

# LAB WEEKLY PROGRAMS OF COMPUTER SCIENCE:



SUBMITTED BY: [Aman Singh](#)

University Roll no. : [2315000237](#)

Section : [AY-II](#)

Branch: [BTech cse](#)

## SUBMITTED TO:

MS. GURPREET KAUR

### WORKSHEET WEEK 1

Q1. Write a program to accept height and base of triangle and calculate area of Triangle Note:  $\text{area} = (h*b)/2$ .

Ans

```
#include <stdio.h>
```

```
int main() {
```

```
    float height, base;
```

```
    printf("Enter the height of the triangle: ");
```

```
    scanf("%f", &height);
```

```
    printf("Enter the base of the triangle: ");
```

```
    scanf("%f", &base);
```

```
    float area = (height * base) / 2;
```

```
    printf("The area of the triangle is: %.2f\n", area);
```

```
    return 0;
```

```
}
```

```
Enter the height of the triangle: 4
```

```
Enter the base of the triangle: 8
```

```
The area of the triangle is: 16.00
```

```
-----
```

```
Process exited after 7.069 seconds with return value 0
```

```
Press any key to continue . . .
```

Q.2 Write a program to accept radius of circle and calculate area of circle Note:  $\text{area} = \pi * r^2$

Ans.

```
#include <stdio.h>
```

```
#include <math.h>
```

```
#define PI 3.14
```

```

int main() {

    float radius;

    printf("Enter the radius of the circle: ");
    scanf("%f", &radius);

    float area = PI * pow(radius, 2);
    printf("The area of the circle is: %.2f\n", area);

    return 0;
}

```

```

Enter the radius of the circle: 7
The area of the circle is: 153.86

-----
Process exited after 7.951 seconds with return value 0
Press any key to continue . . . |

```

Q. Write a program to find the lowest marks of three students using conditional operator.

Ans.

```
#include <stdio.h>
```

```

int main() {

    int marks1, marks2, marks3;

    printf("Enter marks for student 1: ");
    scanf("%d", &marks1);

    printf("Enter marks for student 2: ");
    scanf("%d", &marks2);
    printf("Enter marks for student 3: ");
    scanf("%d", &marks3);

```

```

    int lowestMarks = (marks1 < marks2) ? ((marks1 < marks3) ? marks1 : marks3) : ((marks2 < marks3) ?
marks2 : marks3);

    printf("The lowest marks among the three students is: %d\n", lowestMarks);

    return 0;
}

```

```

Enter marks for student 1: 23
Enter marks for student 2: 43
Enter marks for student 3: 27
The lowest marks among the three students is: 23

-----
Process exited after 6.824 seconds with return value 0
Press any key to continue . . . |

```

Q. Write a program to Calculate Compound Interest.

Ans.

```
#include <stdio.h>
```

```
#include <math.h>
```

```
int main() {
```

```
    float principal, rate, time, compoundInterest;
```

```
    printf("Enter the principal amount: ");
```

```
    scanf("%f", &principal);
```

```
    printf("Enter the rate of interest (in percentage): ");
```

```
    scanf("%f", &rate);
```

```
    printf("Enter the time (in years): ");
```

```
    scanf("%f", &time);
```

```
    compoundInterest = principal * pow((1 + rate / 100), time) - principal;
```

```
    printf("Compound Interest: %.2f\n", compoundInterest);
```

```
    return 0;
```

```
}
```

```
Enter the principal amount: 17
Enter the rate of interest (in percentage): 23
Enter the time (in years): 2
Compound Interest: 8.72

-----
Process exited after 11.23 seconds with return value 0
Press any key to continue . . . |
```

Q5. Write a program to Calculate Cube of a Number.

Ans.

```
#include <stdio.h>
```

```
int main() {
```

```
    int number, cube;
```

```
    printf("Enter a number: ");
```

```
    scanf("%d", &number);
```

```
    cube = number * number * number;
```

```
    printf("The cube of %d is: %d\n", number, cube);
```

```
    return 0;
```

```
}
```

```
Enter a number: 9
The cube of 9 is: 729

-----
Process exited after 3.548 seconds with return value 0
Press any key to continue . . .
```

## **WORKSHEET WEEK 2**

Q.1 Write a program to interchange two values by using Assignment Operator.

