Southern New Hampshire University

CS 260 Final Project Guidelines and Rubric

Overview

Data structures and algorithms are essential tools for anyone entering the field of computer science. These tools consist of specific patterns that, when applied correctly, solve problems. As a professional working in the field of computer science, you must be aware of the various data structures and algorithms that are available, and more importantly, when to apply them. Being familiar with different data structure and algorithm options allows you to solve problems efficiently, because you will not have to re-create a solution repeatedly.

In this project, you will select examples from your coursework to showcase the knowledge and skills you have developed in this course. You will compile this portfolio of work in a compressed ZIP file that will include the sources codes for each of your programs. You will complement your portfolio with a reflection paper that defends the work and articulates the lessons learned from applying algorithms and data structures to solve processing problems.

In this assignment, you will demonstrate your mastery of the following course outcomes:

- Apply appropriate data structures for effectively organizing data given the requirements and constraints of various problems
- Implement technically sound algorithms that accurately perform required functions
- Employ basic algorithms and common data structures in developing effective computer programs
- Assess foundational algorithms and data structures for their use in solving processing problems

Scenario

The source data is a cumulative list of the monthly sales of surplus property items on eBid Nashville beginning in January, 2014. The interactive data set is available online.

For your final project, you will assume the role of a programmer at SNHU Software. Managers of eBid Nashville are looking for a program that allows users to access information on the items sold through the eBid website. Throughout this course, you will contribute specific functional components of this larger piece of software. You will not be responsible for building the entire project; instead, you will submit individual artifacts that employ specific algorithms and data structures to meet the requirements of certain features of the program. The requirements you will be addressing are:

- Bid ID Search (Module Three, Five, and Six)
- Sorting by Bid Title (Module Four)

Southern New Hampshire University

Prompt

Your portfolio of work should be a compressed ZIP file that includes the final, polished versions of each of your programs' source codes. Any programs discussed in your reflection paper must be present in the portfolio.

Your reflection paper should be a polished document that addresses each of the following **critical elements**:

I. **Data Structures:** In the first section of your report, indicate which programs in your portfolio best exemplify your knowledge and skills related to data structures. For each example that you provide, defend your program in terms of how effectively the data is organized, using specific evidence from the problem. Specifically, you must discuss each of the following data structure types:
   A. **Vectors**
   B. **Hash tables**
   C. **Tree structures**

II. **Algorithms:** In the second section of your report, indicate which programs within your portfolio best exemplify your knowledge and skills related to algorithms. For each example that you provide, defend your program as one that is technically sound and can perform the required functions, using specific evidence. Specifically, you must discuss each of the following algorithm types:
   A. **Search**
   B. **Sort**
   C. **Hash/Chaining**

III. **Student's Choice**: Select your best or favorite program that uses both an algorithm and a data structure.
   A. Defend the **overall effectiveness** of this computer program. To what extent did you implement the algorithm(s) and data structure(s) to create a program that works? Support your claims with specific evidence from your work.
   B. Defend your **programming abilities** as they are illustrated in this program:
      1. To what extent is your code **modularly** composed? Support your claims with specific evidence from your work.
      2. To what extent is your code **reusable**? Support your claims with specific evidence from your work.
      3. To what extent are your **annotations** clear in explaining your choices and underlying reasoning? Specifically, how are your choices logically based on the needs of the case? Support your claims with specific evidence from your work.

IV. **Conclusions:** In the final section of your report, reflect on the broader principles and takeaways from this course.

A. First, assess the role and importance of **data structures** in developing computer programs. How does the type of data structure impact or constrain the development of an effective computer program? Support your claims using specific evidence from the course materials.

B. Second, assess the role and importance of **algorithms** in developing computer programs. How does the choice of algorithms impact or constrain the development of an effective and efficient computer program? Support your claims using specific evidence from the course materials.

C. Finally, articulate the **lessons learned** about solving processing problems. Specifically, how might you employ your knowledge of algorithms and data structures to solve a processing problem that is of interest to you in your personal or professional life? Illustrate your response with specific examples.

Final Project Rubric

Guidelines for Submission: Your reflection paper should be 3–4 pages, double spaced, with 12-pt. Times New Roman font and one-inch margins.

Critical Elements	Exemplary	Proficient	Needs Improvement	Not Evident	Value
Data Structures: Vectors		Identifies an appropriate program from the portfolio that uses vectors, and defends its effectiveness in organizing the data given the requirements and constraints of the problem, using specific evidence (100%)	Identifies a program from the portfolio that uses vectors, but does not fully or logically defend its effectiveness in organizing the data given the requirements and constraints of the problem using specific evidence (55%)	Does not identify a program from the portfolio that vectors (0%)	7.5
Data Structures: Hash Tables		Identifies an appropriate program from the portfolio that uses hash tables, and defends its effectiveness in organizing the data given the requirements and constraints of the problem, using specific evidence (100%)	Identifies a program from the portfolio that uses hash tables, but does not fully or logically defend its effectiveness in organizing the data given the requirements and constraints of the problem using specific evidence (55%)	Does not identify a program from the portfolio that uses hash tables (0%)	7.5
Data Structures: Tree Structures		Identifies an appropriate program from the portfolio that uses tree structures, and defends its effectiveness in organizing the data given the requirements and constraints of the problem, using specific evidence (100%)	Identifies a program from the portfolio that uses tree structures, but does not fully or logically defend its effectiveness in organizing the data given the requirements and constraints of the problem using specific evidence (55%)	Does not identify a program from the portfolio that uses tree structures (0%)	7.5

