The assignment will be graded out of 100 points.

Due: 11:59pm Wednesday, July 18, 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
Task7.txt

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

Assignment Specification:

For ALL the Tasks please get the values from user as input.

Task 1 (10 pts.)

In a file called task1.c, write a C program to demonstrate the use of & (address of) and * (dereference operator; value at address) operator. **NOTE**: The addresses in the memory (on your machine) will be different with the expected output.

Get the three values as input from the user.

Expected output:

Pointer: Demonstrate the use of **&** and ***** operator:

====================================

a = 50

ab = 20.10

abc = 's'

Using **&** operator:

=========================

address of a = 0x7ffda2feee48

address of ab = 0x7ffda2feee4c

address of abc = 0x7ffda2feee47

Using **&** and ***** operator:

=========================

value at address of a = 50

value at address of ab = 20.10

value at address of abc = 's'

Task 2 (20 pts.)

In a file called task2.c, write a C program to do the following with pointer:

1) Declare an array of pointers named **alpha** of 15 components of type int.
2) Output the value of the 12th component of the alpha.
3) Set the value of the 5th component of the alpha to 22.
4) Set the value of the 9th component of the alpha to the sum of the 7th and 14th components of the alpha.
5) Set the value of the 3rd component of the alpha to four times the value of the 9th component minus 33.
6) Output alpha so that three (3) components appear on each line.

Task 3 (15 pts.)

In a file called task3.c, write a C program (using a Function and Pointers) that the function has three inputs which are integers. The function returns true if the first number raised to the power of the second number equals the third number.

Example:

Input: 2 4 16
Expected Output: true

Input: 3 4 13
Expected Output: false

Task 4 (15 pts.)

In a file called task4.c, write a C program to find the largest element in an array (using Function and Pointers)

Example:

Int arr[15] = {2,3,6,43,22,11,34,67,43,21,77,46,32,23,10}

Expected output: 77

Task 5 (15 pts.)

In a file called task5.c, write a C function (using pointers) that has an input of a char value and returns a message to the user if the character is lower case or uppercase. It should return an appropriate message if the character is neither lowercase nor uppercase.

Example:

Input: **a**
Expected Output:
The character is lowercase

Input: **H**
Expected Output:
The character is Uppercase

Input: #
Expected Output:
Neither lowercase nor uppercase

Task 6 (15 pts.)

In a file called task6.c, write a C program to store n elements in an array and print the elements using **pointer**.
Get the array length and elements of the array as input from the user.

Example:
Input the number of elements to store in the array: **7**
Input **7** number of elements in the array:
10
11
4
18
13
30
9

Expected Output:
The elements you entered are:
Element [0]: 10
Element [1]: 11
Element [2]: 4
Element [3]: 18
Element [4]: 13
Element [5]: 30
Element [6]: 9

Task 7 (10 pts.)

In a file called **task7.txt**, describe the behavior of the following program. What values get printed? and **Why**?

```c
#include <stdio.h>
int main()
{
    int arr[] = {12,9,98,77,37,11,23,4,88,102,103,99};
    int *x = arr+4;
    int *ptr = &arr[9];
    arr[*ptr] ++;

    printf("First value: %d\n", *ptr );
    printf("Second value: %d\n", *x );

    *x = 5;
    printf("Third value: %d\n", arr[4] + *ptr);
    return(0);
}
```

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).