Preview: IT4776 : Software Testing, Security and Quality Assurance

Syllabus

Course Overview

In this course, you will learn and practice the fundamentals of software testing. You will learn the importance of quality assurance of software systems and the role that testing plays in this assurance. You will examine both static and dynamic testing techniques. You will study and develop the different types of tests and align them with the proper phase of developing software using test levels. You will also apply many testing tools that are commonly used in the testing process.

The course will progress as follows:

- **Unit 1: Software Testing and Quality Assurance.**
  - Learn about the different ways to ensure the quality of software including testing. You will take a quiz that assesses your understanding of the various concepts of the software testing process.
- **Unit 2: Software Development Models and Their Test Levels.**
  - Explore the relationship between the testing process and the software development process. You will learn the various test levels and how they are applied during the testing process. There is no assignment for Unit 2.
- **Unit 3: Test Types.**
  - Learn about the different test types and its objectives. It is built on what you have learned about the testing process and its activities with test types. You will develop a video presentation about the test process, test levels, and test types.
- **Unit 4: Application and Code Security.**
  - Examine the testing process for Java application and code security. You will examine the various rules of application and code security in Java and how they are applied. You will analyze some of these rules and provide examples of compliant and non-compliant Java code.
- **Unit 5: Static Testing Techniques.**
  - Learn about the review process and the static code analysis by tools. You will statically analyze Java code using the NetBeans IDE Inspector tool.
- **Unit 6: Dynamic Test Techniques.**
  - Investigate the experience-based testing techniques and how to conduct them. There is no assignment for Unit 6.
- **Unit 7: White-Box Testing.**
  - Learn the white-box testing techniques and ways to design a test case for this type of testing. It is built on what you have learned about dynamic testing techniques. In Unit 7, you will design component level test cases using white-box techniques.

- **Unit 8: Black-Box Testing.**
    - Conclude with the study of dynamic testing techniques. In Unit 8, you will learn the black-box techniques and how they are carried out using tools. You will dynamically black-box test a component using NetBeans IDE JUnit.
- **Unit 9: Test Planning and Estimation.**
    - Understand the various aspects of test management like test planning and test estimation. There is no assignment for Unit 9.
- **Unit 10: Tools for Software Testing.**
    - Explore the role that tools play in the testing process. You will learn about the various types of tools and their purposes in the testing process. You will write a paper about four of these tools to analyze their usage, benefits, and risks. The paper should be formatted according to APA style and format.

Hardware

Capella University requires learners to meet certain minimum computer requirements. The following hardware may go beyond those minimums and is required to complete learning activities in this course. Headsets and webcams are available for purchase at the Capella University Bookstore. Refer to the manufacturer's directions for installing and connecting the devices to your computer. Note: If you already have the following hardware, you do not need to purchase it.

Narrated PowerPoint Presentation.

1. External or built-in webcam.
2. Headset or device for recording in from of others.
3. Broadband Internet connection.

Kaltura Media

As part of this course, you are required to record video presentations using Kaltura Media or similar software. Refer to Using Kaltura [PDF] for more information about this courseroom tool.

Note: If you require the use of assistive technology or alternative communication methods to participate in these activities, please contact Disability Services to request accommodations.

Course Competencies

(Read Only)

To successfully complete this course, you will be expected to:

1 Apply appropriate testing terminology during the software testing process.

2. Create well-defined testing objectives and targets.
3. Prepare test cases that accurately addresses the software testing objectives.
4. Utilize testing tools in software testing activities.
5. Evaluate application and code security.
6. Perform a complete testing process taking into account practical considerations.
7. Write clear and focused test cases.

Course Prerequisites

IT4772

Syllabus >> Course Materials

Required

The materials listed below are required to complete the learning activities in this course.

Integrated Materials

Many of your required books are available via the VitalSource Bookshelf link in the courseroom, located in your Course Tools. Registered learners in a Resource Kit program can access these materials using the courseroom link on the Friday before the course start date. Some materials are available only in hard-copy format or by using an access code. For these materials, you will receive an email with further instructions for access. Visit the Course Materials page on Campus for more information.

Book

Black, R., Veenendaal, E. V., & Graham, D. (2012). *Foundations of software testing ISTQB certification* (3rd ed.). Boston, MA: Cengage. ISBN: 9781408044056.

Hardware

Capella University requires learners to meet certain minimum computer requirements. The following hardware may go beyond those minimums and is required to complete learning activities in this course. **Note:** If you already have the following hardware, you do not need to purchase it. Visit the Course Materials page on Campus for more information.

Presentation Hardware

- External or built-in microphone
- Headset with microphone
- Broadband Internet connection

Library

The following required readings are provided in the Capella University Library or linked directly in this course. To find specific readings by journal or book title, use Journal and Book Locator. Refer to the Journal and Book Locator library guide to learn how to use this tool.

- Black, R. (2007). *Pragmatic software testing: Becoming an effective and efficient test professional.* New York, NY: Wiley.
- Chemuturi, M. (2010). *Mastering software quality assurance: Best practices, tools and techniques for software developers.* Plantation, FL: J. Ross.
- McGraw, G., & Felten, E. (1998). Twelve rules for developing more secure Java code: Writing security-conscious Java code can help you avoid security surprises. *JavaWorld.*

External Resource

Please note that URLs change frequently. While the URLs were current when this course was designed, some may no longer be valid. If you cannot access a specific link, contact your instructor for an alternative URL. Permissions for the following links have been either granted or deemed appropriate for educational use at the time of course publication.