Ans.

```
#include <stdio.h>
```

```
int main() {
```

```
    int a, b;
```

```
    printf("Enter the value of a: ");
```

```
    scanf("%d", &a);
```

```
    printf("Enter the value of b: ");
```

```
    scanf("%d", &b);
```

```
    a = a + b;
```

```
    b = a - b;
```

```
    a = a - b;
```

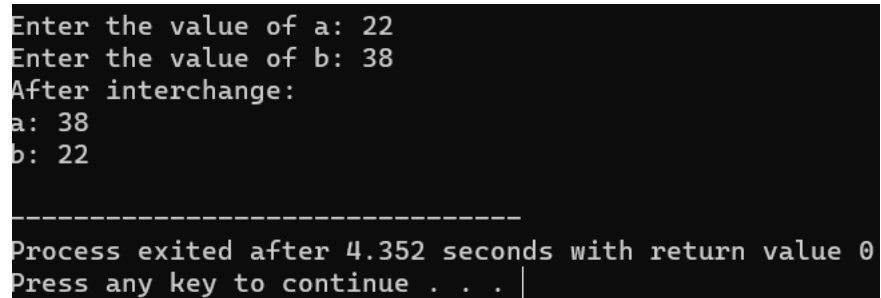
```
    printf("After interchange:\n");
```

```
    printf("a: %d\n", a);
```

```
    printf("b: %d\n", b);
```

```
    return 0;
```

```
}
```



```
Enter the value of a: 22
Enter the value of b: 38
After interchange:
a: 38
b: 22

-----
Process exited after 4.352 seconds with return value 0
Press any key to continue . . . |
```

Q. Write a program to interchange two values by using Arithmetic Operator.

Ans.

```
#include <stdio.h>
```

```
int main() {
```

```
    int a, b;
```

```
    printf("Enter the value of a: ");
```

```
    scanf("%d", &a);
```

```
    printf("Enter the value of b: ");
```

```
    scanf("%d", &b);
```

```
    a = a + b;
```

```
    b = a - b;
```

```
a = a - b;

printf("After interchange:\n");

printf("a: %d\n", a);

printf("b: %d\n", b);


return 0;
}
```

```
Enter the value of a: 11
Enter the value of b: 90
After interchange:
a: 90
b: 11

-----
Process exited after 6.466 seconds with return value 0
Press any key to continue . . . |
```

Q. Write a program to interchange two values by using Bitwise Operator.

Ans.

```
#include <stdio.h>
```

```
int main() {

    int a, b;


    printf("Enter the value of a: ");

    scanf("%d", &a);


    printf("Enter the value of b: ");

    scanf("%d", &b);


    a = a ^ b;

    b = a ^ b;

    a = a ^ b;

    printf("After interchange:\n");

    printf("a: %d\n", a);

    printf("b: %d\n", b);


    return 0;
}
```

```
Enter the value of a: 66
Enter the value of b: 98
After interchange:
a: 98
b: 66

-----
Process exited after 4.865 seconds with return value 0
Press any key to continue . . . |
```

Q. Write a program to find the size of all data types (Int, Float, Char, Double, Long Double, Short Int etc.).

Ans.

```
#include <stdio.h>
```

```
int main() {

    printf("Size of int: %lu bytes\n", sizeof(int));

    printf("Size of float: %lu bytes\n", sizeof(float));

    printf("Size of char: %lu bytes\n", sizeof(char));

    printf("Size of double: %lu bytes\n", sizeof(double));

    printf("Size of long double: %lu bytes\n", sizeof(long double));

    printf("Size of short int: %lu bytes\n", sizeof(short int));

    printf("Size of long int: %lu bytes\n", sizeof(long int));

    printf("Size of unsigned int: %lu bytes\n", sizeof(unsigned int));

    printf("Size of long long int: %lu bytes\n", sizeof(long long int));


    return 0;

}
```

```
Size of int: 4 bytes
Size of float: 4 bytes
Size of char: 1 bytes
Size of double: 8 bytes
Size of long double: 16 bytes
Size of short int: 2 bytes
Size of long int: 4 bytes
Size of unsigned int: 4 bytes
Size of long long int: 8 bytes

-----
Process exited after 0.06272 seconds with return value 0
Press any key to continue . . . |
```

Q. Write a program to find out whether input number is even or odd without using arithmetic operators.

Ans.



```

#include <stdio.h>

int isEven(int num) {
    // Using bitwise AND operator to check the least significant bit
    return (num & 1) == 0;
}

int main() {
    int number;

    // Prompt user for input
    printf("Enter a number: ");
    scanf("%d", &number);

    // Check if the number is even or odd using the isEven function
    if (isEven(number)) {
        printf("%d is even.\n", number);
    } else {
        printf("%d is odd.\n", number);
    }
    return 0;
}

```

```

Enter a number: 2
2 is even.

-----
Process exited after 1.901 seconds with return value 0
Press any key to continue . . .

