CS218 Data Structures

Assignment #01

Instructor	**Dr. Hashim Yasin**
Session	**Fall 2022**

Instructions:

1) You are required to submit a **Microsoft Word** file, containing all your codes alongwith screenshots (complete console) of every question next to it. Any question without the screenshot will not be accepted. PDF or other format files will not be accepted.

2) Late submissions will **not be accepted.**

3) Word file format should be "**Roll-Number-Section-AsNo**", for example *19F-0101-5B-As #01*. Marks will be deducted for not following the correct format.

4) Keep the questions in order.

5) Plagiarism will not be tolerated, either done from internet or from some fellow classmate of same/other section. Plagiarized questions will result in **straight zero** or **negative marking.**

Problem 1

Take inputs in a user defined array. After that if the input is even, place it at an even index and if the input is odd, place it an odd index. If the user puts in an even integer and all even indexes are occupied, ask user to enter an odd integer and vice versa. If all the even and odd entries are filled, then notify the user that the program has ended.

Inputs: 9 8 5 7 4 2 1
Output: 8 9 4 5 2 7 – 1
Index: 0 1 2 3 4 5 6 7

Problem 2

Create a user defined array and then take inputs. Print all the Divisors of every number user gave as input.

Inputs: Size: 3
Elements: 4 5 6
Output:
4: 1 2
5: 1
6: 1 2 3

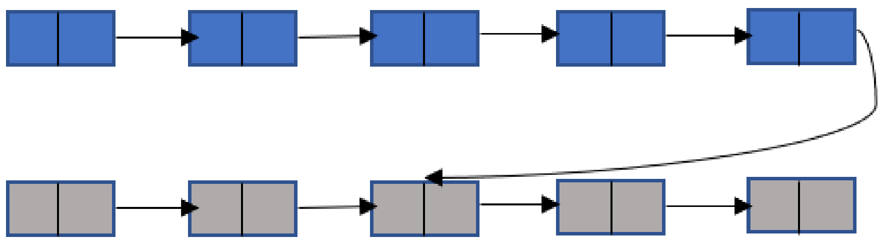
Problem 3:

Your task is to input two sorted singly linked lists in *descending order*. Change the next pointers to obtain a single, merged linked list which also has data in descending order. Note that either head pointer given may be null meaning that the corresponding list is empty. *You must use the following function prototype,*

node* merge_lists(node *, node *);

Problem 4:

Your task is to take two singly linked lists let suppose A and B. The last index of A is connected to 3rd index of the B. Find the common elements in both linked lists. For more idea see the figure below.



Problem 5:

Make a doubly linked list and done all the things mention below. The node will have an int variable in the data part. Your LinkedList will have a head and a tail pointer.
Your LinkedList class must have the following functions:
1) Insert a node
void insertNodeAtBeginning(int data);
void insertNodeInMiddle(int key, int data); //will search for key and insert node after the node where a node's data==key
void insertNodeAtEnd(int data);

2) Delete a node
bool deleteFirstNode(); //will delete the first node of the LL
bool deleteNode(int key); //search for the node where its data==key and delete that particular node
bool deleteLastNode(); //will delete the last node of the LL

3) Searching

4) Display

Note: The program must be Menu Driven and completely generic, especially for deleting/inserting the middle nodes.

Problem 6:

Write a program that prompts the user to input a string and then outputs the string in the pigLatin form. Input the string in a **doubly circular link-list**. The rules for converting a string into pig Latin form are as follows:
If the string begins with a vowel, add the string **"-way"** at the end of the string. Forexample, the pig Latin form of the string **"eye"** is **"eye-way"**.

If the string does not begin with a vowel, first add **"-"** at the end of the string. Then rotate the string one character at a time; that is, move the first character of the string tothe end of the string until the first character of the string becomes a vowel. Then add the string **"ay"** at the end. For example, the pig Latin form of the string **"There"** is **"ere-Thay**

Strings such as **"by"** contain no vowels. In cases like this, the letter **y** can be considered a vowel. So, for this program the vowels are **a, e, i, o, u, y, A, E, I, O, U,and Y**. Therefore, the pig Latin form of **"by"** is **"y-bay"**.

Strings such as **"1234"** contain no vowels. The pig Latin form of the string **"1234"** is**"1234-way"**. That is, the pig Latin form of a string that has no vowels in it is the string followed by the string **"-way"**.

Note: Your program must store the characters of a string into a linked list and use the function rotate, to rotate the string.

Problem 7:

Round-Robin algorithm is an algorithm used in CPU Scheduling. In this algorithm, all processes take turns and run for a specific interval, until they are all completely executed.You are required to make a process that will consist of two things,
ID *(must be unique)*
Execution time

Each process will take a certain amount of time to execute completely, this time will bestored in the "execution time" variable.
You are required to implement a **circular linked list** that will store *user-entered* number of processes (say *n*). You will also input a *time slice* from the user, this will be the time every process will get in one turn.
You will then create a function that will execute every process in the circular linked list using Round-Robin algorithm and store the total time taken by each process to execute. Your final output will be the time required by each process to completely execute.

A **sample output** would be like this

```
Enter number of processes: 4

Enter Process #1 Execution Time: 2
Enter Process #2 Execution Time: 3
Enter Process #3 Execution Time: 9
Enter Process #4 Execution Time: 5

Enter Time Slice: 3

Processes in Linked List: 1 2 3 4 - Process #1 executed for 2 seconds.

Process #1 has been executed. Time Taken: 2

Processes in Linked List: 2 3 4 - Process #2 executed for 3 seconds.

Process #2 has been executed. Time Taken: 5

Processes in Linked List: 3 4 - Process #3 executed for 3 seconds.

Processes in Linked List: 3 4 - Process #4 executed for 3 seconds.

Processes in Linked List: 3 4 - Process #3 executed for 3 seconds.

Processes in Linked List: 3 4 - Process #4 executed for 2 seconds.

Process #4 has been executed. Time Taken: 16

Processes in Linked List: 3 - Process #3 executed for 3 seconds.

Process #3 has been executed. Time Taken: 19

Execution Completed!
```