- Carnegie Mellon University. (n.d.). SEI CERT oracle coding standard for Java. Retrieved from https://www.securecoding.cert.org/confluence/display/java/SEI+CERT+Oracle+Coding+Standard+for+Java

- Hayden, J. (n.d.). A guide to robust unit and integration tests with JUnit. Retrieved from https://www.toptal.com/java/getting-started-with-junit
- JUnit.org. (n.d.). JUnit. Retrieved from http://junit.org/junit4/
- Lahoda, J., & Stashkova, A. (n.d.). Static code analysis in the NetBeans IDE Java Editor. Retrieved from https://netbeans.org/kb/docs/java/code-inspect.html
- NetBeans. (n.d.). Ada for NetBeans - plugin detail. Available from http://plugins.netbeans.org/plugin/13977/ada-for-netbeans
- NetBeans. (n.d.). Java hints. Retrieved from http //wiki.netbeans.org/Java_Hints
- NetBeans. (n.d.). Java SE (8th ed.). Available from http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html
- NetBeans. (n.d.). NetBeans IDE (8th ed.). Available from https://netbeans.org/downloads/
- NetBeans. (n.d.). Writing JUnit tests in NetBeans IDE. Retrieved from https://netbeans.org/kb/docs/java/junit-intro.html
- NetBeansVideos. (2012). Static analysis of Java code in NetBeans IDE. Retrieved from https://www.youtube.com/watch?v=WZHaCjaBPG8
- NetBeansVideos. (2014). New in NetBeans IDE 8.0.1: FindBugs 3.0 for Java 8. Retrieved from https://www.youtube.com/watch?v=bcdvgkXZXHg&feature=youtu.be
- Oracle. (n.d.). Secure coding guidelines for Java SE. Retrieved from http://www.oracle.com/technetwork/java/seccodeguide-139067.html
- Retterer, D. (2016). JUnit with NetBeans introduction. Retrieved from https://www.youtube.com/watch?v=4CehzWlpV48&feature=youtu.be
- University of Maryland. (n.d.). FindBugs - Find bugs in Java programs. Retrieved from http://findbugs.sourceforge.net/

Suggested

Optional

Unit 1 >> Software Testing and Quality Assurance

Introduction

In this first unit of the course you will learn about the following:

- Software quality assurance and the various ways this assurance can be achieved.
- Software testing and its role as a primary means to ensure the quality of software.
- Various concepts and terms that are used in the test process and the testing principles that guide their use.

You will then take a quiz that assesses your understanding of the various concepts of the software testing process.

Learning Activities

u01s1 - Studies

Required Reading

Regarding software testing and how it contributes to software quality assurance:

Note: Please note that this material is dense**.** You should expect to read and re-read this content, and reference it carefully as you work on this unit's assignment.

- Black, R., Veenendaal, E. V., & Graham, D. (2012). *Foundations of software testing ISTQB certification* (3rd ed.). Boston, MA: Cengage.
  - Chapter 1, "Fundamentals of Testing," pages 1–18 (Sections 1–4.)

Regarding quality and quality assurance:

- Chemuturi, M. (2010). *Mastering software quality assurance: Best practices, tools and techniques for software developers.* Plantation, FL: J. Ross.
  - Chapter 1, "Quality Assurance Basics," pages 1–24.

u01s1 - Learning Components

- Understand the key terms of software testing.
- Understand the key terms of software quality assurance.
- Explain the role of testing in software quality assurance.

u01s2 - Software

A Note About the Software for This Course

You can download both Java and NetBeans for free, directly to your computer. They are available on the Internet. The links are provided below.

There is a NetBeans plug-in available for ADA accessibility purposes as well; however, this plug-in is for an older version of NetBeans (6.9). Please let the instructor know if you are using an older version.

Use of the following tools is strongly recommended for completing the course objectives successfully. If you have access to other tools that you believe may meet the requirements of this course, please discuss your selected alternative with your instructor.

Note: If you use assistive technology or any alternative communication methods to access course content, please contact Disability Services with any access-related questions or to request accommodations.

- NetBeans. (n.d.). Java SE (8th ed.). Available from http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html
- NetBeans. (n.d.). NetBeans IDE (8th ed.). Available from https://netbeans.org/downloads/
- NetBeans. (n.d.). Ada for NetBeans - plugin detail. Available from http://plugins.netbeans.org/plugin/13977/ada-for-netbeans

u01q1 - Vocabulary Quiz

Overview

The software testing, security, and quality assurance field are rich in its concepts and terminology. Understanding these concepts and communicating them using the correct terminology is mandatory if you want to succeed in this field.

Use this quiz to test your understanding of some of the fundamental concepts and terms of software testing, security, and quality assurance. Please note that in the later assignments of this course you will again be evaluated on how well you have mastered them.

Instructions

Answer the eight questions in this vocabulary quiz. All content needed to answer the quiz questions can be found in this unit's studies.

You have one attempt at this quiz, and there is a one-hour time limit.

u01d1 - Evaluating Three Software Quality Assurance Techniques and Their Applications

Software testing plays an important role in improving the quality of executing software. But software testing is not the only means of assuring the quality of the developed software. In this discussion, you will examine testing, as well as two other means of software quality assurance, and how they are applied.

