The assignment will be graded out of 100 points.

Due: 11:59pm ending Monday July 2, 2018.

Submission Guidelines:

Assignment will be submitted via Blackboard. If there are multiple files that you are submitting then you should zip your files into a single file and submit it. The name of this file should be in following format: **lastname_firstname_UTANetID**. If you are submitting a single file, make sure it is in the following format: word, txt or pdf and it should be named as mentioned above.

Make sure your name and your student ID are listed in your assignments.

If your assignment is not completed by the deadline, send it anyway and review it with the TA for partial credit. Do not take a zero or excessive late penalties just because it isn't working yet. We will make an effort to grade you on the work you have done.

Note: Please do not forget to zip all the files in ONE folder

Example:

Task1.c
Task2.c
Task3.c
Task4.c
Task5.c
Task6.c

After that, copy all the C files into one folder and compress(zip) the folder. Then, submit the compressed (zipped) folder on Blackboard.

Assignment Specification:

Task 1 (20 pts.)

In a file called task1.c, write a C program that: You are given two arrays, **ArrayA** and **ArrayB**. **ArrayA** has a large enough buffer at the end to hold **ArrayB**. Write a “Function” to merge **ArrayB** into **ArrayA**. Then, write another “Function” to sort the final Array (bubble sort).

Implementation instructions:

- Declare one array of size 20 for **arrayA** and one array of size 5 for **ArrayB**.
- Initialize array **ArrayA** by inputting 15 integers (in any order), and Initialize **ArrayB** by inputting 5 integers (in any order). (Note: do not hard code the values of the arrays; user enters the values of two arrays)
- Merge **ArrayB** with **ArrayA**.
- Sort the updated array **ArrayA** (using bubble sort)
- Print out the updated (and sorted) array A (after merging with **ArrayB**).

Example:

If your input for **ArrayA** is:
1, 3, 11, 15, 16, 25, 34, 54, 56, 59, 64, 92, 98,4,6

and your input for **ArrayB** is:
2, 100, 5, 19, 40

Finally, after merging and sorting Arrays A and B, Array A becomes:
Expected Output: 1, 2, 3, 4, 6, 5, 11, 15, 16, 19, 25, 34, 40, 54, 56, 59, 64, 92, 98,100

Task 2 (20 pts.)

In a file called task2.c, write a C program to ask the user to enter two nouns and a verb, and saves them as variables called first_noun, second_noun, and verb. Then, print a sentence that looks like this: The first_noun verb over the second_noun.

For example: if the user enters nouns "cat" and "wall" and verb "jumps" your program output should look **EXACTLY** like this:

Example:

Enter the first noun: **cat**
Enter the second noun: **wall**
Enter a verb: **jumps**

Expected output:

The cat jumps over the wall.

Task 3 (20 pts.)

In a file called task3.c, write a C program to ask the user to enter a word.

- If word starts with a vowel, the program prints that it starts with a vowel.
- If word starts with a consonant, the program prints that it starts with a consonant.
- If word starts with neither a vowel nor a consonant (e.g., it starts with a number, or a punctuation character, or it is the empty string), the program prints that it starts with neither a vowel nor a consonant.

Examples:

if the user enters "123", your program output should look **EXACTLY** like this:

Please enter a word: 123
123 starts with neither a vowel nor a consonant.

if the user enters "Umbrella", your program output should look **EXACTLY** like this:

Please enter a word: Umbrella
Umbrella starts with a vowel.

if the user enters "cat", your program output should look **EXACTLY** like this:

Please enter a word: cat
cat starts with a consonant

Hint:

- vowels: A, E, I, O, and U (including their lower-case letters).
- consonant letters: B, C, D, F, G, H, J, K, L, M, N, P, Q, R, S, T, V, X, Z, W and Y (including their lower-case letters).

Task 4 (10 pts.)

In a file called task4.c, write a C program to ask the user to enter a word.

- Prints that word, with all vowels removed and replaced with letter Q.

Examples:

if the user enters "UTA", your program output should look **EXACTLY** like this:

Enter a word: UTA
output: QTQ

if the user enters "mountain", your program output should look **EXACTLY** like this:

Enter a word: mountain
output: mQQntQQn

Task 5 (10 pts.)

In a file called task5.c, write a C program to remove special characters and numerical values in String except alphabets. Consider any string with a length of 20 (MAX).

Example:
if the user enters "P2*ro-g!ra#mmi^n*g", your program output should look **EXACTLY** like this:

Enter a word: P2*ro-g!ra#mmi^n*g
output: Programming

if the user enters "UT%Ar#ling1ton!", your program output should look **EXACTLY** like this:

Enter a word: UT%Ar#ling1ton!
output: UTArlington

Task 6 (20 pts.)

In a file called task6.c, write a C program to calculate the mean, median and mode of a given array in the following manner:

- Write three functions mean (), median () and mode () that calculates the mean, median and mode of an array.
- Using these functions write a program that calculates all the three above mentioned values for two different arrays A and B where A is an array that contains the number of hours spent by a person each day for playing video games and B is an array that contains the number of hours spent by a person each day for his studies.
- Based on the above-mentioned values that you calculate using these functions compare the values obtained from both the arrays. If all the three values of array A is greater than array B print statement that warns the person to study. If all the three values of array B is greater than array A print statement that praises him for his studies. If all the three values are equal print statement that tells him he is well balanced. If any other issues, print “Sorry can’t come to an outcome”.

Hint: Sort the arrays first.

Example:

Let **array A** be: 4 5 5 2 8 9 Let **array B** be: 2 3 1 4 8 4
Mean of A is: 5.5 Median of A is: (5+5)/2=5 Mode of A is: 5
Mean of B is: 3.6 Median of B is: (3+4)/2=3.5 Mode of B is: 4

Here Mean, Median and Mode of A is larger than B.

Therefore, we print “You’re playing more video games you have to spent more time studying”.
In case if Mean, Median and Mode of A is lesser than B then you print “Great, you’re spending considerable time in studying”.

Assignment Guidelines:

There will be several programming assignments in this course, typically assigned on a weekly basis. All assignments will have equal weight. No assignment scores will be dropped. The following class policies regarding assignments will be followed:

- All assignments must be submitted via Blackboard.
- No deadline extensions for the entire class will be provided. (See syllabus about policy on extensions for individuals, based on emergencies documented in writing).
- No extra credit will be provided.
- If you make multiple submissions to Blackboard for the same assignment, only the latest submission will be graded.

Late submission policy:

- All assignments are graded out of 100 points. Assignments submitted late will be penalized, at a rate of 4 penalty points per hour. The submission time will be the time shown on Blackboard. Any assignment submitted more than 25 hours late will receive no credit.
- Exceptions to late submission penalties will only be made for emergencies documented in writing, in strict adherence to UTA policy. For all such exception requests, the student must demonstrate that he or she made all efforts to notify the instructor as early as possible.
- Computer crashes, network crashes, and software or hardware failure will NOT be accepted as justification for late submissions. If you want to minimize chances of a late submission, aim to submit early. You can always revise your submission till the deadline.
- Sometimes students submit the wrong files on Blackboard. Unfortunately, no credit or waiver of late penalties can be provided in such cases.
- If you find yourself in an emergency situation and cannot deliver homework on time, immediately inform the instructor and teaching assistant. Even if you have a valid reason for delivering late an assignment, you must make a convincing case that you have notified the instructor and teaching assistant as early as possible.

If you want to minimize chances of a late submission, aim to submit early. You can always revise your submission till the deadline (maximum 3 attempts).