```

### **WORKSHEET WEEK 3**

Q. Write a C program to check whether a given number is even or odd.

Ans.

```

#include <stdio.h>

int main() {
    int number;

    printf("Enter a number: ");
    scanf("%d", &number);

    if (number % 2 == 0) {
        printf("%d is even.\n", number);
    }
}

```

```
} else {  
    printf("%d is odd.\n", number);  
}  
return 0;  
}
```

```
Enter a number: 11  
11 is odd.  
  
-----  
Process exited after 2.421 seconds with return value 0  
Press any key to continue . . . |
```

Q. Write a C program to check whether a given number is positive or negative.

Ans.

```
#include <stdio.h>
```

```
int main() {  
    int number;  
    printf("Enter a number: ");  
    scanf("%d", &number);  
  
    if (number > 0) {  
        printf("%d is positive.\n", number);  
    } else if (number < 0) {  
        printf("%d is negative.\n", number);  
    } else {  
        printf("The number is zero.\n");  
    }  
    return 0;  
}
```

```
Enter a number: 33  
33 is positive.  
  
-----  
Process exited after 1.667 seconds with return value 0  
Press any key to continue . . . |
```

Q. Write a C program to find whether a given year is a leap year or not.

Ans.

```
#include <stdio.h>
```

```
int main() {  
    int year;  
  
    printf("Enter a year: ");  
    scanf("%d", &year);  
  
    if ((year % 4 == 0 && year % 100 != 0) || (year % 400 == 0)) {  
        printf("%d is a leap year.\n", year);  
    } else {  
        printf("%d is not a leap year.\n", year);  
    }  
  
    return 0;  
}
```

```
Enter a year: 2222  
2222 is not a leap year.  
  
-----  
Process exited after 5.122 seconds with return value 0  
Press any key to continue . . .
```

Q. Write a C program to find the largest of three numbers.

Ans.

```
#include <stdio.h>
```

```

int main() {

    int num1, num2, num3;

    printf("Enter three numbers: ");

    scanf("%d %d %d", &num1, &num2, &num3);

    int largest = (num1 > num2) ? ((num1 > num3) ? num1 : num3) : ((num2 > num3) ? num2 : num3);


    printf("The largest number is: %d\n", largest);


    return 0;

}

```

```

Enter three numbers: 6 77 56
The largest number is: 77

-----
Process exited after 6.464 seconds with return value 0
Press any key to continue . . . |

```

Q. Write a C program to read temperature in centigrade and display a suitable message.

Ans.

```

#include <stdio.h>

int main() {

    float temperature;


    printf("Enter the temperature in centigrade: ");

    scanf("%f", &temperature);

    if (temperature < 0) {

        printf("Freezing weather.\n");

    } else if (temperature >= 0 && temperature <= 10) {

        printf("Very Cold weather.\n");

    } else if (temperature > 10 && temperature <= 20) {

```

```

        printf("Cold weather.\n");
    } else if (temperature > 20 && temperature <= 30) {
        printf("Normal in Temp.\n");
    } else if (temperature > 30 && temperature <= 40) {
        printf("It's Hot.\n");
    } else {
        printf("It's Very Hot.\n");
    }
    return 0;
}

```

```

Enter the temperature in centigrade: 33
It's Hot.

-----
Process exited after 3.261 seconds with return value 0
Press any key to continue . . . |

```

Q. Write a C program to read any digit and display it in the word.

Ans

```
#include <stdio.h>
```

```

int main() {
    int digit;

    printf("Enter a digit (0-9): ");
    scanf("%d", &digit);

    switch (digit) {
        case 0:
            printf("Zero\n");
            break;
        case 1:
            printf("One\n");
            break;
        case 2:
            printf("Two\n");
            break;
        case 3:
            printf("Three\n");
            break;
    }
}

```

```

    case 4:
        printf("Four\n");
        break;
    case 5:
        printf("Five\n");
        break;
    case 6:
        printf("Six\n");
        break;
    case 7:
        printf("Seven\n");
        break;
    case 8:
        printf("Eight\n");
        break;
    case 9:
        printf("Nine\n");
        break;
    default:
        printf("Invalid digit\n");
}

return 0;
}

```

```

Enter a digit (0-9): 7
Seven

-----
Process exited after 4.365 seconds with return value 0
Press any key to continue . . . |

