1. Write a program to find the sizes of various data types available in C

2. Write a program to count the number of one bits and zero bits in an unsigned integer

3. Write a program to find out whether a given number is a power of 2 or not (use bit wise operations)

4. Write a program to accept “N” integers from the user. ‘N’ has to be taken from the user. Enter the numbers one at a time and find out the count of +ve numbers, -ve numbers and zeros.

   The program should not accept non-integer value. If a non-integer value is entered, user must be asked to re-enter. Use the return value of scanf to find out whether the user entered integer or not. You would also have to clear the buffer before taking the next input

   Use the following to clear the buffer

```
While (‘\n’ != getchar())
{
        ;
}
```

5. Implement the calculator for addition, subtraction, multiplication and division with the following functions

- Initialize a variable 0. Get an input and operation from the user. This operation has to be performed on initial variable using user input. The result has to be used for subsequent operations.
- Implement functions for different operations
- Display a menu for different operations and exit. The program exits when user chooses exit
- Use global variables, only if necessary

- Create a header file, implementation file and interface file

6. Write macro definitions with arguments for calculation of area of a triangle, square and a circle

Use macros for both constants as well as formula evaluations

Store these macro definitions in a header file and invoke the macros from main function

7.

a. Write a function that converts Fahrenheit temperatures to Celcius.

b. Write a function that determines whether a given positive integer is a prime

c. Write a a function that reverses the digits of a positive integer (for example reverse (49025) would return 52094.

8. Write program to test the following concepts on Pointers

    - Pointer to Constant, Constant Pointer
    - Types of Pointers and their Sizes
    - Operations on Pointers (++, --, subtraction of one pointer from another)
    - Void Pointers
    - Double Pointers, Call By Value, Call by Reference

9. A) Write a function that finds maximum, second maximum value of an array of integers.

    b) Write a function that rotates an array of integers a given number of positions.
    c) Write a function that determines whether a given array is sorted.
    d) Given two arrays A and B. A contains 10 elements and B contains 15 elements. Write a function to create an array C that contains elements that are common to A and B.

10. Write a program to check whether a given matrix is a magic square or not

- A matrix is a magic square if all column sum’s, row sum’s and diagonal sum’s are equal

- It is a square matrix

11. Accept a number say ***max*** and a string say ***pattern*** as command line arguments. The program must take a maximum of max strings from the user. If the user enters “end” before reaching “max” that should be treated as end of input.

- Accept the user input into an aaray. The same array has to used accept strings from user always
- Copy all strings that has pattern in it into an array of pointers (Use strstr for the same). Allocate appropriate memory of each pointer. Array should be dynamic
- Display the count of strings that has pattern in it
- Concatenate the last string onwards to the first string and display the result

12. Accept 3 file names as command line arguments. The first two are input files in which the first file has to be created as an integer file and the second as a string file. Merge the contents of these 2 files into the 3rd file. It should be one integer from first file followed by a line from second file. Display the merged file.

13. Define a structure which stores the details of the employees like ID, name, department and phone number. Get the details from the user into an array of structures. Stores the employees data in a binary file. Now perform the following operations:

- User should have an option to get the record by specifying the name
- User should be able to update the phone number of a record specifying the record number

14. Write a program to implement a singly linked list for simple structure only an integer. The program should have the following functionality
    - Insert at the beginning of linked list
    - Insert at the end of the linked list
    - Insert at the proper position in list (maintain sorted list)
    - Delete from the beginning of the list
    - Delete from the end of the list
    - Delete a node from sorted linked list
    - Display the content of list after each addition and deletion
    - Free the list before the end of program

15. Modify Program 5 implement a doubly linked list for simple structure only an integer. The program should have the same functionality as 5.

16. Write a program to store a polynomial in a single circular list. Display the list. Now write a function to find it derivaties and store in another circular linked list.

17. Write a program to do the following
    1. Accept ‘n’ names, add them in a list

    2. Accept a number as lucky number say ‘m’

    3. Delete mth name

    4. Repeat 3rd step till there is only 1 name remaining in the list

    5. Declare that name as winner

18. Write a program to implement a stack using linked list to find different palindrome words in a file
    - Create a file with some English sentences
    - Get the filename as input
    - Identify the palindrome words amongst the file (read one word at a time and check) using stack
    - Write the palindromes into another file and display from the file

19. Write a program to Convert an Infix expression(Q) to Postfix expression(P).

    1. Push “(“ onto STACK and add “)” to the end of Q

    2. Scan Q from left to right and repeat steps 3 to 6 for each element of Q until STACK is empty

    3. If an opernad is encountered, add it to P

    4. If a left parentheses is encountered , push it onto STACK

    5. If an operator **op** is encountered, then:

        a) Repeated pop from STACK and add P to each operator(on top of STACK) which has same or higher precedence than **op.**

        b) Add **op** to STACK

    6. If a right parenthes is encountered, then:

        - Repeated pop from STACK and add P to each operator (on top of STACK) until a left parentheses is encountered

        - Remove the left parenthesis (don’t add to P)

20. Write a program to implement a Queue using circular array(Queue of names)

21. Write a program to implement Priority Queue using linked list.

22. Implement binary search to search for a name in a set of names. The names should be sorted using bubble sort

23. Implement the following sorting techniques on an array of integers/names
    - Insertion Sort
    - Quick Sort
    - Merge Sort
    - Selection Sort
    - Shell Sort
    - Heap Sort

24. Write a C program to achieve the following
    a. Read the name of one input file and 1 output file through command line.
    b. Read all the words from the input file, remove duplicates and then store the words in the output file one on each line.
    c. Please think of good method to remove duplicates.

25. Implement binary search tree using linked representation
    - Write functions for search, insertion, deletion, tree traversal (preorder, inorder, postorder)

26. Implement AVL trees – Write functions for insertion, deletion and search of elements in AVL tree (perform rotations when required).

27. Write a program which take ten strings from the user as input one by one. Declare a hash table of size 10. Use ASCII value addition method as hash function. Store the strings in array of strings and find how many collisions occur.

28. Implement a hash table for the following

```
struct device

{

    int device;

   char * device_type;
}

struct presence

{

    int p_id;

    char *name;

    struct device d;

}
```

Create a hash table of size 5 of presence. Hash function should be on p_id. Chaining should be used for collision resolution.

29. Write a Graph in memory using Linked list.

a) Insert a node into Graph.
b) Insert an edge into Graph.
c) Display all edges from a node.
d) Delete an edge.
e) Delete a node.
f) Find and store the indegree and outdegree of each node of the Graph.

30. Create an Adjacency list representing a Graph. Ask the user to enter starting node.
a) Print list of nodes traversed using Breadth First Traversal from starting node
b) Print list of nodes traversed using Depth First Traversal from starting node