- Review the resources in the studies of this unit.
- Describe three software testing techniques (needed to assess the quality of software.)

    - Explain the context in which each technique is applied.
    - Explain the role that each technique plays to ensure the quality of the developed software.

Response Guidelines

Write at least two response posts. In your posts, you might make a comparison, add information, ask questions, or respond in any way that will help your fellow learners.

Course Resources

Undergraduate Discussion Participation Scoring Guide

Unit 2 >> Software Development Models and Their Test Levels

Introduction

You will learn about the following in this unit:

- Various models of developing software and their phases.
- How the test process relates to the various phases of developing software via test levels.

There is no assignment for Unit 2. You are encouraged to read the instructions for the Unit 3 assignment, as you may wish to begin this assignment early.

Learning Activities

u02s1 - Studies

Required Readings

Regarding the psychology and ethics of software testing:

- Black, R., Veenendaal, E. V., & Graham, D. (2012). *Foundations of software testing ISTQB certification* (3rd ed.). Boston, MA: Cengage.
    - Chapter 1, "Fundamentals of Testing," pages 18–26 (Sections 5 and 6.)

Regarding the relationship between the software development lifecycle, software testing, and test levels:

Note: Please note that this material is dense. You should expect to read and re-read this content.

- Black, R., Veenendaal, E. V., & Graham, D. (2012). *Foundations of software testing ISTQB certification* (3rd ed.). Boston, MA: Cengage.
    - Chapter 2, "Testing Throughout the Software Life Cycle," pages 27–39 (Sections 1 and 2.)

u02s1 - Learning Components

- Understand what is meant by software test levels.
- Understand what is meant by software test types.

u02s2 - Assignment Preparation for Unit 3

You do not have an assignment in this unit. However, in the next unit, you will record and submit a presentation on test processes, test levels, and test types using Kaltura Media or similar software. Refer to the Using Kaltura [PDF] for directions on recording and submitting your presentation. We recommend reading the instructions for this assignment early because it involves the use of recording technology (such as Kaltura) as well as the mastery of course content.

Note: If you require the use of assistive technology or alternative communication methods to participate in this activity, please contact Disability Services to request accommodations.

Recommended Reading

- Using Kaltura [PDF].
- Unit 3 assignment instructions.

u02d1 - Connecting Test Activities With Software Development Activities

All models of software development incorporate testing activities as a part of their development life cycle. In this discussion, you will explain how the testing activities are related to the activities of a software development model of your choosing.

Choose a software development model like agile, waterfall, rapid application development, or any other model and describe its core activities.

- Explain how the development activities and the testing activities are related to your chosen software development model.
- Determine the levels of the testing activities of your chosen software development model.

Response Guidelines

Write at least two response posts. In your posts, you might make a comparison, add information, ask questions, or respond in any way that will help your fellow learners.

Course Resources

Undergraduate Discussion Participation Scoring Guide

Unit 3 >> Test Types

Introduction

In this unit you will explore the following:

- Various software test types, especially black-box and white-box.
- Objective of each type of tests.
- Test types that relate to test levels.

For the assignment in this unit, you will develop a video presentation about the test process, levels, and types.

Learning Activities

u03s1 - Studies

Required Readings

Regarding the different types of test and their objectives.

- Black, R., Veenendaal, E. V., & Graham, D. (2012). *Foundations of software testing ISTQB certification* (3rd ed.). Boston, MA: Cengage.
  - Chapter 2, "Testing Throughout the Software Life Cycle," pages 40–49 (Sections 3 and 4.)

Multimedia

- Click **Kaltura Basics Tutorial** to view the interactive.
  - This may be useful to you as you record your presentation for this unit's assignment.

Course Resources

Kaltura Basics Tutorial

u03s1 - Learning Components

- Explain the relationship between the software test process, levels, and types.

u03a1 - Presenting Test Processes, Levels, and Types

Overview

The software test process involves activities that carry out different types of tests at various levels of the software development life cycle.

In this assignment, imagine yourself as a software test engineer who has been asked make a presentation on the test process via a video (visual and narration.) Your presentation should include the various test types and test levels used within the process.

Imagine that your audience for this presentation is the senior management team of your organization and they have requested for a time limit of 15 minutes. They are considering whether or not they will establish a separate test unit within the organization. Your objective is to help the senior management understand the field of software testing and its main concepts. You do not need to persuade these senior managers in their decision about the separate test unit. Your task with this presentation is to cover all key content effectively within 15 minutes or less.

Design this video according to professional standards, and produce it according to the requirements listed below.

Preparation

- Content: To prepare for this assignment, review the textbook readings for this unit, especially pages 27–44.
- Producing the video: We recommend using Kaltura to create your video. To familiarize yourself with this tool:
  - Visit the Using Kaltura [PDF]. Please note that even if you use a different recording tool to create your video, you must use Kaltura to upload the video file into the courseroom. The Using Kaltura [PDF] includes instructions for uploading video and the supported file types.
  - View the interactive *Kaltura Basics Tutorial* linked in the Resources.