```

Q. Write a C program to create a Simple Calculator using a switch case.

Ans

```
#include <stdio.h>
```

```

int main() {

    char Operator;

    double num1, num2;


    printf("Enter an Operator (+, -, *, /): ");

    scanf(" %c", &Operator);


    printf("Enter two numbers: ");

    scanf("%lf %lf", &num1, &num2);


    switch (Operator) {

        case '+':

            printf("%.2f + %.2f = %.2f\n", num1, num2, num1 + num2);

            break;

        case '-':

            printf("%.2f - %.2f = %.2f\n", num1, num2, num1 - num2);

            break;

        case '*':

            printf("%.2f * %.2f = %.2f\n", num1, num2, num1 * num2);

            break;

        case '/':

            if (num2 != 0) {

                printf("%.2f / %.2f = %.2f\n", num1, num2, num1 / num2);

            } else {

                printf("Error! Division by zero.\n");

            }

            break;

        default:

            printf("Invalid Operator.\n");

    }


    return 0;

}

```

```

Enter an Operator (+, -, *, /): +
Enter two numbers: 23 77
23.00 + 77.00 = 100.00

-----
Process exited after 14.48 seconds with return value 0
Press any key to continue . . . |

```

Q. Write a C program using C Switch...Case to Calculate the Area of Rectangle/Circle/Triangle.

Ans.

```
#include <stdio.h>
```

```
#define PI 3.14
```

```
int main() {
```

```
    int choice;
```

```
    int area, base, height, radius;
```

```
    printf("Choose a shape to calculate its area:\n");
```

```
    printf("1. Rectangle\n");
```

```
    printf("2. Circle\n");
```

```
    printf("3. Triangle\n");
```

```
    printf("Enter your choice (1, 2, or 3): ");
```

```
    scanf("%d", &choice);
```

```
    switch (choice) {
```

```
        case 1:
```

```
            printf("Enter the length of the rectangle: ");
```

```
            scanf("%d", &base);
```



```

printf("Enter the width of the rectangle: ");

scanf("%d", &height);

area = base * height;

printf("Area of the rectangle: %d\n", area);

break;

```

case 2:

```

printf("Enter the radius of the circle: ");

scanf("%d", &radius);

area = PI * radius * radius;

printf("Area of the circle: %d\n", area);

break;

```

case 3

```

printf("Enter the base of the triangle: ");

scanf("%d", &base);

printf("Enter the height of the triangle: ");

scanf("%d", &height);

area = base * height / 2;

printf("Area of the triangle: %d\n", area);

break;

```

default:

```

printf("Invalid choice. Please enter 1, 2, or 3.\n");

```

```

}

```

```

return 0;

```

```

}

```

```

Choose a shape to calculate its area:
1. Rectangle
2. Circle
3. Triangle
Enter your choice (1, 2, or 3): 2
Enter the radius of the circle: 33
Area of the circle: 3419

-----
Process exited after 3.305 seconds with return value 0
Press any key to continue . . .

```

Q. Write a C program to calculate the sum and average of positive numbers.

Ans.

```

#include <stdio.h>

```

```

int main() {

    int num, count = 0, sum = 0;

    printf("Enter positive numbers (enter a negative number to finish):\n");

    while (1) {

        printf("Enter a number: ");

        scanf("%d", &num);

        if (num < 0) {

            break;

        }

        sum += num;

        count++;

    }

    if (count > 0) {

        int average = sum / count;

        printf("Sum: %d\n", sum);

        printf("Average: %d\n", average);

    } else {

        printf("No positive numbers were entered.\n");

    }

    return 0;

}

```

```

Enter positive numbers (enter a negative number to finish):
Enter a number: 4
Enter a number: 6
Enter a number: 9
Enter a number: 0
Enter a number: -8
Sum: 19
Average: 4

-----
Process exited after 13.16 seconds with return value 0
Press any key to continue . . .

```

Q.

Ans.

```
#include <stdio.h>
```

```
#include <stdlib.h>
```

```
#include <unistd.h>
```

```
#include <time.h>
```

```

int main() {

    int hours, minutes, seconds;

    while (1) {

        system("clear");

```

```
time_t now;  
  
time(&now);  
  
hours = (now / 3600) % 24;  
  
minutes = (now / 60) % 60;  
  
seconds = now % 60;  
  
printf("Digital Clock: %02d:%02d:%02d\n", hours, minutes, seconds);  
  
sleep(1);  
  
}  
  
return 0;  
  
}
```

```
'clear' is not recognized as an internal or external command,  
operable program or batch file.  
Digital Clock: 19:53:24  
'clear' is not recognized as an internal or external command,  
operable program or batch file.  
Digital Clock: 19:53:25  
'clear' is not recognized as an internal or external command,  
operable program or batch file.  
Digital Clock: 19:53:26
```

