Arkansas CTE

Programming II: Grades 9, 10, 11, 12

Adopted 2003

Review Programming Techniques, Ethics, and Privacy

1.1 Discuss the ethical and privacy issues of programming

1. Identify ethical and privacy practices in computer programming 1.1.1

1.2 List the steps of the programming process

1. When given an example, be able to identify the correct step 1.2.1

Data Validation

2.1 Explain the importance of data validation

1. Give examples of good data validation rules for a variety of situations 2.1.1

2.2 Explain the logic of numeric range checks

1. Write programs that use range checks 2.2.1

2.3 Explain the logic of data validation to match a particular pattern

1. Write programs that require data to fit a specified pattern (i.e., Social Security Number 123-45-6789, phone number 123-456-7890, etc.) 2.3.1

String Manipulation

3.1 Explain the syntax and features of various commands dealing with ASCII or Unicode numbers and their corresponding characters

1. Write program lines to determine the ASCII number of a character 3.1.1
2. Write programs to use the ASCII number to print the corresponding character 3.1.2

3.2 Explain the syntax and purpose of commands that handle all or part of a string and that concatenate strings

1. Write programs to determine the number of characters in a string 3.2.1
2. Write programs to print a particular group of characters that are contained in a string 3.2.2
3. Write programs that concatenate multiple strings into one 3.2.3

3.3 Explain the reasons why breaking a string into its component parts is important

1. Write code that will take the first part of a string from a longer string (i.e., taking the area code from a telephone number) `3.3.1`
2. Write code that will take characters from the middle of a string (i.e., removing the middle name from the full name) `3.3.2`
3. Write code that will take characters from the right side of a string (i.e., ZIP code from the address) `3.3.3`

Procedures/Subprograms/Functions with Parameters

4.1 Explain the difference between argument and parameter

1. Give examples of arguments and parameters `4.1.1`

4.2 Explain the matching of arguments in the function call to the function parameters

1. Write programs that use arguments in function/procedure calls `4.2.1`

4.3 Explain when to use value parameters

1. Write functions/procedures with value parameters `4.3.1`

4.4 Explain when to use reference parameters

1. Write functions/procedures with reference parameters `4.4.1`

4.5 Explain when to use functions that return a value

1. Write functions that return values `4.5.1`

4.6 Explain why array parameters and other data structure parameters should be passed by reference

1. Write functions that have array or other data structure parameters `4.6.1`

4.7 Explain when to use a constant reference parameter (in languages where available)

1. Write functions that have constant reference parameters (in languages where available) `4.7.1`

Data Types - Boolean and Enumerated Types

5.1 Explain when and where to use Boolean expressions and variables

1. Write programs that use Boolean expressions and variables `5.1.1`

5.2 Discuss enumerated type (i.e., color, cards, etc.)

1. Write programs that declare and use enumerated types `5.2.1`

5.3 Explain the purpose of using enumerated types

1. Give examples of when enumerated types are needed `5.3.1`

One-dimensional Arrays or Vectors

6.1 Explain arrays and vectors

1. Give examples of when the use of arrays or vectors is appropriate `6.1.1`

6.2 Explain the use of dimensions and subscripts and the syntax of commands to use them

1. Write an appropriate program to dimension one-dimensional arrays or vectors `6.2.1`
2. Use subscript to access particular elements in a one-dimensional array or vector `6.2.2`

6.3 Explain the logical steps in initializing and loading a one-dimensional array

1. Write program lines that initialize a one-dimensional array or vector `6.3.1`
2. Write program lines that read data from a file into a one-dimensional array or vector `6.3.2`

6.4 Explain the logical steps in traversing a one-dimensional array to perform calculations and comparisons

1. Write loops that traverse a one-dimensional array or vector, performing calculations and comparisons `6.4.1`

6.5 Explain the logical steps in printing an entire array of data

1. Write loops that print the contents of a one-dimensional array or vector `6.5.1`

6.6 Explain the logical steps to insert a value in a one-dimensional array or vector

1. Write code that inserts values into an existing array or vector `6.6.1`

6.7 Explain the logical steps in deleting elements in a one-dimensional array or vector

1. Write code that deletes elements from an existing array or vector `6.7.1`

6.8 Explain the use of parallel one-dimensional arrays or vectors

1. Write programs that contain parallel one-dimensional arrays or vectors `6.8.1`

Structures

7.1 Discuss structure

1. Give an example of a useful structure `7.1.1`

7.2 Explain the advantages of structure in handling large amounts of related data

1. Write a program that uses a structure `7.2.1`

7.3 Explain the process of traversing an array/vector of structures to process data

1. Write functions/procedures that traverse an array of structures to process data `7.3.1`

7.4 Explain the logical steps to insert a value in a one-dimensional array or vector of structures

1. Write code that inserts values in an existing array or vector of structures `7.4.1`

7.5 Explain the logical steps in deleting elements in a one-dimensional array or vector of structures

1. Write code that deletes elements from an existing array or vector of structures `7.5.1`

Classes

8.1 Discuss class

1. Give an example of a class used in this unit `8.1.1`

8.2 Explain the advantages of using classes

1. Give an example of a useful class `8.2.1`

8.3 Discuss object

1. Give examples of data that would be contained in example classes and the methods that would be needed to manipulate that object `8.3.1`

8.4 Discuss instantiation

1. Write a program that instantiates an object of the class `8.4.1`

8.5 Explain member methods

1. Write a program that uses a member of the class `8.5.1`

Sequential Text Files

9.1 Discuss files

1. Give examples of when using files would be useful `9.1.1`

9.2 Explain the difference between sequential and random access of files

1. List the advantages of using sequential files vs. random-access files `9.2.1`

9.3 Explain the difference between opening text files for output, append, and input

1. Write programs that open files appropriately for output, append, and input `9.3.1`

9.4 Explain the logic of reading data sequentially from a text file

1. Write programs that read files `9.4.1`

9.5 Explain the terminator characters at the end of lines and end of the file

1. Write programs that read the data until the end of file (EOF) `9.5.1`

9.6 Explain the logic of writing data sequentially to a text file

1. Write programs that write data sequentially to a text file `9.6.1`

9.7 Explain the function of closing the file

1. Close files in all programs using files 9.7.1