The Cyclomatic Complexity is commonly considered in modules on testing the validity of code design today. However, in your opinion, should it be? Does it remain relevant today? Specific to the focus of this module, is it relevant in our quest to develop secure software? Justify all opinions which support your argument and share your responses with your team.

Cyclomatic complexity remains relevant today for multiple reasons:

1. Cyclomatic complexity directly impacts the ability to identify and test security vulnerabilities.

Complex code with lengthy decision points is more prone to errors and as such poses a significant security risk (McCabe, n.d.). Attackers can pursue specific edge cases within the decision trees, and exploit interdependent relationships which may result in unexpected behaviour. Code with high cyclomatic complexity requires consideration of additional testing methods, such as basis path testing (GeeksForGeeks, 2022) and ensuring test coverage is sufficient.

2. It ensures software quality and maintainability

High cyclomatic complexity risks affecting the efficiency of development teams, as the code base can grow to become unmanageable and more tests are required to be written and potentially re-written (Sonar, n.d.). Interestingly, Ebert et al. (2016) highlight how, despite the metric being challenged often in academic circles, it remains highly popular within the industry.

While cyclomatic complexity is not a perfect metric, its ability to quantify code complexity, guide testing efforts, and identify potential security vulnerabilities makes it an invaluable tool in modern secure software development. Its mathematical foundation

and practical applications continue to provide relevant insights for developing and maintaining secure, high-quality software systems to this day.

References:

GeeksForGeeks (2022) Basis Path Testing in Software Testing. GeeksForGeeks. Available from: https://www.geeksforgeeks.org/basis-path-testing-in-software-testing/ [Accessed 22 November 2024]

McCabe (n.d) Using Cyclomatic Path Analysis to Detect Security Vulnerabilities. McCabe Software. Available from: http://www.mccabe.com/pdf/Cyclomatic_Security_Vulnerability_Detection.pdf [Accessed 22 November 2024]

Sonar (n.d.) cyclomatic complexity: developer's guide. Sonar Source. Available from: https://www.sonarsource.com/learn/cyclomatic-complexity/#how-to-test-cyclomatic-complexity [Accessed 22 November 2024]