Department of Computer Science and Engineering

Assignment 7

Subject : Programming and Data Structure (CS19003)

Instructions:

- Give the name of the programs files as <Your roll>_<assignment number>_<question number>.c. For example, 21XX12345_A1_Q1.c for question 1 of assignment 1 of a student with roll 21XX12345. **Follow the file naming convention strictly**.
- Apart from the main .c file for each program, you should also upload one additional temporary .c file for each program (such as when you have finished half of the code). The naming for the temporary file should be in the format <Your roll>_<assignment number>_<question number>_temp.c. For example, 21XX12345_A1_Q1_temp.c **Make sure that your main code do not deviate much from its temporary code for each program**.
- You should upload the main .c file and the temporary .c file individually to the Moodle course web page once you finish answering each question. No need to zip the files.
- The **deadline** to upload the programs is 12:00PM. Beyond that, submission will be closed (No extensions will be granted).
- If you do not follow the instructions, your marks may be deducted.

[Total Marks = 100 (30 + 35 + 35)]

Answer all questions

1. A country has ‘n’ number of zoos, where each zoo has a certain number of tigers, lions and elephants. It is required to perform some analysis on the number of animals within each zoo of the country. Declare a structure named ‘zoo’ that contains three integer type values, each to hold the number of tigers, lions and elephants within that zoo. Also declare an array ‘zoo_count[ ]’ of size n, which is an array of structure of the type ‘zoo’ that holds the information about all the n zoos of the country. Take, ‘n’ as input from the keyboard and fill out the array ‘zoo_count[ ]’ taking the number of tigers, lions and elephants of each zoo as user input. Note that zoo_count[0] holds the number of animals (tigers, lions and elephants) of the first zoo, zoo_count[1] holds the number of animals of the second zoo and so on. Also as zoo_count[ ] is an array of structures, each element of the array is actually a structure of type ‘zoo’. After you have taken input, perform the following three operations and print the result.

   - Print the total number of tigers in all the zoos within the county.
   - Print all the zoo numbers (i.e., the index of the zoo_count[ ] array) that contains less than or equal to 6 lions.
   - Print the zoo numbers (i.e., the indices of the zoo_count[ ] array) according to the increasing order of the number of elephants contained within each zoo.

   Example:

   Input:

   Enter ‘n’ : 3

```
Enter for zoo 1
Tigers - 4
Lions - 3
Elephants - 10

Enter for zoo 2
Tigers - 2
Lions - 8
Elephants - 5

Enter for zoo 3
Tigers - 8
Lions - 5
Elephants - 4
```

Output:

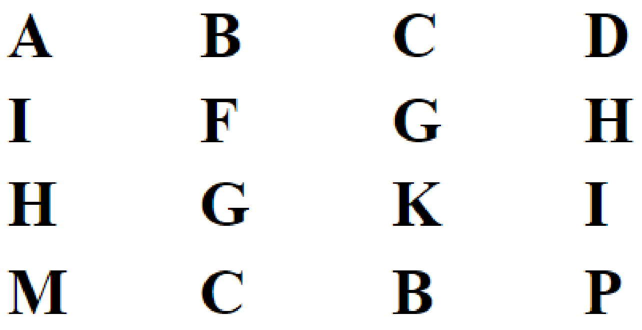
```
Total number of tigers = 14
Zoos with lions less than or equal to 6 = 1, 3
Zoo indices as per increasing order of elephants = 3, 2, 1
```

[30]

2. A square 2D character array of size nXn is called 'character-mirrored' if it satisfies the following condition.

   - Consider all the left diagonals of the array except the principal left diagonal, the upper right corner element of the array and the lower left corner element of the array. A **symmetric pair** of left diagonals is such a pair which consists of an upper left diagonal and a lower left diagonal, both separated equally from the principal left diagonal. If, for **ALL** symmetric pairs of left diagonals of the array, the characters (elements) of upper left diagonal of each symmetric pair is equal to the reverse order of the characters (elements) of its corresponding lower left diagonal of that symmetric pair, then the array is called 'character-mirrored', otherwise it is not 'character-mirrored'.