Directions

- Use Kaltura or a different video recording tool of your choice to design and produce a video presentation:
  - **Content requirements:**
    - Describe the main steps in the software test process.
    - Explain the test levels.
    - Explain the test types.
    - Explain the relationship between the test process, levels, and types.
  - **Video production requirements:**

        - 10 to 15 minutes is the time limit for the video. Your audience will cut you off at the 15-minute mark. Ensure that you cover all key materials concisely and clearly in the presentation.
        - Record your own voice narrating the video and include a visual component. You may record a visual of yourself speaking or create a slideshow on your computer and display the slides as you talk.
        - Edit your video. The video presentation should be well-designed, adequately produced, and clearly viewed. If you use a video production tool other than Kaltura, make sure that your video is of similar or higher quality. The Using Kaltura [PDF] includes recommendations for editing. Note the following:
            - Use the **Trim** function to edit out unnecessary content.
            - Create a title and description slide at the start of the video.
            - Add a slide with the credits at the end of the video if you have referenced any resources in the presentation.
        - Submit your video. If you have recorded your video with a tool other than Kaltura, you will still need to use Kaltura to upload the video to the courseroom. The Using Kaltura [PDF] explains this process.
    - **Transcript requirement:**
        - Include a written transcript of the spoken part of your presentation. The transcript can be a Word document or a PDF. It must be submitted alongside the presentation.

Refer to the Using Kaltura [PDF] for directions on recording or uploading a video.

Note: If you require the use of assistive technology or alternative communication methods to participate in these activities, please contact Disability Services to request accommodations.

Submission Requirements

- **Communication:** Communicate in a manner that is scholarly, professional, respectful, and consistent with expectations for professional practice in education. Original work and critical thinking are required.
- **Media presentation:** 10–15 minutes presentation, with notes or a transcript to ensure accessibility to everyone. Upload the presentation using instructions given in the Using Kaltura [PDF].

Scoring note: You will need to show excellence in the production of your video for a distinguished score on the quality of your presentation.

Course Resources
Kaltura Basics Tutorial | Transcript
Using Kaltura [PDF]
Disability Services
Required Hardware

1. Headset with microphone
2. External or built-in microphone
3. Broadband Internet connection

u03d1 - Examining Two Test Types and Their Applications at Two Test Levels

Test types clearly define the objective of the test at a certain test level. In this discussion, you will examine the different test types and how they are applied at various test levels.

- Describe the objective of any two test types.
- Explain how these test objectives are applied to at least two test levels.

Response Guidelines

Write at least two response posts. In your posts, you might make a comparison, add information, ask questions, or respond in any way that will help your fellow learners.

Course Resources

Undergraduate Discussion Participation Scoring Guide

Unit 4 >> Application and Code Security

Introduction

In this unit you will examine the following:

- The various rules of the Java programming language for application and code security.
- How the rules are applied in order to test and ensure the security of the Java code.

For the assignment in this unit, you will analyze some of these rules and provide examples of compliant and non-compliant Java code.

Learning Activities

u04s1 - Studies

Required Readings

Regarding application rules, code security rules, and standard rules for developing secure Java code:

- Oracle. (n.d.). Secure coding guidelines for Java SE. Retrieved from http://www.oracle.com/technetwork/java/seccodeguide-139067.html
- McGraw, G., & Felten, E. (1998). Twelve rules for developing more secure Java code: Writing security-conscious Java code can help you avoid security surprises. *JavaWorld.*
  - Although this article was written in 1998, its rules are still considered as the golden rules for secure Java code. These rules represent the best practices when it comes to writing secure Java code.

Required Research

For this unit's assignment, you will need to study the rules defined in the following resource and be prepared to apply four of these rules in your work:

- Carnegie Mellon University Software Engineering Institute. (n.d.). SEI CERT oracle coding standard for Java. Retrieved from https://www.securecoding.cert.org/confluence/display/java/SEI+CERT+Oracle+Coding+Standard+for+Java

u04s1 - Learning Components

- Classify Java application and code security rules.
- Explore the vulnerabilities and protections provided by the Java application and code security rules.
- Understand how the Java application and code security rules are applied.

u04a1 - Analyzing Application and Code Security Rules

Overview

Application and code security rules are coding rules designed to prevent the software security exploitations and vulnerabilities.

In this assignment, you will examine a number of application and code security rules that are maintained by the Software Engineering Institute (SEI). You will identify four such rules and analyze them in terms of the vulnerability against which they protect. You will also identify how they provide this protection. Provide code examples showing rule compliant Java code and rule non-compliant Java code.

Submit your completed assignment as an APA-formatted paper.

Preparation

To prepare for this assignment, review the studies in this unit.

Directions

1. Identify four application and code security rules from the SEI Cert Oracle Coding Standard for Java (for example, name or classification of the rule.) For a distinguished score on this criterion, frame them clearly using your own words.
2. Analyze the four application and code security rules according to the criteria below. For a distinguished score on these criteria, your explanation should be well organized and well documented.
    - Explain the code vulnerabilities against which each of four application and code security rules will protect.
    - Explain how each of four application and code security rules protects against vulnerabilities.
3. Explain how four application and code security rules are applied to Java code (with compliant and non-compliant code examples.) For a distinguished score on this criterion, your explanation should be well organized and well documented.

Submission Requirements

Submit the completed assignment as an APA-formatted paper.

Course Resources

SEI CERT Oracle Coding Standard for Java

u04d1 - Providing Examples of Compliant and Non-Compliant Code for Secure Java Code Rules

Writing secure Java code has been a major concern of the Java community for Java's early beginners. In this discussion, you will explore some of the classic rules of writing secure Java code and how these rules are applied in code. Review the article by McGraw and Felten (1998) for a better understanding. Although this article was written in 1998, its rules are still considered as the golden rules for writing secure Java code. These rules represent the best practices when it comes to writing secure Java code.

