CSCI-C 200

**Lab 6 – Array Operations**

**Description**

Get user input and create a 2-D array with at least 5 rows and 3 columns of data, then create a new array whose values are copied from the first array, plus a new column whose values are generated based on data in the first array (2-D array). E.g. Your program will ask an HR personnel to enter employee first name, last name, and date of birth (for at least 5 employees until an entry to indicate “done”). You will use an array to store all the information, then use a loop to copy all the information from the first array to the second array, at the same time, compute the age based on the date of birth, and store the age information into the fourth column of the second array. The new column values must be generated using a user-defined function.

Example 1:

Array 1:

John, Smith, 10/01/2000

Alex, May, 02/15/2003

…

Array 2:

John, Smith, 10/01/2000, 22 (fourth column contains age converted from date of birth)

Alex, May, 10/01/2000, 20

…

Example 2:

Array 1:

John, Smith, 95

Alex, May, 88

…

Array 2:

John, Smith, 95, A, (fourth column contains letter grade converted from the score)

Alex, May, 88, B+

You don’t have to use the above examples. You can use the same idea as in your lab 5.

**Rubric**

* 1. Program simulates a real-world scenario. (5’)
  2. Initial array values are taken from the user inputs and invalid entries are denied. (5’)
  3. Loops are used until the user indicate “done” after at least 5 sets of entries. (5’)
  4. The first 2-D array stores user entries and has at least 5 rows and 3 columns . (5’)
  5. Define at least one function to perform the task of generating values for the fourth column. (5’)
  6. Cannot hard code indexes other than index 0 and -1. (5’)
  7. Loops must be used to copy values from the first array to the second array. (5’)
  8. Display values from both arrays as the final output. (5’)
  9. Program runs successfully and generates correct and meaningful output. (30’)
  10. Follow professional coding standards (file headers, comments, code blocks, naming conventions, etc). (10’)
  11. Use pseudocodes to describe a detailed algorithm design in .docx file. Your code must follow the algorithm design. Your pseudocodes must be more similar to the Python language rather than the English language.
      1. Pseudocodes are used (5’)
      2. Algorithm design clearly illustrates ideas on how the problem is solved. (10’)
      3. Program follows the algorithm design. (5’)

**Submission**

Submit your executable file and utility file(s) on Canvas.