NORTH SOUTH UNIVERSITY

CSE115
Assignment

There are 5 problems in this assignment. Please read the problem statements carefully and try to match the output as best as you can. Each problem must be done in the same *'.c'* file which is provided to you.

Total Mark= 95 +5(viva)=100

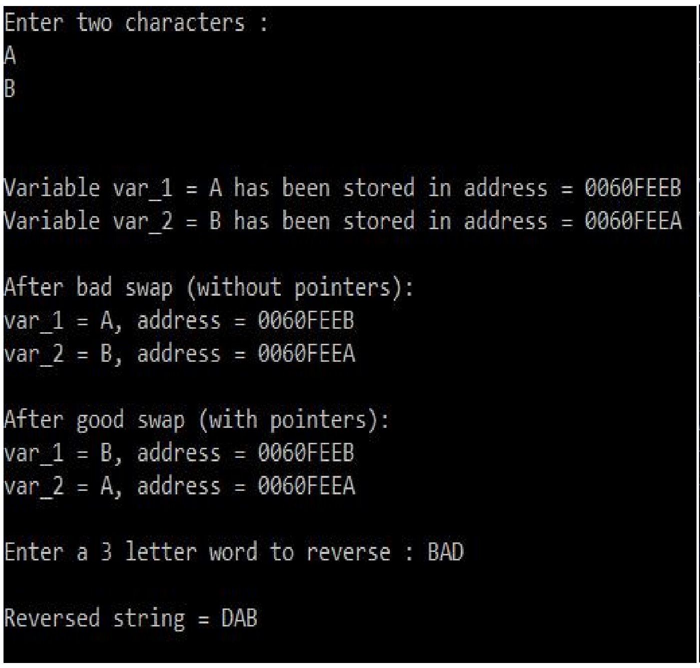
****

Problem 1 [15pts] : The problem_1.c file contains the code for a swap function called bad_swap. Check why the swap function does not work.

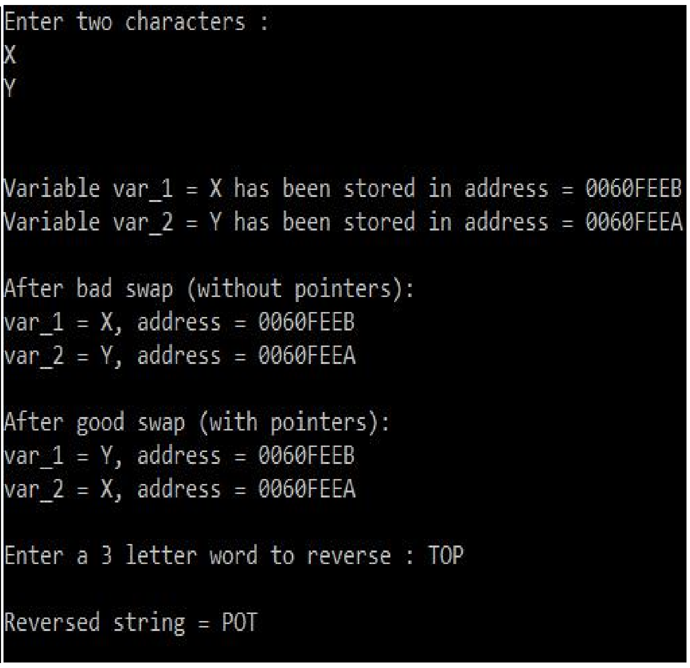
a) Write a swap function (good_swap) that can swap two character elements using pointers. [5pts]

b) In the main function declare two character variables then print them and their addresses. Now use these variables as parameters and apply the bad_swap and print the results(value of variables and addresses). Do the same with the good_swap and print the results. [5pts]

c) Write a function "reverse" that takes in a word of three letters as user input and reverses it (Swap the first and the last element). [5pts]



```
Enter two characters :
A
B


Variable var_1 = A has been stored in address = 0060FEEB
Variable var_2 = B has been stored in address = 0060FEEA

After bad swap (without pointers):
var_1 = A, address = 0060FEEB
var_2 = B, address = 0060FEEA

After good swap (with pointers):
var_1 = B, address = 0060FEEB
var_2 = A, address = 0060FEEA

Enter a 3 letter word to reverse : BAD

Reversed string = DAB
```