- Briefly describe one of the twelve security rules mentioned in the article.
- Demonstrate how your chosen rule is applied for writing secure Java code by providing examples of both compliant and non-compliant Java code.

Response Guidelines

Write at least two response posts. In your posts, you might make a comparison, add information, ask questions, or respond in any way that will help your fellow learners.

Course Resources
Undergraduate Discussion Participation Scoring Guide
Twelve Rules for Developing More Secure Java Code: Writing Security-Conscious Java Code Can Help You Avoid Security Surprises

Unit 5 >> Static Testing Techniques

Introduction

In this unit you will examine the following:

- Methods for conducting static tests of software code.
- Code reviews.
- Tools for static code analysis.

For the assignment in this unit, you will statically analyze Java code using the NetBeans IDE Inspector tool.

Learning Activities

u05s1 - Studies

Required Readings

Regarding software static testing techniques including reviews and static analysis tools:

- Black, R., Veenendaal, E. V., & Graham, D. (2012). *Foundations of software testing ISTQB certification* (3rd ed.). Boston, MA: Cengage.
  - Chapter 3, "Static Techniques", pages 50–70 (Sections 1–3.)

Assignment Tools and Required Materials

- University of Maryland. (n.d.). FindBugs - Find bugs in Java programs. Retrieved from http://findbugs.sourceforge.net/
  - This is a FindBugs tool.

- NetBeans (n.d.). Java Hints. Retrieved from http://wiki.netbeans.org/Java_Hints
    - You will need this website for this unit's assignment.

- Unit 5 Assignment Submission Template [DOC].
- Unit 5 Assignment Library Zip File [ZIP].

Other Useful Resources

Regarding static code analysis in the NetBeans IDE Java Editor and how to use it:

- Lahoda, J., & Stashkova, A. (n.d.). Static code analysis in the NetBeans IDE Java Editor. Retrieved from https://netbeans.org/kb/docs/java/code-inspect.html
- NetBeansVideos. (2012). Static analysis of Java code in NetBeans IDE. Retrieved from https://www.youtube.com/watch?v=WZHaCjaBPG8
- NetBeansVideos. (2014). New in NetBeans IDE 8.0.1: FindBugs 3.0 for Java 8. Retrieved from https://www.youtube.com/watch?v=bcdvgkXZXHg&feature=youtu.be

u05s1 - Learning Components

- Build skills for conducting a code review.
- Classify the techniques of static testing.
- Use tools to perform a static analysis of Java code.

u05a1 - Analyzing Java Code Statically

Overview

Static testing techniques provide a powerful way to assure the quality of software without actually executing the software.

In this assignment, your will utilize the NetBeans IDE Java Editor Inspector tool to statically analyze the code of a given NetBeans Java project.

The requirements of this assignment are as follows. The given NetBeans Java project will contain two classes (Book.java and Library.java) whose code you will statically analyze using the Inspector tool.

You will use the following two configurations of the Inspector tool to run your static analysis:

- The FindBugs configuration.
- The NetBeans Java Hints configuration.

You will explain your execution of the static analysis using the Inspector tool and you will also evaluate the results of your inspection. Submit your completed assignment using the provided submission template for this unit.

Preparation

To prepare for this assignment:

1. Review the studies in this unit.
2. Download the Unit 5 Assignment Library Zip File linked in the Resources. This zip file contains a NetBeans project made of the two classes of Book.java and Library.java. The source code of the Books.java and the Library.java is the subject of your static analysis using the Inspector tool.
3. Download the Unit 5 Assignment Submission Template linked in the Resources. Write your responses and use it to submit your assignment.

Directions

To complete this assignment:

1. Unzip the Unit 5 Assignment Library Zip File in a directory of your choice and load it into your NetBeans IDE.
2. Inspect the Book.java and Library.java source code using the NetBeans Inspector FindBugs Integration configuration. You might need to install the FindBugs Integration plug-in first if it is not already installed on your NetBeans. Click Tools > Plugins > Available Plugins and search for FindBugs to find the plugin. Use the following parameters for your inspection
    - Scope: The Library project.
    - Configuration: FindBugs.
3. Document your work by taking a screenshot of the resulting Inspector window showing the results of the FindBugs inspection. For a distinguished score on this criterion, provide the screenshots together with a step-by-step description of the process.
4. Select three different FindBugs possible bugs and examine their meanings, their categories, and the possible remedial actions.
5. Inspect the Book.java and Library.java source code using the NetBeans Inspector Java Hints configuration. Use the following parameters for your inspection:
    - Scope: The Library project.
    - Configuration: NetBeans Java Hints.
6. Document your work by taking a screenshot of the resulting Inspector window showing the results of the NetBeans Java Hints inspection. For a distinguished score on this criterion, provide screenshots together with a step-by-step description of the process.
7. Select three different NetBeans Java Hints and examine their meanings, their categories, and the possible remedial actions.
8. Explain the approach that you took to complete this assignment and the major decisions you made.
9. Evaluate the results of your selected three FindBugs possible bugs (their meanings, their categories, and the possible remedial actions.)
10. Evaluate the results of your selected three NetBeans Java Hints (their meanings, their categories, and the possible remedial actions.)

