CS 410 Project One Guidelines and Rubric

Competencies

In this project, you will demonstrate your mastery of the following competencies:

- Utilize fundamental reverse engineering technologies and practices
- Apply the fundamentals of assembly language in reverse engineering tasks

Scenario

SNHU Investments is a large investment company that offers brokerage and retirement services. They have been in business since the early 1980s. Recently, they have been trying to move their legacy code into a cloud-based application. This resulted in the company looking to hire an additional software engineer. You applied for the position and were selected to participate in a proficiency test as part of the interview process. In the proficiency test, you must demonstrate your ability to reverse engineer code.

For the purposes of this proficiency test, you will be given an existing client management application that the company uses. You will convert the application into assembly code, explain the code, and convert it into C++ to create a working program.

Directions

For this project, you will take the existing binary file (O file) for the client management application, convert it to assembly code, convert the assembly code to C++ (CPP file) with an added output statement, and create a new binary file.

1. Convert the binary file (O file) into assembly code.
    - Decompile the binary file found in the Software Reverse Engineering Playground in the Project One Files folder in Codio.
2. Explain the functionality of the blocks of assembly code.
    - Use the Project One Proficiency Test Template, located in the Supporting Materials section, to complete this step.
3. Convert the assembly code into C++ code (CPP file).
    - Compile the C++ code in the Eclipse IDE.
4. Add an output statement to the beginning of the C++ code (CPP file).
    - For example, you could add a "Created by (name)" statement for when the program opens, or a statement of your choice.
5. Convert the CPP file to a new binary file (O file).

What to Submit

To complete this project, you must submit the following:

Project One Proficiency Test Template

This should be a Word Document (DOCX) with an explanation of all blocks of assembly code. Use the template linked in the Supporting Materials section.

C++ File (CPP file)

This file is needed to ensure that the code can be altered. It is not the executable file. This will include your added output statement.

Binary File (O file)

This file is needed to run the application showing your added output statement. This is considered the executable file.

Supporting Materials

The following resource(s) may help support your work on the project:

Project One Proficiency Test Template Word Document

Use this template to explain the functions of the assembly code blocks. This document includes tables with headings to include explanations for "main", Change Customer Choice, and additional functions.

Guide to Software Reverse Engineering

This guide provides information on the process of software reverse engineering, assembly language, binary, C++, and more.

Codio Guide

This guide provides information on how to use the Codio platform to complete aspects of this project.

Project One Rubric

Criteria	Exemplary	Proficient	Needs Improvement	Not Evident	Value
Binary to Assembly Conversion	N/A	Converts binary file into assembly code (100%)	Shows progress toward proficiency, but with errors or omissions; areas for improvement may include converting the binary file into assembly code (55%)	Does not attempt criterion (0%)	16.67
Assembly Code Functionality Explanation	Exceeds proficiency in an exceptionally clear, insightful, sophisticated, or creative manner (100%)	Explains the functionality of the blocks of assembly code with adequate detail and no inaccuracies (85%)	Shows progress toward proficiency, but with errors or omissions; areas for improvement may include explanations of the functionality of the blocks of assembly code with richer detail and no inaccuracies (55%)	Does not attempt criterion (0%)	23.33
Assembly Code Conversion to C++	N/A	Converts assembly code into C++ code with no inaccuracies or missing functionality (100%)	Shows progress toward proficiency, but with errors or omissions; areas for improvement may include converting assembly code into C++ code with few inaccuracies or missing functionality (55%)	Does not attempt criterion (0%)	26.67
Added Output Statement	Exceeds proficiency in an exceptionally clear, insightful, sophisticated, or creative manner (100%)	Adds an output statement to the beginning of C++ code (85%)	Shows progress toward proficiency, but with errors or omissions; areas for improvement may include adding an output statement to the beginning of the C++ code (55%)	Does not attempt criterion (0%)	13.33
C++ Conversion to Binary	N/A	Converts C++ code into binary code (100%)	Shows progress toward proficiency, but with errors or omissions; areas for improvement may include converting C++ code into binary code (55%)	Does not attempt criterion (0%)	16.67
Articulation of Response	Exceeds proficiency in an exceptionally clear, insightful, sophisticated, or creative manner (100%)	Clearly conveys meaning with correct grammar, sentence structure, and spelling, demonstrating an understanding of audience and purpose (85%)	Shows progress toward proficiency, but with errors in grammar, sentence structure, and spelling, negatively impacting readability (55%)	Submission has critical errors in grammar, sentence structure, and spelling, preventing understanding of ideas (0%)	3.33
				Total:	100%