Basic questions to improve your coding skills : Set 2

1. Menu-driven program to do the following conversions. Use **functions** for each conversion.

i) decimal to binary	ii) binary to decimal
iii) decimal to hexadecimal	iv) hexadecimal to decimal

2. Write a C program to implement recursive **Binary Search** algorithm to search if an integer is present in an array or not. If present, print the position of the element in the array.

3. Write a function that implements the **Euclidean Algorithm** to return the GCD of two given positive integers. Use this function to find the LCM of two given positive integers.

4. Write a function that counts the number of times an item *X* appears among the first *N* elements of an array *A* and returns that count as the frequency of *X* in *A*.

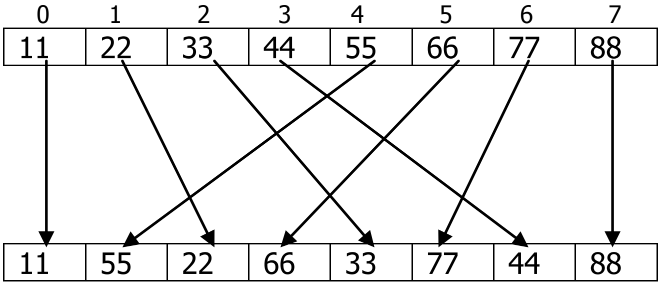
5. Consider an integer array of size *N*, whose values are generated using a random number generating function. Write a C program that sorts the elements from the smallest to the largest value using the **Bubble Sort** algorithm.

6. Implement the **Sieve of Eratosthenes** to find prime numbers less than *N*.

7. Given a string as input, write a function that prints the number of occurrences of each of the following : lower case letters, upper case letters, numbers and special characters.

8. WAP to input two strings into str1 and str2 respectively. Check if str2 is a substring of str1. Do NOT use in-built function to check for substring.

9. WAP to implement the **Perfect Shuffle** of a 1D array with an even number of elements. For example, it would replace {11,22,33,44,55,66,77,88} with {11,55,22,66,33,77,44,88)



10. WAP to delete all duplicate characters in a given string.
Eg : i/p : ABRACADABRA , o/p :ABRCD

11. Write a program that includes a function which rotates 90degree clockwise a two-dimensional square array of integers.

```
Eg :  i/p :    10 20 30          o/p :   70 40 10
               40 50 60                  80 50 20
               70 80 90                  90 60 30
```

12. Write a function that attempts to delete an item from an array.
int delete(int A[], int N, int X)
The function searches the first *N* elements of the array *A* for the item *X*. If *X* is found, it is deleted, all the elements above that position are shifted down, *N* is decremented, and *1* is returned to indicate a successful removal. If *X* is not found, the array is left unchanged and 0 is returned to indicate "failure."

13. Write a function that "rotates" the first *N* elements of the array *A*, *k* positions to the right (or *k* positions to the left if *k* is negative). The last *k* elements are "rotated" around to the starting of the array.
Eg: i/p : 22 33 44 55 66 77 88 99 and k=3

```
    o/p :  77 88 99 22 33 44 55 66
```

Note that k=-5 would give the same output.

14. WAP that compares 2 two-dimensional integer arrays of the same size and assigns -1, 0, or 1 to each element of a third array of the same size according to whether the corresponding element of the first array is less than, equal to, or greater than the corresponding element of the second array.
Eg :

```
Matrix 1 :  22  44   66          Matrix 2 :   33  44    55
            50  50   50                       50  50    80
```

then the third array will be

```
- 1     0       1
  0     0      -1
```

14. WAP to find the **Saddle point** in a 2D array.

15. Write a program that reads a sequence of names, one per line, and then sorts and prints them.

16. Write a program that reads one line of text and then prints it with each word reversed. Eg :

```
        i/p :   Today is Tuesday
        o/p :    yadoT si yadseuT
```

18. Implement solution to **Tower of Hanoi** puzzle.