Scoring note: Provide a well organized and well presented interpretation for a distinguished score on the criteria listed above. Screenshots may be needed in some cases.

Submission Requirements

- Prepare all the required screenshots.
- Prepare all of your explanations and evaluations.
- Complete all sections (taking screenshots, explanation of work, and evaluation of results) of the Unit 5 Submission Template linked in the Resources.
- Submit the completed submission template.

Course Resources
Unit 5 Assignment Library Zip File [ZIP]
FindBugs - Find Bugs in Java Programs
Java Hints
Unit 5 Assignment Submission Template [DOC]

u05d1 - Explaining the Purpose, Activities, Roles, and Responsibilities of Reviews

Reviews are powerful static test techniques to improve not only the quality of software but also the productivity of the development team. In this discussion, you will examine the various types of reviews and the purpose they serve.

- Describe the purpose of any two types of reviews.
- Explain the activities, roles, and responsivities of your two chosen types of reviews.

Response Guidelines

Write at least two response posts. In your posts, you might make a comparison, add information, ask questions, or respond in any way that will help your fellow learners.

Course Resources
Undergraduate Discussion Participation Scoring Guide

Introduction

In this unit you will need to do:

- Examine the various techniques of dynamic testing of software.
- Learn the objectives of these various techniques and when each should be used.
- Focus on the experience-based testing technique in this unit.

There is no assignment for Unit 6.

Learning Activities

u06s1 - Studies

Required Readings

Regarding the various techniques of dynamic testing:

- Black, R., Veenendaal, E. V., & Graham, D. (2012). *Foundations of software testing ISTQB certification* (3rd ed.). Boston, MA: Cengage.
  - Chapter 4, "Test Design Techniques," pages 71–81 (Sections 1 and 2.)

Regarding experience-based testing techniques:

- Black, R., Veenendaal, E. V., & Graham, D. (2012). *Foundations of software testing ISTQB certification* (3rd ed.). Boston, MA: Cengage.
  - Chapter 4, "Test Design Techniques," pages 106–120 (Sections 5 and 6.)

u06s1 - Learning Components

- Classify the software dynamic test techniques.
- Explain the purpose and objectives of each technique.

u06d1 - Classifying Dynamic Test Techniques and the Defects They Reveal

There are three main types of dynamic testing techniques. These are specification-based (black-box), structure-based (white-box), and dynamic (experience-based) testing techniques.

In this discussion, you will classify these different types of testing techniques and explore the different categories of defects they reveal.

- Identify one of the dynamic testing techniques and its test basis.
- Describe the test levels at which your dynamic testing technique is applied.
- Classify the types of defects that your dynamic testing technique reveals.

Response Guidelines

Write at least two response posts. In your posts, you might make a comparison, add information, ask questions, or respond in any way that will help your fellow learners.

Course Resources

Undergraduate Discussion Participation Scoring Guide

Unit 7 >> White-Box Testing

Introduction

In this unit you will need to:

- Continue to explore the various techniques of software dynamic testing.
- Concentrate on the design and development of test cases using the white-box testing technique.
- Learn the various methods to design test cases using this technique.

For the assignment in this unit, you will design component level test cases using white-box techniques.

Learning Activities

u07s1 - Studies

Required Readings

Regarding the white-box testing techniques that you will apply in this unit's assignment:

- Black, R., Veenendaal, E. V., & Graham, D. (2012). *Foundations of software testing ISTQB certification* (3rd ed.). Boston, MA: Cengage.
    - Chapter 4, "Test Design Techniques," pages 99–106 (Section 4.)

Regarding the practical applications of white-box test design techniques:

- Black, R. (2007). *Pragmatic software testing: Becoming an effective and efficient test professional.* New York, NY: Wiley.
    - Chapter 21, "Control-Flow Testing," pages 281–286.
    - Chapter 22, "Control-Flow Testing Exercise," pages 287–289.

u07s1 - Learning Components

- Build skills to design test cases using the structure-based technique.
- Classify the structure-based techniques for measuring coverage.
- Learn how to determine software coverage in testing.

u07a1 - Designing Component Level Test Cases Using White-Box Techniques

Overview

White-box testing techniques serve the two following purposes:

- They provide a measure for the coverage of the source code.
- They guide you in the design of test cases.

In this assignment, you will apply these white-box testing techniques to design test cases at the component level (method of a class) of a given NetBeans Java project.

The requirements of this assignment are as follows. The given NetBeans Java project contains two classes (Book.java and Library.java). The Library.java class contains a method (describeBooksBy()) which is the method for which you will design white-box test cases. You will design your test cases for the describeBooksBy() of the Library.java class for these coverages:

- Statement coverage.
- Decision (if, loop, and case) coverage, if applicable.

Each test case should clearly document the following:

- The setup environment for the test case, it's pre- and post-conditions.
- The selected input to the test case.
- An expected output of the test case.

You will explain your approach to the test case design and you will also evaluate your own design. Submit your completed assignment using the provided submission template for this unit.

Preparation

To prepare for this assignment:

1. Review the studies in this unit.
2. Download the Unit 7 Assignment Library Zip File linked in the Resources. The zip file contains a NetBeans project with the two classes of Book.java and Library.java. You will design white-box test cases for the describeBooksBy() method of the Library.java class.
3. Download the Unit 7 Assignment Submission Template linked in the Resources. Write your responses and use it to submit your assignment.

