Test Case Testing vs. Exploratory Testing

Testing can take multiple forms, but the goal is always the same: to find bugs and flaws quickly.

One approach is to create test cases, where you can use automated tests or real people — professional testers — to engage with your software, providing them with a specific test script to follow.

Another is to have testers perform exploratory testing, which is generally less structured — there's no test script to follow, but rather, a simple set of instructions. However, since there are less specific instructions, there's also more room for creativity.

Test case testing is great for ascertaining if your software is working as predicted. Throughout the software development life cycle, you will no doubt run multiple test cases. As mentioned, test case testing can be an automated process, as a program can run through the script you provide, or you can have crowdtesters follow your test scripts. But relying on this type of testing alone might mean some bugs go undetected because this approach is only checking for flaws you already think might exist.

An automated testing program with test cases can run from point A to B to C, and let you know if bugs are detected, but real users don't always move from point A to B to C. Humans are unpredictable and don't always operate linearly. Exploratory testing can provide answers to questions you didn't even know you had and can find problems you didn't imagine existed by bridging the gaps left by scripted test cases.

Test Case Testing

- Directed from requirements
- Test cases created in advance
- Confirmation of testing requirements
- Emphasizes prediction
- Like giving a prepared speech; it's drafted
- The script is in control

VS

Exploratory Testing

- Min. requirements; focuses on exploration
- Test cases created during testing
- Investigation of system application
- Emphasizes adaptability and learning
- Like having a conversation; it's spontaneous
- The tester's mind is in control