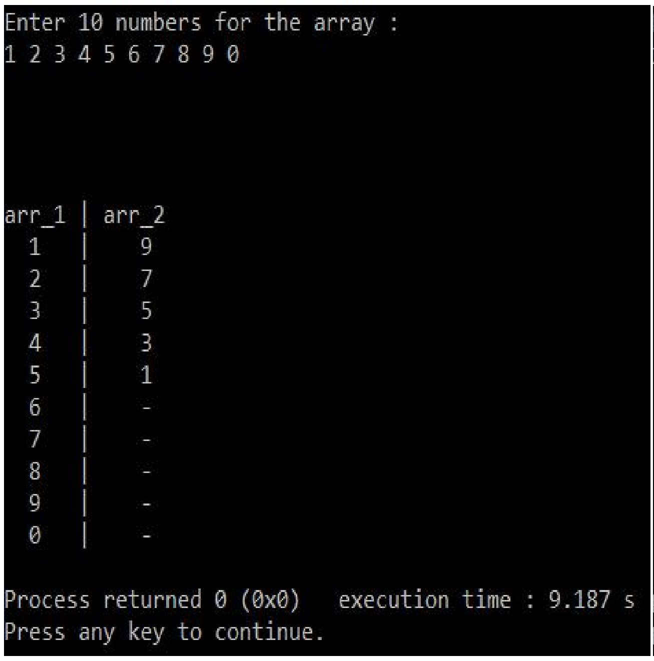
```
Enter two characters :
X
Y


Variable var_1 = X has been stored in address = 0060FEEB
Variable var_2 = Y has been stored in address = 0060FEEA

After bad swap (without pointers):
var_1 = X, address = 0060FEEB
var_2 = Y, address = 0060FEEA

After good swap (with pointers):
var_1 = Y, address = 0060FEEB
var_2 = X, address = 0060FEEA

Enter a 3 letter word to reverse : TOP

Reversed string = POT
```

Problem 2 [20pts] : Write the following code in the main function of the problem_2.c file.

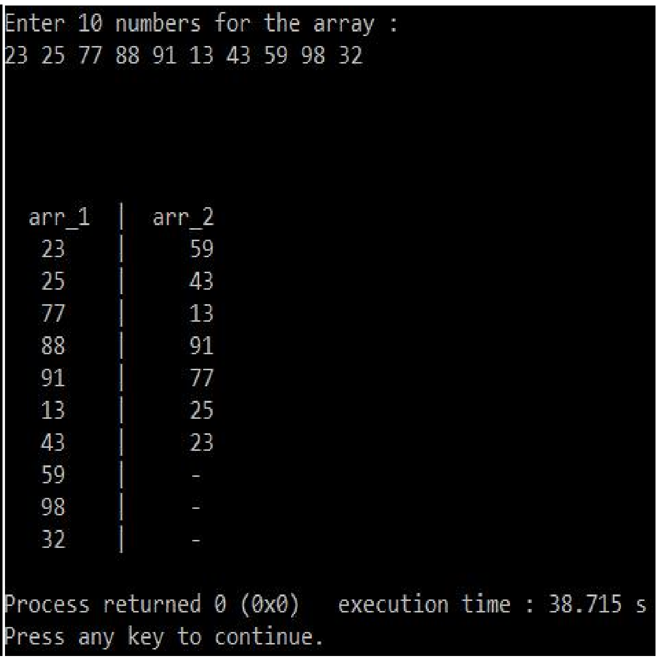
a) Write a program that takes in 10 numbers as user input and stores it in an array(arr_1). [5pts]

b) Then create another array(arr_2) that only stores the odd numbers from the first array(arr_1) in the reverse order. [Hint:The display has already been formatted. Change the variable names accordingly.]
As for example: if arr_1[] = {1,2,3,4,5,6,7,8,9,10} then arr_2[] = {9,7,5,3,1}. [5pts]