Directions

To complete this assignment:

1. Unzip the Unit 7 Assignment Library Zip File in a directory of your choice and load it into your NetBeans IDE.
2. Examine the source code of both Book.java and Library.java
3. Design white-box test cases for the describeBooksBy() method of the Library.java class to meet the requirements stated in the Unit 7 Assignment Submission Template linked in the Resources.
4. Explain the approach you took to design your software test cases and the major decisions you made. Self-reflect on the learning experience and lessons learned for a distinguished score on this criterion.
5. Evaluate your test cases design in terms of the following:
    - Coverage.
    - Environment setup and pre- or post-conditions.
    - Selected input.
    - Expected output.

Scoring note: Provide well organized and well presented content for a distinguished score on the criteria above. You are encouraged to reflect on what you have learned for at least one instance.

Submission Requirements

- Prepare all test cases design.
- Prepare all your explanations and evaluations.
- Complete all sections (test case design, explanation of work, and evaluation of test cases design) of the Unit 7 Assignment Submission Template linked in the Resources.
- Submit the completed submission template.

Course Resources

Unit 7 Assignment Library Zip File [ZIP]

Unit 7 Assignment Submission Template [DOC]

u07d1 - Classifying Structure-Based Techniques to Measure Coverage and Design Tests

There are a number of different structure-based techniques to measure coverage in software code and to design white-box test cases for it.

In this discussion, you will classify these different structure-based techniques of measuring coverage and explore how they are used to design white-box tests.

- Identify one of the structure-based techniques for measuring coverage in software code.
- Explain how code coverage is calculated in your chosen technique.
- Describe how white-box test cases can be designed from the coverage measurement of your chosen technique.

Response Guidelines

Write at least two response posts. In your posts, you might make a comparison, add information, ask questions, or respond in any way that will help your fellow learners.

Course Resources

Undergraduate Discussion Participation Scoring Guide

Unit 8 >> Black-Box Testing

Introduction

In this unit you will:

- Continue to study the various techniques of dynamic testing.
- Learn the black-box or the specification-based testing techniques.
- Examine the various methods of conducting software tests using the black-box technique.
- Learn how tools can be used to automate black-box testing.

For the assignment in this unit, you will dynamically black-box test a component using NetBeans IDE JUnit.

Learning Activities

u08s1 - Studies

Required Readings

Regarding black-box testing techniques:

- Black, R., Veenendaal, E. V., & Graham, D. (2012). *Foundations of software testing ISTQB certification* (3rd ed.). Boston, MA: Cengage.
    - Chapter 4, "Test Design Techniques," pages 81–98 (Section 3.)

Regarding JUnit testing in NetBeans and how it is used to develop and execute black-box testing:

- NetBeans. (n.d.). Writing JUnit tests in NetBeans IDE. Retrieved from https://netbeans.org/kb/docs/java/junit-intro.html
- Retterer, D. (2016). JUnit with NetBeans introduction. Retrieved from https://www.youtube.com/watch?v=4CehzWIpV48&feature=youtu.be
- Hayden, J. (n.d.). A guide to robust unit and integration tests with JUnit. Retrieved from https://www.toptal.com/java/getting-started-with-junit
- JUnit.org. (n.d.). JUnit. Retrieved from http://junit.org/junit4/

u08s1 - Learning Components

- Classify the different methods of specification-based testing.
- Explore the process of specification-based testing.
- Practice using tools to automate the testing of specification-based test cases.

u08a1 - Dynamically Black-Box Testing a Component Using NetBeans JUnit

Overview

Black-box testing of software is an input and output testing technique. It views the executing software as a black-box with only inputs and outputs (without any knowledge of what is inside the box.)

In this assignment, you will use the Netbeans JUnit framework to dynamically black-box test a component (method of a class) of a given NetBeans Java project.

The requirements of this assignment are as follows:

- The given NetBeans Java project contains two classes (Book.java and Library.java).
- The Library.java class contains a method (describeBooksBy()). It is the method that you will black-box test.

You will use the NetBeans JUnit framework to develop four black-box test methods for the describeBooksBy() method of the Library.java class. You will also execute these test methods and report on the results.

You will explain your approach to the development and execution of the test methods and you will also evaluate your own black-box testing. Submit your completed assignment using the provided submission template for this unit.

Preparation

To prepare for this assignment:

1. Review the studies in this unit.
2. Download the Unit 8 Assignment Library Zip File linked in the Resources. The zip file contains a NetBeans project with the two classes of Book.java and Library.java. You will develop the JUnit test methods for the describeBooksBy() method of the Library.java class.
3. Download the Unit 8 Assignment Submission Template linked in the Resources. You will use it to submit your response to this assignment.

Directions

To complete this assignment:

1. Unzip the Unit 8 Assignment Library Zip File in a directory of your choice and load it into your NetBeans IDE.
2. Examine the source code of both Book.java and Library.java
3. Develop four black-box test methods for the describeBooksBy() method of the Library.java class using JUnit.
4. Execute your developed test methods and revise if necessary.
5. Report on the results of your test executions.
6. Document your work by taking screenshots of your test execution results.
7. Explain the approach that you took to develop and execute your software test methods and the major decisions you made. Self-reflect on the learning experience and lessons learned for a distinguished score.
8. Evaluate the results of your black-box test (quantity of test methods, quality of test methods, number of test methods that passed, and the number of test methods that failed.)
9. Employ the vocabulary of software testing, security, and quality assurance accurately in your writing.

