1. What is the primary purpose of Quality Assurance (QA)?
The primary purpose of Quality Assurance (QA) is to ensure that a product or service meets established quality requirements and standards before it is delivered to the end user. QA aims to prevent defects and issues through systematic processes and practices, ensuring that the final product is reliable, functional, and free of significant problems. Essentially, QA focuses on improving and stabilizing development and production processes to enhance overall product quality.

2. What is the difference between a test case and a test plan?

Test Case can be definition with a detailed document that outlines specific conditions, inputs, actions, and expected results to verify that a particular feature or functionality of the software behaves as expected. Includes test case ID, description, preconditions, test steps, expected results, and postconditions.

> Example: For a login feature, a test case might involve entering valid credentials and checking if the user is redirected to the dashboard.

Test Plan can be definition: A test plan is a high-level document that defines the overall strategy, scope, resources, schedule, and approach for testing a software application. Includes scope, objectives, testing strategy, resource requirements, schedule, deliverables, and risk assessment.

> Example: A test plan for a project might detail the testing types (e.g., functional, performance, security), the testing tools and environments, the testing schedule, and roles and responsibilities.

In summary, test cases are used to validate individual functionalities, while a test plan provides a strategic overview of how testing will be conducted for the entire project.

3. What does the term 'regression testing' refer to?
Regression testing refers to a type of software testing that ensures that recent changes or updates to the codebase have not adversely affected the existing functionality of the application. The primary goal is to confirm that previously developed and tested software still performs after a change.

4. How would you create a template to describe bugs on a software that is being constantly released?
Creating a template to describe bugs for a software that is being constantly released involves ensuring that the template captures all necessary details for effective tracking and resolution of issues. Here’s a comprehensive template you can use:

Bug Report Template

---

1. Title
    - *Short and descriptive title of the bug*

2. Description
    - *Detailed description of the bug, including what was expected versus what actually occurred*

3. Steps to Reproduce
    1. *Step-by-step instructions to reproduce the bug*
    2. *Additional steps if needed*

4. Expected Result
    - *What should have happened if the bug did not exist*

5. Actual Result
    - *What actually happened when the bug was encountered*

6. Severity
    - *Severity level of the bug (e.g., Minor, Major, Critical)*

7. Priority
    - *Priority level for fixing the bug (e.g., Low, Medium, High)*

8. Environment
    - *Details about the environment where the bug was found (e.g., OS version, browser version, device type)*

9. Build/Version
    - *Specific build or version of the software where the bug was identified*

10. Attachments
    - *Screenshots, videos, or logs that help illustrate the bug*

11. Resolution
    - *Description of how the bug was fixed or why it was closed*

12. Verification
    - *Confirmation that the fix has been verified and the bug is resolved*

---

5. What is the importance of test automation in QA? What would you explore in terms of automation for games being developed in a short period of time?

Test automation plays a crucial role in Quality Assurance (QA) for several reasons:

1. Efficiency: Automated tests can run quickly and repeatedly, significantly reducing the time needed for testing compared to manual testing.
2. Consistency: Automation ensures that tests are executed in the same manner every time, eliminating human errors and inconsistencies.
3. Reusability: Once created, automated tests can be reused across different versions of the software, making them cost-effective in the long run. This is particularly useful for regression testing.
4. Speed of Feedback: Automation provides faster feedback on code changes, enabling developers to identify and fix issues earlier in the development cycle.
5. Continuous Integration/Continuous Deployment (CI/CD): Automation is integral to CI/CD pipelines, allowing for automated testing with every code change, build, or deployment, thus supporting rapid development and release cycles.

For games developed quickly, test automation can be especially beneficial. Here's how you might explore and implement automation in this context:

1. Define Key Areas for Automation:
    - Core Gameplay Mechanics: Automate testing of core game functionalities, such as movement, interactions, and scoring systems.
    - Regression Testing: Create automated tests to ensure that new updates do not break existing game features.
    - User Interface (UI) Testing: Automate tests for various UI elements to verify that menus, buttons, and other interface components function correctly.
    - Performance Testing: Automate performance tests to ensure the game runs smoothly under different conditions and loads.

2. Integrate with Development Workflow:
    - CI/CD Pipelines: Incorporate automated tests into your CI/CD pipelines to run tests automatically with every build or deployment.
    - Frequent Builds: Ensure that your automation framework can handle frequent builds and updates, providing quick feedback to developers.

3. Monitor and Report Results:
    - Reporting Tools: Utilize reporting tools to analyze test results, track issues, and generate reports on test coverage and outcomes.

4. Adapt and Evolve:
    - Flexibility: Be prepared to adapt and update your automation approach as the game evolves, ensuring that automated tests remain relevant and effective.

6. Describe the main stages of the Software Testing Life Cycle (STLC).
The Software Testing Life Cycle (STLC) encompasses the various stages involved in the testing process of a software product.

1. Requirement Analysis
    - Objective: Understand and analyze the requirements of the software to determine the testable aspects.
    - Activities: Review requirement documents, identify testable requirements, and clarify any ambiguities.
2. Test Planning
    - Objective: Define the scope, approach, resources, and schedule for the testing activities.
    - Activities: Develop the Test Plan, identify testing objectives, scope, strategies, resource requirements, timelines, and risk management.
3. Test Design
    - Objective: Create detailed test cases and test scripts based on the requirements and test plan.
    - Activities: Design test cases, create test data, and prepare test scripts for manual or automated testing.
4. Test Execution
    - Objective: Execute the test cases and scripts to validate that the software meets the specified requirements.
    - Activities: Run the test cases, record the test results, and log any defects found during testing.
5. Defect Reporting and Management
    - Objective: Identify, report, and manage defects found during the test execution stage.
    - Activities: Report defects, track defect status, prioritize and categorize defects, and work with development teams to resolve them.
6. Test Closure
    - Objective: Finalize and close the testing process by evaluating test coverage and preparing for the release.
    - Activities: Conduct test summary meetings, evaluate test results, ensure all defects are addressed, and prepare final test reports.
7. Test Review and Retrospective
    - Objective: Review the testing process and outcomes to identify lessons learned and areas for improvement.
    - Activities: Conduct retrospectives, review the test process, and gather feedback from stakeholders.

7. How would you handle a situation where a bug you reported is marked as 'not reproducible' by the development team?
Initially, I would specify the details, such as whether the bug stopped occurring because of a new release. For example, if I test again and the bug is no longer present in the new release, I would be satisfied. However, if I verify that there has been no new release or development and the bug

persists, I would retest the issue and discuss with the person who closed the bug to consider reopening the issue.

General Overview

○ **Even though this is not a question, we'd like to hear why you selected this game.**
What are the ups and downs?
Because this game have a great scenario more dark, and the skulls remember for me the mexicans skulls. And like this diferents things for my culture, are interesting.

○ **What would you consider a factor that would make you play again?**
I would like to play this game again for the chance to win a little money

○ **Was this an engaging experience?**
At first, I was a little confused because I didn't know what to do, but after clicking on the buttons, I figured it out and had fun afterward