Module One

Learning Objectives

By the end of this module, you will meet these learning objectives:

- Describe software techniques, tools, and applications
- Describe the role of testing in the software development life cycle

Module Overview

Welcome to CS 320: Software Testing, Automation, and Quality Assurance! In this course, we will explore multiple testing techniques and their uses. Starting in Module Three, we will focus on gaining hands-on experience with JUnit testing, a unit testing framework for the Java programming language. You will use the JUnit skills you build in Modules Three, Four, and Five to test your final project software application in Module Six. You will also submit your summary and reflection report in Module Seven.

Now, let's dive into our introduction to software testing. Why is software testing critical to the success of a software project? Software testing is the most expensive process in the software life cycle. Catching defects early in the software life cycle will save the project time and money compared with finding defects later. This is because it is easier to resolve defects while the code is being written. In addition, the lack of software testing can result in costly software rewrites and increased technical debt, a concept in programming that reflects the added cost of development work when software developers fail to create modular, maintainable, readable, and testable code the first time through. Software testing impacts the software product and the entire team. If another team member needs to make changes to your code and there is not sufficient testing, your code may be rewritten or modified to support testing, adding again to this concept of technical debt. The goal of this course is to provide you with new software testing skills and also an awareness of soft skills that are important for a software developer to have, which include this idea of avoiding technical debt.

Software quality and testing differ across organizations, depending on the company and the software product. For example, avionic software must go through a rigorous verification and validation process. Verification is software testing conducted to ensure that you have developed or met the software requirements of what you said you were going to develop, and validation software testing is testing done to ensure that you have correctly built what you said you were going to build and that customers receive what they want. We will talk about this again in the next module. Avionic software

is tested at the unit level—integration and system testing. We will go into the specifics of these testing techniques in the next module. Once avionic software is accepted, it is locked down and cannot be changed without the customer's approval.

In addition, in this module, you will dive into software testing philosophies and basic concepts. You will explore why software testing is important and how it fits in the software life cycle by responding to the discussion prompt and to other students. You will also access your course tools and complete your final project review.

Module at a Glance

This is the recommended plan for completing the reading assignments and activities within the module. Additional information can be found in the module Resources section and on the module table of contents page.

1. Review the Module One resources.
2. Post your initial response to this week's discussion (initial post due Thursday).
3. Access the course tools and software. Reach out to your instructor immediately if you have any issues.
4. Complete the Module One journal.
5. Review the projects for this course.
6. Post peer responses to the discussion.