c) Assume you have two non-empty array of same size representing two non-negative integers. The most significant digit comes first and each of their cell contains a single digit. Add the two numbers and return the sum. Print the list. [Hint: Make sure to take care of the carry. Print it like sample output] [10pts]
Sample Input: A=[1,5,6,7,1] and B= [2,5,6,7,2]   Sample Output: C= [4,1,3,4,3]



```
Enter 10 numbers for the array :
1 2 3 4 5 6 7 8 9 0



arr_1 | arr_2
  1   |    9
  2   |    7
  3   |    5
  4   |    3
  5   |    1
  6   |    -
  7   |    -
  8   |    -
  9   |    -
  0   |    -

Process returned 0 (0x0)   execution time : 9.187 s
Press any key to continue.
```



```
Enter 10 numbers for the array :
23 25 77 88 91 13 43 59 98 32



  arr_1   |   arr_2
   23     |    59
   25     |    43
   77     |    13
   88     |    91
   91     |    77
   13     |    25
   43     |    23
   59     |    -
   98     |    -
   32     |    -

Process returned 0 (0x0)   execution time : 38.715 s
Press any key to continue.
```

Problem 3 [30pts] : The "my_file.txt" file contains the Name, ID, Course, Section of 2 students. Please read the statements carefully and start editing the problem_3.c file.

a) Edit the structure "info" to add four variables(arrays): Name, ID, Course, Section of size 50. [5pts]

b) Write a function "student_info" that will take input for the structure "info". It will take "N" number of entries where "N" is a user input that has been declared globally and has been initialized at the beginning of the main function(see problem_3.c file). Each entry comprises of Name, ID, Course and Section. The input is given by the user and stored in the structure array declared globally(S[100]).[5pts]

c) Write a "printStr" function that takes in a character array as a parameter and prints it. [5pts]

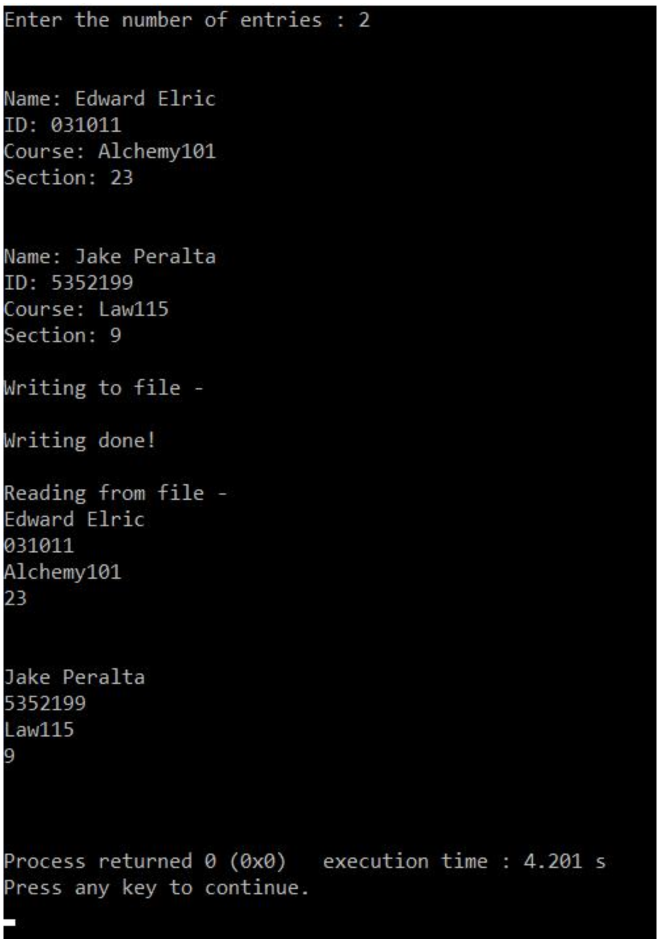
d) Write a "read_from_file" function that reads from the "my_file.txt" file and prints each line from the file. [5pts]

e) Write a "write_to_file" function that writes to the "my_file.txt" file from the structure. Write three or more more student information(Name, ID, Course, Section) using your function.[5pts]

f) (**Most Important**)Make sure the last entry is your own Name, ID, Course and Section. [5pts]