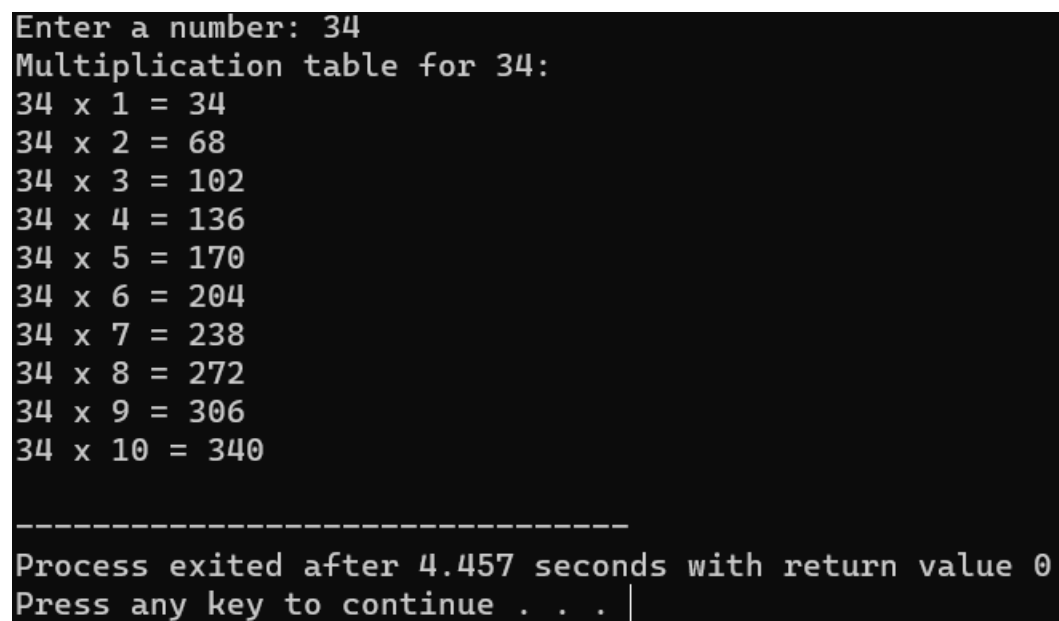
#### WORKSHEET WEEK 4

Q. Write a C program to print multiplication table of a number.

Ans.

```
#include <stdio.h>
```

```
int main() {  
    int num, i;  
  
    printf("Enter a number: ");  
    scanf("%d", &num);  
  
    printf("Multiplication table for %d:\n", num);  
    for (i = 1; i <= 10; ++i) {  
        printf("%d x %d = %d\n", num, i, num * i);  
    }  
  
    return 0;  
}
```



```
Enter a number: 34  
Multiplication table for 34:  
34 x 1 = 34  
34 x 2 = 68  
34 x 3 = 102  
34 x 4 = 136  
34 x 5 = 170  
34 x 6 = 204  
34 x 7 = 238  
34 x 8 = 272  
34 x 9 = 306  
34 x 10 = 340  
  
-----  
Process exited after 4.457 seconds with return value 0  
Press any key to continue . . . |
```

Q. Write a C program to calculate factorial of a number.

Ans.

```
#include <stdio.h>
```

```
int factorial(int n) {  
    if (n == 0 || n == 1) {  
        return 1;  
    } else {  
        return n * factorial(n - 1);  
    }  
}
```

```

}

int main() {

    int num;

    printf("Enter a number: ");

    scanf("%d", &num);

    printf("Factorial of %d is: %d\n", num, factorial(num));

    return 0;

}

```

```

Enter a number: 22
Factorial of 22 is: -522715136

-----
Process exited after 1.959 seconds with return value 0
Press any key to continue . . .

```

Q. Write a C program to check whether a number is palindrome or not.

Ans.

```

#include <stdio.h>

int isPalindrome(int num) {

    int reversed = 0, original = num;

    while (num != 0) {

        reversed = reversed * 10 + num % 10;

        num /= 10;

    }

    return reversed == original;

}

int main() {

    int num;

    printf("Enter a number: ");

    scanf("%d", &num);

    if (isPalindrome(num)) {

        printf("%d is a palindrome.\n", num);

    } else {

        printf("%d is not a palindrome.\n", num);

    }

}

```

```
}  
  
return 0;  
  
}
```

```
Enter a number: 19  
19 is not a palindrome.  
  
-----  
Process exited after 2.705 seconds with return value 0  
Press any key to continue . . . |
```

