BİL 214 – System Programming

Homework #4

Assigned on 11.10.2022 – Due on 18.10.2022

- Submit one C source file per question.
- If a source file fails to compile with the gcc compiler, you get zero credits for that question.
- Make sure your submission file names are formatted as:
  FirstName_LastName_StudentID_HW#_Q#.c
  For example: **Toygar_Akgun_123456789_HW4_Q1.c**
  Toygar_Akgun_123456789_HW4_Q2.c …

1. [5 points] Write a C program that decomposes a given integer as powers of its prime divisors. For example, given 12 as input, its output should be $2^2 * 3$, or given 9 as input, its output should be 3^2. The way this output is displayed on the terminal is up to you, as long as it is easy to read and understand.

2. [10 points] Write a C program to swap two int arrays using pointers. Then write a random initializer to randomly fill two int arrays. When your program is run, it should first prompt the user for an array length that must be less than 20. Once the array length is given, your program should create, randomly initialize, and print two random int arrays. Finally, it should swap these arrays and print the final arrays.

3. [15 points] Write a C program to set all values of a 2-D int array to a given int value using one pointer. Then write a random initializer to randomly fill a 2-D int array. When your program is run, it should first prompt the user for an int value that will assigned to all array elements and the array dimensions that must be less than 10. Once the int value and the array dimensions are given, your program should create, randomly initialize, and print a 2-D int array. Finally, it should set all the array elements to the int value given by the user and print the final array.

4. [30 points] Write a C program to merge two sorted (increasing array index means increasing value) int arrays using pointers. Arrays can have different lengths. Then write a random initializer to create and randomly initialize two sorted int arrays. When your program is run, it should first prompt the user for two array lengths that must be less than 50. Once the array lengths are given, your program should create, randomly initialize, and print two sorted int arrays. Finally, it should merge these arrays and print the final sorted array.

5. [40 points] Go to the textbook's (Deitel & Deitel – C How to Program 8th Ed.) 6th Chapter and find the example given in Fig. 6.22 (Pages 282 – 284 for the 8th Ed.). Rewrite the code of Fig. 6.22 to use a menu-driven interface by using arrays of pointers to functions. The program should offer the user five options as follows:

```
Enter a choice:
0  Print the array
1  Find the minimum grade
2  Find the maximum grade
3  Print the average on all tests for each student
4  Print the modified average on all tests for each student
5  End program
```

One restriction on using arrays of pointers to functions is that **all the pointers must have the same type**. The pointers must be to functions of the same return type that receive arguments of the same type. For this reason, the functions in Fig. 6.22 must be modified so that they each return the same type and take the same parameters. Modify functions minimum and maximum to print the minimum or maximum value and return nothing. For option 3, modify function average of Fig. 6.22 to output the average for each student (not a specific student). Function average should return nothing and take the same parameters as printArray, minimum and maximum. For option 4, add a function called modifiedAverage to output the "modified" average for each student (not a specific student) such that the modified average values are computed by **excluding the maximum and minimum scores** (for each student). Function modifiedAverage should return nothing and take the same parameters as printArray, minimum, maximum and average. Store the pointers to these five functions in array processGrades and use the choice made by the user as the index into the array for calling each function.