   For example, consider the 4x4 character array below:



A	B	C	D
I	F	G	H
H	G	K	I
M	C	B	P

Now consider the same array with different color codings to help you understand

A	B	C	D
I	F	G	H
H	G	K	I
M	C	B	P

Principal left diagonal = [A, F, K, P] (shown in red)

Upper right corner element = D (highlighted in blue)

Lower left corner element = M (highlighted also in blue)

1^{st} symmetric pair (highlighted in yellow both above and below the principal left diagonal):

Upper left diagonal = [B, G, L]

Lower left diagonal = [L, G, B]

2^{nd} symmetric pair (highlighted in green both above and below the principal left diagonal):

Upper left diagonal = [C, H]

Lower left diagonal = [H, C]

Now consider the two symmetric pairs. For the 1st pair the characters in the upper left diagonal are exactly the reverse of the characters of the lower left diagonal. For the 2^{nd} symmetric pair also, the same observation can be made. Hence this particular 2D character array is 'character-mirrored'.

Write a C program to input an n X n 2D character array and print all pairs of symmetric left diagonals. Also, check whether the 2D array is 'character-mirrored' or not. **[35]**

3. Consider a date calculator that performs the following operations as per the following C functions -

   i leap_year(date) - checks if the date belongs to a leap year or not, returns 1 if date is in leap year, returns 0 otherwise.

   ii day_identify (date) - identify the day of the date i.e., Monday, Tuesday... It does not return anything, prints the day within the function.

   iii _diff(date1, date2) - calculates the difference between two dates passed as arguments in the number of days and returns it.

   iv date_identify(date1, n)- identify the date that is after n days from the given date. This returns the final date.

Write a program that performs all the above operations as per the menu selected by the user (see example below). For storing the date, **use a structure** named date that contains three members of integer type: namely day, month and year. You must declare a function for each of the above mentioned operations. For taking input the dates, you can follow the example shown below. No need to keep a separate logic that checks whether the input date is valid or not, you may assume that the user always enters a valid date.

Example 1:

Input:

− − − − − − − −−MENU− − − − − − − −−
1. Check leap year
2. Identify corresponding day of a given date
3. Compute difference between two dates
4. Calculate the next date after n days from a given date

Enter your option:
1

Enter date:
Day -12
Month -5
Year - 2022

Output:
The given date is does not belongs to a leap year

Example 2:

Input:

− − − − − − − −−MENU− − − − − − − −−
1. Check leap year
2. Identify corresponding day of a given date
3. Compute difference between two dates
4. Calculate the next date after n days from a given date

Enter your option:
2

Enter date:
Day - 12
Month - 1
Year - 2022

Output:

The corresponding day of the given date is Wednesday

Example 3:

Input:

− − − − − − − −−MENU− − − − − − − −−

1. Check leap year
2. Identify corresponding day of a given date
3. Compute difference between two dates
4. Calculate the next date after n days from a given date

Enter your option:

3

Enter first date:

Day - 12

Month - 2

Year - 2022

Enter second date:

Day - 29

Month - 1

Year - 2022

[NOTE:- The first date need not necessarily always be before the second date as per calendar order. Please consider this fact]

Output:

There are 14 days between the given two dates.

Example 3:

Input:

− − − − − − − −−MENU− − − − − − − −−

1. Check leap year
2. Identify corresponding day of a given date
3. Compute difference between two dates
4. Calculate the next date after n days from a given date

Enter your option:

3

Enter first date:

Day - 12

Month - 2

Year - 2022

Enter second date:

Day - 29

Month - 1

Year - 2022

[NOTE:- The first date need not necessarily always be before the second date as per calendar order. Please consider this fact]

Output:

There are 14 days between the given two dates.

Example 4:

Input:

− − − − − − − −MENU− − − − − − − −

1. Check leap year
2. Identify corresponding day of a given date
3. Compute difference between two dates
4. Calculate the next date after n days from a given date

Enter your option:

4

Enter date:

Day - 28

Month -1

Year - 2022

Enter number of days: 12

Output:

The corresponding date after 12 days is 9/2/2022 [Print in format dd/mm/yyyy] **[35]**