****

**Al-Isra University**

**Faculty of IT**

**Department of Computer Science**

**جــامـعـة الإسراء**

**كـليـة تكـنولوجيـا المعلومـات**

**قـســم هندسة البرمجيات**

|  |  |  |
| --- | --- | --- |
| **Department:**  SE | **Assignment:**  **#1** | **Submission Deadline:** |
| **Semester:**  Second | **Year**:  2021/2022 | **Instructor**  Dr. Dimah Fraihat |
| **Course No.:** | **Course Name:**  Software Testing | **Section:**  1 |
| **Student No.:**  **AB1199** | **Student Name:**  **حمزة خالد مسلم** | **Submission date:**  30/04/2024 |

|  |  |  |
| --- | --- | --- |
| Question No. | Mark | |
| Max | Score |
| 1 | **5** |  |
| **Total Mark** |  |  |

**Note: 1- Copy and paste are prohibited**

**2- Don’t copy and paste from the Internet resources**

**Q#** #Differentiate (in details) between STATIC and DYNAMIC verification in software testing.

|  |  |  |
| --- | --- | --- |
|  | Static Verification | Dynamic Verification |
| Definition | * Static verification involves examining the software code, design, or documentation without executing the program. It focuses on identifying defects, errors, or anomalies in the artifacts themselves rather than observing the behavior of the running software. | * Dynamic verification involves executing the software and observing its behavior to ensure that it meets specified requirements and functions correctly. It focuses on testing the software's runtime behavior and performance. |
| Methods | * Code Reviews: Developers or peers examine the code line by line to identify issues such as logic errors, syntax mistakes, or coding standards violations. * Inspections: Formalized meetings where a group of stakeholders review software artifacts like requirements, design documents, or test plans to uncover defects and ensure compliance with standards. * Walkthroughs: Similar to inspections, but less formal, walkthroughs involve a guided tour of the software artifacts to gather feedback and identify potential issues. * Static Analysis Tools: Automated tools that analyze code or documentation to detect issues such as code smells, potential security vulnerabilities, or adherence to coding standards. | * Unit Testing: Tests individual units or components of the software in isolation to verify their functionality and behavior. * Integration Testing: Tests the interactions between different units or modules to ensure they work together as intended. * System Testing: Tests the entire system as a whole to verify that it meets specified requirements and functions correctly in its intended environment. * Acceptance Testing: Tests conducted by end-users or stakeholders to validate that the software meets their expectations and requirements. |
| Advantages | * Early Detection of Defects: Issues can be identified and addressed before the software is even executed, saving time and effort in later stages. * Cost-Effectiveness: Static verification methods are generally less resource-intensive compared to dynamic testing approaches. * Improved Quality: By catching errors early, static verification helps improve the overall quality and reliability of the software. | * Realistic Assessment: Dynamic testing provides a more realistic assessment of the software's behavior under actual operating conditions. * Comprehensive Coverage: Dynamic testing methods can uncover a wide range of defects, including runtime errors, logic flaws, and performance issues. * Feedback Loop: Test results provide valuable feedback to developers, helping them identify and fix issues promptly. |
| Limitations | * Limited Scope: Static verification methods may not uncover all types of defects, especially those related to runtime behavior or integration issues. * Human Error: The effectiveness of static methods relies on the expertise and diligence of the individuals performing the reviews, inspections, or analyses. * Time-Consuming: Formal inspections or code reviews can be time-consuming, especially for large or complex software projects. | * Late Detection of Defects: Dynamic testing typically occurs after the software has been developed, making it more costly and time-consuming to address defects discovered during testing. * Resource Intensive: Dynamic testing requires resources such as test environments, test data, and time for execution, which can be significant for large or complex systems. * Limited Scope: It may be challenging to achieve complete test coverage, especially for systems with intricate logic or complex interactions. |