Q. Write a C program to count frequency of digits in a given number.

Ans

```
#include <stdio.h>
```

```
int isPalindrome(int num) {  
    int reversed = 0, original = num;  
  
    while (num != 0) {  
        reversed = reversed * 10 + num % 10;  
        num /= 10;  
    }  
    return reversed == original;  
}  
  
int main() {  
    int num;  
    printf("Enter a number: ");  
    scanf("%d", &num);  
    if (isPalindrome(num)) {  
        printf("%d is a palindrome.\n", num);  
    } else {  
        printf("%d is not a palindrome.\n", num);  
    }  
    return 0;  
}
```

```
Enter a number: 222  
222 is a palindrome.  
  
-----  
Process exited after 2.581 seconds with return value 0  
Press any key to continue . . .
```

Q. Write a C program to find HCF(GCD) AND LCM of two numbers.

Ans.

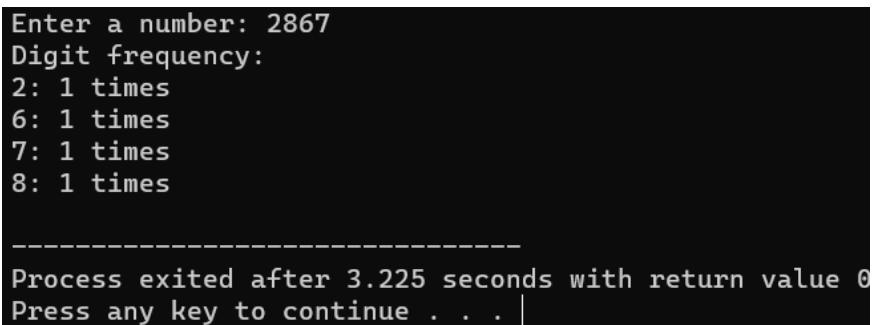
```
#include <stdio.h>

void countDigits(int num) {
    int count[10] = {0};
    while (num != 0) {
        count[num % 10]++;
        num /= 10;
    }
    printf("Digit frequency:\n");
    for (int i = 0; i < 10; ++i) {
        if (count[i] > 0) {
            printf("%d: %d times\n", i, count[i]);
        }
    }
}

int main() {
    int num;

    printf("Enter a number: ");
    scanf("%d", &num);
    countDigits(num);

    return 0;
}
```

A screenshot of a terminal window showing the execution of the C program. The user enters the number 2867. The program outputs the digit frequency: 2 appears 1 time, 6 appears 1 time, 7 appears 1 time, and 8 appears 1 time. Below this, a separator line is shown, followed by the message "Process exited after 3.225 seconds with return value 0" and "Press any key to continue . . . |".

```
Enter a number: 2867
Digit frequency:
2: 1 times
6: 1 times
7: 1 times
8: 1 times

-----
Process exited after 3.225 seconds with return value 0
Press any key to continue . . . |
```

Q.

Ans.

```
#include <stdio.h>

int findGCD(int a, int b) {
    while (a != b) {
        if (a > b) {
            a -= b;
        } else {
            b -= a;
        }
    }
}
```

```

    }
}
return a;
}

int findLCM(int a, int b) {
    return (a * b) / findGCD(a, b);
}

int main() {
    int num1, num2;

    printf("Enter two numbers: ");
    scanf("%d %d", &num1, &num2);

    printf("GCD of %d and %d is: %d\n", num1, num2, findGCD(num1, num2));
    printf("LCM of %d and %d is: %d\n", num1, num2, findLCM(num1, num2));

    return 0;
}

```

```

Enter two numbers: 3647 8
GCD of 3647 and 8 is: 1
LCM of 3647 and 8 is: 29176
-----
Process exited after 6.382 seconds with return value 0
Press any key to continue . . . |

```

Q. Write a C program to print all prime numbers between 1 to n.

Ans.

```

#include <stdio.h>

int isPrime(int num) {
    if (num < 2) {
        return 0;
    }

    for (int i = 2; i * i <= num; ++i) {
        if (num % i == 0) {
            return 0;
        }
    }

    return 1;
}

int main() {
    int n;

    printf("Enter a number (n): ");
    scanf("%d", &n);

```



```
printf("Prime numbers between 1 and %d are:\n", n);  
for (int i = 2; i <= n; ++i) {  
    if (isPrime(i)) {  
        printf("%d\n", i);  
    }  
}  
return 0;  
}
```

```
Enter a number (n): 64  
Prime numbers between 1 and 64 are:  
2  
3  
5  
7  
11  
13  
17  
19  
23  
29  
31  
37  
41  
43  
47  
53  
59  
61  
  
-----  
Process exited after 3.205 seconds with return value 0  
Press any key to continue . . .
```