Algorithms: Search		Identifies an appropriate program from the portfolio that uses a search algorithm, and defends its technical soundness and its ability to perform the required functions, using specific evidence (100%)	Identifies an appropriate program from the portfolio that uses a search algorithm, but does not fully or logically defend its technical soundness or its ability to perform the required functions using specific evidence (55%)	Does not identify a program from the portfolio that uses a search algorithm (0%)	7.5
Algorithms: Sort		Identifies an appropriate program from the portfolio that uses a sort algorithm, and defends its technical soundness and ability to perform the required functions, using specific evidence (100%)	Identifies an appropriate program from the portfolio that uses a sort algorithm, but does not fully or logically defend its technical soundness or its ability to perform the required functions using specific evidence (55%)	Does not identify a program from the portfolio that uses a sort algorithm (0%)	7.5
Algorithms: Hash/Chaining		Identifies an appropriate program from the portfolio that uses a hash algorithm, and defends its technical soundness and ability to perform the required functions, using specific evidence (100%)	Identifies an appropriate program from the portfolio that uses a hash algorithm, but does not fully or logically defend its technical soundness or its ability to perform the required functions using specific evidence (55%)	Does not identify a program from the portfolio that uses a hash algorithm (0%)	7.5
Student's Choice: Overall Effectiveness		Defends the overall effectiveness of the program in terms of how the algorithm(s) and data structure(s) were properly implemented, using specific evidence (100%)	Defends the overall effectiveness of the program, but fails to fully or logically demonstrate how the algorithm(s) and data structure(s) were properly implemented using specific evidence (55%)	Does not defend the overall effectiveness of the program (0%)	7
Student's Choice: Programming Abilities: Modularly	Meets "Proficient" criteria and demonstrates sophisticated programming abilities or develops particularly elegant code (100%)	Defends the quality of the code in terms of its modularity using specific supporting evidence (85%)	Defends the quality of the code but fails to fully or accurately demonstrate its modularity using specific evidence (55%)	Does not defend the quality of the code, or code is not modular (0%)	7

Student's Choice: Programming Abilities: Reusable	Meets "Proficient" criteria and demonstrates sophisticated programming abilities or develops particularly elegant code (100%)	Defends the quality of the code in terms of its reusability using specific supporting evidence (85%)	Defends the quality of the code but fails to fully or accurately demonstrate its reusability using specific evidence (55%)	Does not defend the quality of the code, or code is not reusable (0%)	7
Student's Choice: Programming Abilities: Annotations	Meets "Proficient" criteria and demonstrates sophisticated programming abilities or develops particularly elegant code (100%)	Defends the quality of the annotations in terms of their clarity and logical foundation using specific supporting evidence (85%)	Defends the quality of the annotations but fails to fully or logically demonstrate their clarity or foundation using specific supporting evidence (55%)	Does not defend the quality of the annotations, or annotations are not present (0%)	7
Conclusions: Data Structures	Meets "Proficient" criteria and demonstrates nuanced understanding of the potential for foundational algorithms and data structures to be useful in solving processing problems (100%)	Assesses the role and importance of data structures, including explaining how data structures can impact or constrain the development of effective computer programs, using specific supporting evidence from the course materials (85%)	Assesses the role and importance of data structures but fails to fully or logically explain how data structures can impact or constrain the development of effective computer programs using specific supporting evidence from the course materials (55%)	Does not assess the role and importance of data structures in developing effective computer programs (0%)	7.5
Conclusions: Algorithms	Meets "Proficient" criteria and demonstrates nuanced understanding of the potential for foundational algorithms and data structures to be useful in solving processing problems (100%)	Assesses the role and importance of algorithms, including explaining how the choice of algorithms can impact or constrain the development of effective computer programs, using Big O notation to support (85%)	Assesses the role and importance of algorithms but fails to fully or logically explain how the choice of algorithms can impact or constrain the development of effective computer programs using Big O notation to support (55%)	Does not assess the role and importance of algorithms in developing effective computer programs (0%)	7.5
Conclusions: Lessons Learned	Meets "Proficient" criteria and demonstrates deep appreciation for the potential of foundational algorithms and data structures to be useful in solving processing problems (100%)	Articulates lessons learned by identifying algorithms and data structures that can be applied to a processing problem of interest, using specific illustrating examples (85%)	Articulates lessons learned but fails to fully or logically identify algorithms and data structures that can be applied to a processing problem of interest using specific illustrating examples (55%)	Does not articulate lessons learned about using algorithms and data structures to solve processing problems (0%)	7.5

Articulation of Response	Submission is free of errors related to grammar, spelling, syntax, and organization and is presented in a professional and easy-to-read format (100%)	Submission has no major errors related to grammar, spelling, syntax, or organization (85%)	Submission has major errors related to grammar, spelling, syntax, or organization that negatively impact readability and articulation of main ideas (55%)	Submission has critical errors related to grammar, spelling, syntax, or organization that prevent understanding of ideas (0%)	4.5
				Total	**100%**