all critical conditions can cause undetected defects in manufacturing.

Example: If finding out focuses only on top notch situations (e.g., legitimate inputs), it would omit thing times or terrible eventualities (e.g., invalid inputs) that would result in crucial problems.

- **Human Error in Testing:**

Risk: Mistakes made with the aid of attempting out organizations, which incorporates incorrect take a look at information or overlooking check cases, can result in faulty outcomes.

Example: A tester via twist of fate makes use of manufacturing information in preference to test information, leading to unintended consequences or facts loss.

- **Regression Test Suite Management:**

Risk: Managing a large suite of regression checks can be difficult, possibly primary to inefficient trying out or omitted regression troubles.

Example: If regression tests are not nicely prepared or maintained, it may be tough to pick out which tests want to be completed after code changes.

Q.2.3

- **Error:**

An error is a mistake made through a human at some point of the development technique. It is a deviation from the real and predicted conduct of a software. Errors are brought because of elements, which incorporates misconception requirements, typos, logical mistakes, or wrong implementation of algorithms.

Example of an Error:

Suppose a developer is operating on a utility to calculate the average of a listing of numbers. However, due to a typo inside the code, they by way of chance use addition in region of department even as calculating the not unusual. This mistake is an error.

- **Defect:**

A illness, moreover known as a computer virus, is a flaw or fault in a software program that causes it to act in an unintentional or faulty way. Defects stand up because of errors made in some

unspecified time in the future of the development method. They can range from minor issues to important issues, which have an effect on the capability, standard performance, or safety of the software application.

Example of a Defect:

Using the same instance as above, if the code consists of the error said, it will bring about a contamination within the software. When completed, it will produce a wrong stop result for the average.

Reference: Hove, EH. (2023) ‘Importance of Software testing’, *SQAT6322: Software and Quality Testing.* Rosebank College. 02 October.

Reference: Hove, EH. (2023) ‘Testing phases’, *SQAT6322: Software and Quality Testing.* Rosebank College. 02 October.

Reference: Hove, EH. (2023) ‘The testing workbench process’, *SQAT6322: Software and Quality Testing.* Rosebank College. 02 October.

Reference: Hove, EH. (2023) Capability Maturity Model Integration (CMMI), *SQAT6322: Software and Quality Testing.* Rosebank College. 02 October.