Q. Write a C program to print Fibonacci series up to n terms.

Ans.

```

#include <stdio.h>

void fibonacci(int n) {

    int a = 0, b = 1, next;

    printf("Fibonacci series up to %d terms:\n", n);

    for (int i = 0; i < n; ++i) {

        printf("%d, ", a);

        next = a + b;

        a = b;

        b = next;

    }

}

int main() {

    int terms;

    printf("Enter the number of terms: ");

    scanf("%d", &terms);

    fibonacci(terms);

    return 0;

}

```

```

Enter the number of terms: 45
Fibonacci series up to 45 terms:
0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, 610, 987, 1597, 2584, 4181, 6765, 10946, 17711, 28657, 46368, 75
025, 121393, 196418, 317811, 514229, 832040, 1346269, 2178309, 3524578, 5702887, 9227465, 14930352, 24157817, 39088169,
63245986, 102334155, 165580141, 267914296, 433494437, 701408733,
-----
Process exited after 1.452 seconds with return value 0
Press any key to continue . . . |

```

Q. Write a C program to print Armstrong numbers from 1 to n AND check a given number is Armstrong numbers or not.

Ans.

```

#include <stdio.h>

#include <math.h>

int isArmstrong(int num) {

    int original = num, result = 0, n = 0;

    while (original != 0) {

        original /= 10;

        ++n; }

    original = num;

    while (original != 0) {

```

```
        int remainder = original % 10;

        result += pow(remainder, n);

        original /= 10;
    }

    return result == num;
}

int main() {

    int num;

    printf("Enter a number: ");

    scanf("%d", &num);

    if (isArmstrong(num)) {

        printf("%d is an Armstrong number.\n", num);

    } else {

        printf("%d is not an Armstrong number.\n", num);

    }

    return 0;
}
```

```
Enter a number: 2836
2836 is not an Armstrong number.

-----
Process exited after 2.417 seconds with return value 0
Press any key to continue . . .
```

Q.

Ans.

```
#include <stdio.h>
```

```
int isPerfect(int num) {
```

```
    int sum = 0;
```

```
    for (int i = 1; i <= num / 2; ++i) {
```

```
        if (num % i == 0) {
```

```
            sum += i;
```

```
        }
```

```
    }
```

```
    return sum == num;
```

```
}
```

```
void printPerfectNumbers(int n) {
```

```
    printf("Perfect numbers between 1 and %d are:\n", n);
```

```
    for (int i = 1; i <= n; ++i) {
```

```
        if (isPerfect(i)) {
```

```
            printf("%d\n", i);
```

```
        }
```

```
    }
```

```
}
```

```
int main() {
```

```
    int num;
```

```
    printf("Enter a number: ");
```

```
    scanf("%d", &num);
```

```
    if (isPerfect(num)) {
```

```
        printf("%d is a Perfect number.\n", num);
```

```
    } else {
```

```
        printf("%d is not a Perfect number.\n", num);
```

```
    }
```

```
    return 0;  
}
```

```
Enter a number: 3  
3 is not a Perfect number.  
  
-----  
Process exited after 2.378 seconds with return value 0  
Press any key to continue . . . |
```

Q. Write a C program to print all Strong Numbers between 1 to n.

Ans

```
#include <stdio.h>
```

```
int factorial(int n) {  
    if (n == 0 || n == 1) {  
        return 1;  
    } else {  
        return n * factorial(n - 1);  
    }  
}
```

```

int isStrong(int num) {
    int original = num, sum = 0;
    while (num != 0) {
        int digit = num % 10;
        sum += factorial(digit);
        num /= 10;
    }

    return sum == original;
}

void printStrongNumbers(int n) {
    printf("Strong numbers between 1 and %d are:\n", n);
    for (int i = 1; i <= n; ++i) {
        if (isStrong(i)) {
            printf("%d\n", i);
        }
    }
}

int main() {
    int num;
    printf("Enter a number: ");
    scanf("%d", &num);

    printStrongNumbers(num);
    return 0;
}

```

```

Enter a number: 3744
Strong numbers between 1 and 3744 are:
1
2
145

-----
Process exited after 2.094 seconds with return value 0
Press any key to continue . . . |

```

## WORKSHEET WEEK 5

Pattern A

```
#include <stdio.h>
```

```
int main() {  
    printf("a.\n");  
    for (int i = 5; i >= 1; i--) {  
        for (int j = 1; j <= i; j++) {  
            printf("%d", j);  
        }  
        printf("\n");  
    }  
  
    return 0;  
}
```

```
a.  
12345  
1234  
123  
12  
1  
-----  
Process exited after 0.01285 seconds with return value 0  
Press any key to continue . . . |
```

