Assignment 4

- Software Engineering
- Professor: Aldamuratov J.U.
- Kakharmanova Aruzhan

KBTU 2023

Task 1

Chapter 8. Exercise 8.5

1. What is regression testing?

Regression testing is a testing of the software's which undergo many changes, validating whether the functionalities of the components of software remains unchanged. It involves validating whether changes, such as code modifications, bug fixes, or new feature additions, adversely impact existing functionalities. Regression testing is is a critical aspect of software testing that aims to ensure the stability and reliability of a software application over time.

The purpose of regression testing is to verify that alterations to the software do not break or compromise existing functionalities. This type of testing is crucial for maintaining the integrity of the software throughout its development and maintenance lifecycle. Test cases for regression testing are typically derived from existing test suites, covering a broad spectrum of the application's functionalities.

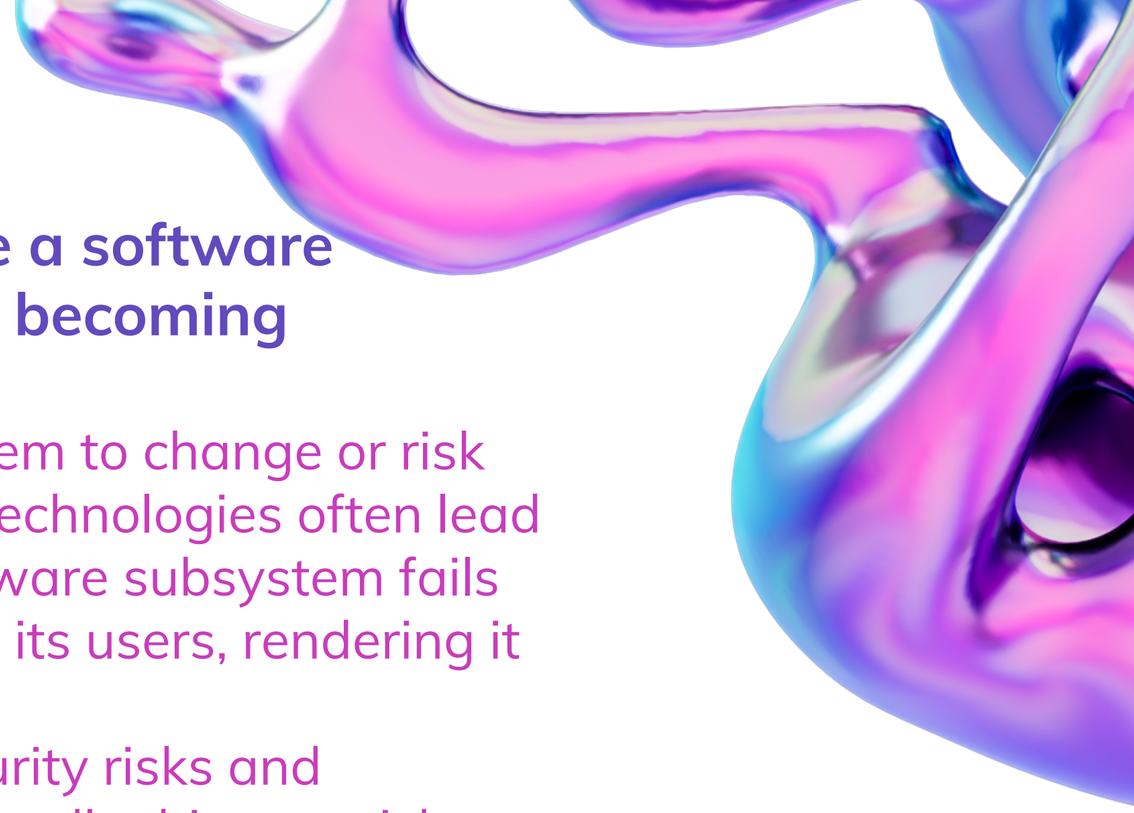
2. Explain how the use of automated tests and a testing framework such as JUnit simplifies regression testing.

Having automation frameworks, it's easy to perform testing on the software's where test suite runs in a very small time on all the functionalities of the software including the newly added changes. Cost of regression testing is reduced by using automation tests. Automation makes the regression testing more easier and reduces effort.

In essence, the combination of automated tests and a testing framework like JUnit simplifies regression testing by accelerating test execution, promoting reusability, and offering timely feedback on code changes. This approach not only enhances the overall efficiency of the testing process but also contributes to the reliability and robustness of the software being developed.

Task 2

- **Chapter 9. Exercise 9.1**
  - Explain how advances in technology can force a software subsystem to undergo change or run the risk of becoming useless?



Technological advances can force the software subsystem to change or risk becoming obsolete for a number of reasons. First, new technologies often lead to changes in user expectations and behaviors. If a software subsystem fails to adapt to these changes, it may not meet the needs of its users, rendering it obsolete.
Second, technological advances can introduce new security risks and vulnerabilities. A software subsystem not designed to handle this new risk could be liable, potentially leading to compromise of sensitive data.
Furthermore, technological advances tend to improve productivity and productivity. A software subsystem that does not take advantage of these enhancements can be slow and inefficient compared to alternatives, making them unattractive to users.
Finally, other systems of collaboration are necessary in the rapidly developing technological environment. A software subsystem that does not integrate well with new technologies can be isolated and incompatible with other systems, limiting its usefulness

Thank you for attention!