COMP 101 Study Guide for FUPE

Format
The exam consists of 10 pages of short answer, essay, problems, and programming for 150 points. Passing is 80%. It is two hours in length and is closed book, and notes. For those students where English is a second language, a translation dictionary may be used. Otherwise, no external materials can be used.

Notes
This is a difficult exam. You will be expected to do problems related to both reading and writing code. Scoping rules, arrays, variables, control structures, and general computer concepts such as bits and bytes, number systems, etc. are all fair game. The outcomes below contain more detail about the exam contents.

Languages
The exam currently uses pseudo-code, flow-charts, and JavaScript as its programming languages. A review of these languages would be appropriate.

Recommended textbook

Course Description
Many organizations today utilize computers and information systems to store, organize, analyze, and summarize data to solve problems. As a result, computing is a tool that can benefit students in many different fields. At the heart of solving problems with computers is the study of structured thinking using algorithms. This course is designed for students with no prior programming experience and teaches the building blocks of algorithms, including variables, expressions, selection and repetition structures, functions and parameters, and array processing

Course Outcomes
1. Solve basic problems using programming.
2. Discuss the ethical, legal, and social issues related to the use of computers in society.

Module 1 outcomes
1. Describe the relationship between hardware and software.
2. Describe how computers store data.
3. Describe how a program works.
4. Summarize the various features of high-level programming languages.
5. Produce a simple flowchart (outputs only) using RAPTOR.

Module 2 outcomes
1. Explain the relationships among the program development cycle, top-down design, and testing.
2. Process the steps required in designing a program which include algorithms, pseudocode, flowcharts, and code.
3. Write a short program with user input, processing, and formatted output.
4. Assign variables with names and values and perform calculations on those variables.
5. Develop test cases to check program code.
6. Document a program.

Module 3 outcomes
1. Define and call a module or function in a program.
2. Reuse code through modules.
3. Determine the value of local and global variables in a program.
4. Pass arguments to modules.

Module 4 outcomes
1. Use Boolean logic and relational operations and logic operators in pseudocode, flowcharts, and JavaScript programs.
2. Use appropriate decision structures for pseudocode, flowcharts and JavaScript programs (e.g., if, if-then, and if-else statements).
3. Compare strings in a program.
4. Convert a program to use functions and parameters.
5. Write code using if and if-else in sample JavaScript programs.

Module 5 outcomes
1. Use repetition structures in pseudocode, flowcharts, and JavaScript programs.
2. Use appropriate condition-controlled loops for pseudocode, flowcharts, and JavaScript programs (e.g., while, do-while, and do-until).
3. Use Count-Controlled Loops and the For statement.
4. Properly calculate a running total.
5. Properly use sentinels.
6. Use nested loops properly.

Module 6 outcomes
1. Write a set of simple functions that return values.
2. Write a program to read and display data.
3. Write a program to save data.
4. Generate random numbers.
5. Write your own functions in JavaScript code.
6. Use library functions using JavaScript code.

**Module 7 outcomes**
1. Examine program code and resolve errors.
2. Write a program to add simple exception handling.
3. Become a defensive programmer.

**Module 8 outcomes**
1. Understand array basics including elements and subscripts.
2. Learn how to find high and low values of an array.
3. Learn how to step through and process the elements of an array.
4. Learn how to pass arrays as arguments to a module or a function.
5. Understand two-dimensional arrays.
6. Learn how to write arrays in pseudocode, flowcharts, and JavaScript code.