ESS101 : Programming 1 (C Programming)
LAB - 9

Due: 30-Oct-2019, 5 pm

Problem 1 (Usage of `argc, argv`): Write a (C) program `expr` which evaluates an operation from command line inputs, where each operator or operand is a separate argument. For example:

```
expr 20 30 +
```

evaluates 20 + 30 and

```
expr 5 10 *
```

evaluates 5 * 10. Print out the result.
NOTE: Use command line arguments, namely, `argc and argv`, to read in the operators and operand. Make sure you check for the number of arguments before evaluating them.

Problem 2: Write a program to input two complex numbers (real and imaginary parts) and store them as elements of structure variables (use floating point numbers - precision 2 places after decimal). Perform addition, subtraction, multiplication on them and print the result appropriately. Note that there are spaces before and after the plus separating the real and imaginary parts.

Sample Input:
2.43 9.68
-2.43 -9.68
Sample Output:
0.00 + 0.00i
4.86 + 19.36i
87.80 + -47.04i

Problem 3: Define a structure in (C) to hold a name (string), roll number (string), age (integer) and marks (integer) of n students. Write a program to input details of n students and store them in an array of structures. The first line of input stands for the number of students followed by the format: name, roll, age, marks. Write a program to print all the input details.

Sample Input:
1
abcdef imt2019999 20 100
Sample Output:
abcdef
imt201999
20
100

Problem 4: Write a (C) program to input integers and create a linked list of the integers. Since this is a linked list where memory needs to be allocated dynamically the count of integers is not needed in the beginning.
First, Walk the list and print the integers in the linked list.
Next, delete all the odd valued integers. Print the new list (remember it may end up being a NULL list).
NOTE: You need to create a node structure which has atleast two fields : one to store the integer value, another as a pointer to a neighboring node that contains the next integer and so on. The last node points to a NULL pointer indicating end of the list.

Sample Input:
5
1 3 6 7 8
Sample Output:
1−− >3−− >6−− >7−− >8−− >NULL
6−− >8−− >NULL
Note: There is no space between the numbers, the hyphens and the equality symbol.