Computer Science II
Course Overview
[136-140 hours total]

1 Introduction to Programs and the Software Development Life Cycle

4 Hours

OBJECTIVES (concepts) – The student will be able to:

1.1 Define the following concepts:

- Algorithm
- Program
- Programming language
- Statement
- Input
- Output
- IDE

1.2 Identify features of the Java and Visual Basic programming languages.
1.3 Start an IDE.
1.4 Create a simple computer program in an IDE.
1.5 Run a computer program in an IDE.
1.6 Modify a program in an IDE.
1.7 Save a program in an IDE.
1.8 Identify vendors and websites that sell IDEs or offer them for free.
1.9 Identify a statement within a program.
1.10 Explain how whitespace such as embedded spaces and continuation across multiple lines affects statements.

OBJECTIVES (skills) – The student will be able to:

1.11 Describe the following steps of the Software Development Life Cycle and explain their purpose

- Specification
- Design
- Coding
- Debugging
- Testing
- Delivery

1.12 Use a standard process for writing a program:

- Identify the expected result of the problem
- Determine whether enough information has been given to solve the problem
- List detailed steps required to complete the tasks or solve a problems (algorithm)
- Determine the input and output required to satisfy program specifications.
- Delivery

1.13 Create a program with useful comments

2 Input/Output and Testing Programs

4 Hours

OBJECTIVES (concepts) – The student will be able to:

2.1 Create and run a program that outputs a string to the user.
2.2 Create and run a program that inputs a string from the keyboard.
2.3 Create and run a program that outputs numeric data to the user.
2.4 Create and run a program that inputs numeric data from the user.
2.5 Explain the Unicode and ASCII sets and use them in a program.

OBJECTIVES (skills) – The student will be able to:

2.6 Explain the importance of test data
2.7 Compose a test plan
2.8 Create test data and use it to test a program

3 Data, Variables, and Tracing Programs

8 Hours

OBJECTIVES (concepts) – The student will be able to:

3.1 Explain the role of a variable in a program.
3.2 Assign a value to a variable.
3.3 Compare and contrast the purpose of integers, floating point numbers, and strings.
3.4 Explain the difference between a variable declaration and a statement.
3.5 Initialize a variable.
3.6 Declare a named constant.
3.7 Identify and define the scope of a variable.
3.8 Identify the operators for addition, subtraction, multiplication, division, and modulo.
3.9 Create and run a program using the standard arithmetic operations.
3.10 Create and run a program using string concatenation.
3.11 Create and run programs that processes strings in each of the following ways:

- Determine whether two strings are equal
- Find the length of a string
- Find a substring within a string
- Find whether a string contains a given substring
- Find whether a string is before another in alphabetical order

3.12 Convert one data type to another.
3.13 Explain round-off error.
3.14 Use functions to compute the following:

- Random numbers
- Square roots
- Exponents
- Rounding numbers

3.15 Evaluate arithmetic expressions with multiple operators using knowledge of the order in which operations are evaluated.
3.16 Compare and contrast runtime errors with compile-time errors and be able to distinguish between them.

OBJECTIVES (skills) – The student will be able to:

3.17 Give meaningful names to variables
3.18 Trace by hand the value of variables through multiple assignment statements and operations.

3.19 Use debugging and tracing facilities provided by an IDE

4 Conditionals and Pseudo-Code

12 Hours

OBJECTIVES (concepts) – The student will be able to:

4.1 Identify the following relational operators:

- Equals
- Not equals
- Greater than
- Less than
- Less than or equal to
- Greater than or equal to

4.2 Create and run a program using an if statement.

4.3 Create and run a program using an if-else statement.

4.4 Create and run a program using a conditional containing a block of statements.

4.5 Create and run a program using a nested conditional.

4.6 Create and run a program using a conditional with more than two cases, such as an else-if or case.

4.7 Declare and use Boolean variables.

4.8 Evaluate expressions using logical operators (AND, OR, and NOT).

4.9 Create and run a program using logical operators.

OBJECTIVES (skills) – The student will be able to:

4.10 Explain the importance of planning the program during the design step.

4.11 Use pseudo-code to design a program using conditionals.

4.12 Properly indent a program using conditionals.

4.13 Trace the execution of a program that contains conditionals.

4.14 Define the term Code Coverage.

4.15 Generate test data that tests all statements in a program.

5 Loops and Debugging

24 Hours

OBJECTIVES (concepts) – The student will be able to:

5.1 Identify the purpose of looping.

5.2 Create and run a program using a while loop where the condition is evaluated at the top of the loop.

5.3 Create and run a program using a while loop where the condition is evaluated at the bottom of the loop.

5.4 Create and run a program using a for loop.

5.5 Create and run a program which increments and decrements variables.

5.6 Create and run a program using nested loops.

5.7 Select an appropriate looping structure for a given task.

5.8 Explain the concept of an infinite loop and how to avoid one.

5.9 Define and use flags/sentinels in a program.

OBJECTIVES (skills) – The student will be able to:

5.10 Debug a program by inserting debugging statements.

5.11 Generate test data that tests boundary conditions for a program.

5.12 Use stepwise refinement to design a program.

6 Methods, Parameters and Top-Down Design

12 Hours

OBJECTIVES (concepts) – The student will be able to:

6.1 Define the following terms and explain the similarities and differences between them:

- Module
- Procedure
- Subroutine
- Method
- Function
- Parameter
- Argument
- Return value
- Local variable
- Global variable

6.2 Create and run a program using a module that does not return a value.

6.3 Create and run a program using a module that returns a value.

6.4 Explain what it means to pass parameters to a module.

6.5 Create and run a program that uses a module with several parameters.

OBJECTIVES (skills) – The student will be able to:

6.6 Explain the importance of modularity

6.7 Use top-down decomposition to design a program

6.8 Document a module by specifying pre- and post-conditions

6.9 Choose appropriately which data should be passed through parameters and which should be stored using local variables

6.10 Test modules by creating stubs and drivers

7 Arrays

24 Hours

OBJECTIVES (concepts) – The student will be able to:

7.1 Define the terms composite data type, array, element, subscript, and index.

7.2 Describe how array elements are numbered in Java and Visual Basic.

7.3 Create and run a program that declares and uses an array.

7.4 Create and run a program that uses an array as a series of counters.

7.5 Create and run a program that initializes an array.

7.6 Distinguish between pass by reference and pass by value.

7.7 Create and run a program where an array is passed as a parameter into a module.

7.8 Create and run a program that uses a multidimensional array.

7.9 Define sorting.

7.10 Create and run a module that sorts the elements of an array using a selection sort strategy.

7.11 Create and run a module that sorts the elements of an array using a bubble sort strategy.

7.12 Compare and contrast selection and bubble sort.

7.13 Create and run a program that uses linear search to search for an item in an array.

7.14 Create and run a program that uses binary search to search for an item in an array.

7.15 Explain what is meant by the efficiency of an algorithm.

OBJECTIVES (skills) – The student will be able to:

7.16 Analyze the efficiency of a simple algorithm (optional)

8 Classes and Instances

12 Hours

OBJECTIVES (concepts) – The student will be able to:

8.1 Define the terms encapsulation, class, method, instance, and constructor.

8.2 Create a program that makes and uses an instance of a class.

8.3 Create a class that has one or more instance variables.

8.4 Create and run a program that calls a method on an instance of a class.

8.5 Create and run a program that uses a constructor to initialize an instance of a class.

8.6 Create and use a class with multiple constructors.

8.7 Explain the difference between public and private variables and methods.

8.8 Explain the difference between declaring a variable and setting that variable to an instance of a class.

8.9 Explain the meaning of a null instance

8.10 Explain how runtime errors result from a null instance, and how to avoid them

OBJECTIVES (skills) – The student will be able to:

8.11 Use encapsulation to determine which methods and variables should be private and which should be public

9 Polymorphism and Inheritance (Optional)

4 Hours

OBJECTIVES (concepts) – The student will be able to:

9.1 Explain the concept of overloaded methods

9.2 Create and use a class with overloaded methods

9.3 Define the terms superclass, subclass, and inheritance

9.4 Create and run a program that uses a subclass

OBJECTIVES (skills) – The student will be able to:

9.5 Read and create a UML class hierarchy diagram

9.6 Design a class hierarchy

10 Graphical User Interfaces

16 Hours

OBJECTIVES (concepts) – The student will be able to:

10.1 Create and run a program using each of the following components:

- Windows
- Labels
- Buttons
- Input boxes
- List boxes

10.2 Explain event handlers

10.3 Create and run a program using event handlers
10.4 Create and run a program that groups GUI components into a larger container
10.5 Create and run a program that uses color graphics
10.6 Create and run a program that draws lines and geometric shapes
10.7 Create and run a program that renders an image on the screen
10.8 Create and run a program using a timer event
10.9 Create and run a program using animation (optional)

OBJECTIVES (skills) – The student will be able to:

10.10 Design and implement a graphical user interface

11 Computer Hardware and Software

8 Hours

OBJECTIVES (skills) – The student will be able to:

11.1 Define the following terms relating to computer hardware:

- RAM
- ROM
- CPU
- ALU
- Storage
- Bits
- Bytes
- Peripherals
- Logic gates
- AND gates
- OR gates
- NOT gates

11.2 Explain the VonNeumann data path

11.3 Define each of the following terms and explain their roll in the function of computer programs:

- Machine language
- Machine instruction
- Assembler
- Compiler
- Linker
- Interpreter
- Bytecode
- Virtual Machine

11.4 Distinguish between operations performed by the hardware and operations performed by the operating system

11.5 Perform the following operations on binary numbers

- Convert a binary number to a decimal number
- Add two binary numbers

11.6 Identify the roll of the following in the evolution of programming languages:

- High-level languages

- FORTRAN
- COBOL
- Basic
- Lisp

- Block Structured Languages
  - ALGOL
  - PASCAL
  - C

- Object-Oriented Languages
  - SmallTalk
  - C++

- Scripting Languages
  - PERL
  - JavaScript
  - Python

- Markup Languages
  - HTML
  - XML

12 Written Communication and Information Technology Issues

12 Hours

OBJECTIVES (skills) – The student will be able to:

12.1 Identify and describe current issues relating to computer use.

12.2 Reach well-reasoned conclusions about current events and defend those conclusions in writing.

12.3 Demonstrate the ability to express written ideas clearly.

12.4 Read and interpret periodic literature and other public media relating to information technology.

12.5 Use search engines on the World Wide Web to research topics in information technology.