Scoring note: Provide well organized and well presented content for a distinguished score on the criteria above. You are encouraged to reflect on what you have learned for at least one instance.

Submission Requirements

- Prepare a Zip file of your NetBeans project that includes your test methods. Prepare text copies of your four test methods.
- Prepare all black-box test reports. Take all the required screenshots.
- Prepare all of your explanations and evaluations.
- Complete all sections (copies of test methods, NetBeans project file, report, screenshots, explanation of work, evaluation of results) of the Unit 8 Assignment Submission Template linked in the Resources.
- Submit the completed submission template.

Course Resources

Unit 8 Assignment Library Zip File [ZIP]

Unit 8 Assignment Submission Template [DOC]

u08d1 - Classifying Specification-Based Techniques

There is a number of different specification-based techniques to test executing software without the knowledge of its inner structure. In this discussion, you will classify these different specification-based techniques and examine how they are used to design black-box tests.

- Identify one of the specification-based techniques.
- Explain how black-box test cases can be designed using your chosen technique.

Response Guidelines

Write at least two response posts. In your posts, you might make a comparison, add information, ask questions, or respond in any way that will help your fellow learners.

Course Resources
Undergraduate Discussion Participation Scoring Guide

Unit 9 >> Test Planning and Estimation

Introduction

In this unit you will examine the following:

- Various managerial and administrative aspects of testing.
- The structure and organization of the test team and role played by its members.
- Test plans, test strategies, and test schedules.

There is no assignment for Unit 9.

Learning Activities

u09s1 - Studies

Required Readings

Regarding managing the testing process:

- Black, R., Veenendaal, E. V., & Graham, D. (2012). *Foundations of software testing ISTQB certification* (3rd ed.). Boston, MA: Cengage.
    - Chapter 5, "Test Management," pages 121–136 (Sections 1 and 2.)

u09d1 - Examining Two Types of Test Strategies

Test strategies and approaches are powerful factors for the success of the test effort and for the accuracy of test plan and estimation. In this discussion, you will examine the different types of test strategies that are in common use and explain when each is most suitable for a testing project.

- Identify two types of test strategies that are in common use.
- Explain when each of your chosen test strategies is most suitable for a testing project.

Response Guidelines

Write at least two response posts. In your posts, you might make a comparison, add information, ask questions, or respond in any way that will help your fellow learners.

Course Resources

Undergraduate Discussion Participation Scoring Guide

u09d1 - Learning Components

- Classify the different test approaches or strategies and when each should be taken.
- Study the purpose of a test plan.

Unit 10 >> Tools for Software Testing

Introduction

In this unit you will examine the following:

- The many different tools that are used throughout the test process.
- The purpose of these tools and how they are used in testing.
- The risks associated with using these tools.

For the assignment in this unit, you will write an APA-formatted paper about four of these tools to analyze their usage, benefits, and risks.

Learning Activities

u10s1 - Studies

Required Readings

Regarding the various tools that can be used during the testing process:

- Black, R., Veenendaal, E. V., & Graham, D. (2012). *Foundations of software testing ISTQB certification* (3rd ed.). Boston, MA: Cengage.
    - Chapter 6, "Tool Support for Testing," pages 160–186 (Sections 1–3.)

u10s1 - Learning Components

- Understand the purpose of software testing tools and the support they provide to the testing process.
- Classify the tools that are used in testing.

u10a1 - Identifying, Classifying, and Analyzing Four Testing Tools

Overview

You have been using the NetBeans built-in testing tools for your testing assignments in this course. But there is a number of other testing tools that bring significant productivity to the testing process and at the same time are risky to use.

In this assignment, you will perform the following:

- Identify four other testing tools (apart from the NetBeans built-in testing tool.)
- Classify them according to their roles in the testing process.
- Analyze their usages, benefits, and risks.

Submit your completed assignment as an APA formatted written paper.

Preparation

To prepare for this assignment, review the studies in this unit.

Directions

To complete this assignment:

1. Search the Internet and other sources for available testing tools.
2. Select four of these tools.
3. Briefly identify the four testing tools (for example, maker of the tool, open source, and commercial tool.)
4. Classify the four testing tools according to the testing activities they support (for example, a testing tool for static testing as well as a testing tool for test execution and logging.)
5. Analyze the four testing tools:
    - Explain how the tools support the testing process according to their classification.
    - Explain the benefits of using the tools.
    - Explain the risks associated with using the tools.
6. Employ the vocabulary of software testing, security, and quality assurance accurately in your writing.

Scoring note: Provide well organized and well presented content for a distinguished score on the criteria above. Use the course vocabulary consistently and accurately.

Submission Requirements

Submit the completed assignment as an APA formatted written paper.

u10d1 - Defining Test Tools and Their Purpose With Examples

In testing, there are always opportunities to utilize different test tools throughout the test process. In this discussion, you will define the term test tool and describe the purpose of these test tools with examples.

- Define what is meant by a test tool.
- Describe at least two objectives or purposes of using test tools.
- Support your description with concrete examples.

Response Guidelines

Write at least two response posts. In your posts, you might make a comparison, add information, ask questions, or respond in any way that will help your fellow learners.

Course Resources