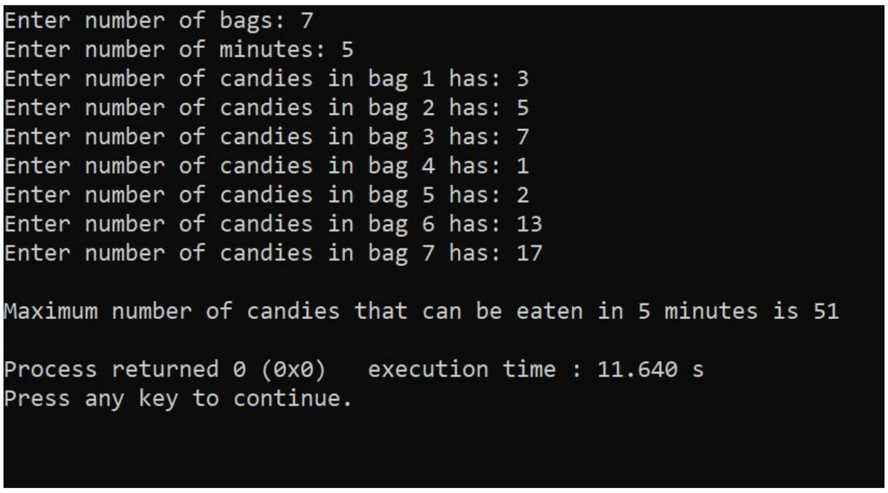
Note:
*A variable of struct "info" has been declared globally(S[100]), so you don't need to pass or return any structures.
*You can use the globally declared N variable to keep track of the entries. You can update it at the beginning of the program and access it from other functions.
*Both your read_from_file and write_to_file can be of void type and you can get by without passing any parameters to these.

*If you're stuck look into - fflush(), fgets(), fputs(), getc(), getchar(), scanning string input with white-space and appending files.



```
Enter the number of entries : 2


Name: Edward Elric
ID: 031011
Course: Alchemy101
Section: 23


Name: Jake Peralta
ID: 5352199
Course: Law115
Section: 9

Writing to file -

Writing done!

Reading from file -
Edward Elric
031011
Alchemy101
23


Jake Peralta
5352199
Law115
9



Process returned 0 (0x0)   execution time : 4.201 s
Press any key to continue.
```

Problem 4 [10pts] : Please use the display function provided in the problem_4.c file.
You have N magical bags of candies in front of you. The *i* th bag has A_i candies in it. It takes you one minute to finish a bag of candies,regardless the number of candies in it. Every time you finish a bag with *X* candies in it, the bag is magically replenished with *X/2* (rounded down to the nearest integer more candies.) Write a program that determines the maximum number of candies you can eat in K minutes.
The input is a sequence of integers. The first integer N is the number of bags. The next integer K is the number of minutes you have. The next N integers is the number of candies in the bags. The output of your program is a single integer which represents the maximum number of candies you can eat.



```
Enter number of bags: 7
Enter number of minutes: 5
Enter number of candies in bag 1 has: 3
Enter number of candies in bag 2 has: 5
Enter number of candies in bag 3 has: 7
Enter number of candies in bag 4 has: 1
Enter number of candies in bag 5 has: 2
Enter number of candies in bag 6 has: 13
Enter number of candies in bag 7 has: 17

Maximum number of candies that can be eaten in 5 minutes is 51

Process returned 0 (0x0)   execution time : 11.640 s
Press any key to continue.
```

Problem 5 [20pts] Solve the following problems using recursion. The functions are given in problem_5.c file, in addition to a function that can generate random function. Use the function to generate random inputs.

a) A) Write some code to find GCD (HCF) of two numbers using recursion.[5pts]

int GCD (int a, int b)

b) Write a recursive function that converts a decimal number to a binary number.[5pts]

int DecToBin(int dec);

c) Write a recursive function that finds the sum of the following series.[5pts]

$$1 + \frac{1}{2} + \frac{1}{4} + \frac{1}{8} + \dots + \left(\frac{1}{2}\right)^n$$

d) Study the following sequence 3, 1, 3, 7, 11, 21, 39, 71, 131, 241, 443, 815, ...

In mathematical terms, the sequence T_n is defined as follows:

$$T_n = T_{n-1} + T_{n-2} + T_{n-3} \quad \text{(for n >= 0)}$$

with initial values,

$$\mathbf{T_0 = 0,\ T_1 = 1,\ T_2 = 0}$$

Now take an integer input *'n'* from the user and print the sequence up to *'n'*.[5pts]

☺