Pattern B

```
#include <stdio.h>
```

```
int main() {  
    printf("b.\n");  
    for (int i = 5; i >= 1; i--) {  
        for (int j = 1; j <= 5; j++) {  
            printf("%d", j);  
        }  
        printf("\n");  
    }  
  
    return 0;  
}
```

```
b.  
12345  
12345  
12345  
12345  
12345  
  
-----  
Process exited after 0.07638 seconds with return value 0  
Press any key to continue . . . |
```

Pattern C

```
#include <stdio.h>
```

```
int main() {  
    printf("c.\n");  
    for (int i = 1; i <= 4; i++) {  
        for (int j = 1; j <= i; j++) {  
            printf("%d", i);  
        }  
        printf("\n");  
    }  
}
```



```
}

return 0;

}
```

```
c.
1
22
333
4444
-----
Process exited after 0.08872 seconds with return value 0
Press any key to continue . . .
```

Pattern D

```
#include <stdio.h>
```

```
int main() {
    printf("d.\n");
    for (int i = 1; i <= 4; i++) {
        for (int j = 1; j <= i; j++) {
            printf("%d", j);
        }
        printf("\n");
    }

    return 0;
}
```

```
d.
1
12
123
1234

-----
Process exited after 0.01394 seconds with return value 0
Press any key to continue . . . |
```

Pattern E

```
#include <stdio.h>
```

```
int main() {
    printf("e.\n");
    for (int i = 1; i <= 5; i++) {
        for (int j = 1; j <= i; j++) {
            printf("*");
        }
        printf("\n");
    }
}
```

```
    return 0;
```

```
}
```

```
e.
*
**
***
****
*****

-----
Process exited after 0.06764 seconds with return value 0
Press any key to continue . . . |
```

## WEEK 6:

Q1. Write a menu driven program to insert and delete elements of kth position to an array of size N.

```
# include <stdio.h>

int main ()
{
    int a;
    printf("enetr the no of the elements of the array:- ");
    scanf("%d",&a);
    int n[a];
    for(int i=0;i<a;i++)
    {
        printf("enetr the %d element of the array:- ",i+1);
        scanf("%d",&n[i]);
    }
    int k;
    printf("enter the element which u want to delete:- ");
    scanf("%d",&k);
    int g;
    printf("enetr the element insert behalf of delete element:- ");
    scanf("%d",&g);
    for(int i=0;i<a;i++)
    {
        if(n[i]==k)
        {
            n[i]=g;
        }
        else
        {
            printf("not found!");
            break;
        }
    }
    for(int i=0;i<a;i++)
    {
        printf("\n%d ",n[i]);
    }
    return 0;
}
```

Q2. Write the program to print the biggest and smallest element in an array.

```
#include <stdio.h>

int main() {
    int size;

    // Get the size of the array
    printf("Enter the size of the array: ");
    scanf("%d", &size);
```

```

// Check for invalid input
if (size <= 0) {
    printf("Invalid array size. Exiting...\n");
    return 1;
}

// Create an array of integers
int arr[size];

// Get elements from the user
printf("Enter %d elements:\n", size);
for (int i = 0; i < size; i++) {
    printf("Element %d: ", i + 1);
    scanf("%d", &arr[i]);
}

// Initialize variables for the largest and smallest elements
int largest = arr[0];
int smallest = arr[0];

// Find the largest and smallest elements
for (int i = 1; i < size; i++) {
    if (arr[i] > largest) {
        largest = arr[i];
    }
    if (arr[i] < smallest) {
        smallest = arr[i];
    }
}

// Print the results
printf("The largest element is: %d\n", largest);
printf("The smallest element is: %d\n", smallest);

return 0;
}

```

Q3. Write the program to print the sum and average of an array.

```

#include <stdio.h>

int main() {
    int size;

    // Get the size of the array
    printf("Enter the size of the array: ");
    scanf("%d", &size);

    // Check for invalid input
    if (size <= 0) {
        printf("Invalid array size. Exiting...\n");
        return 1;
    }

    // Create an array of integers
    int arr[size];

    // Get elements from the user
    printf("Enter %d elements:\n", size);
    for (int i = 0; i < size; i++) {
        printf("Element %d: ", i + 1);
        scanf("%d", &arr[i]);
    }
}

```

```

    }

    // Calculate the sum
    int sum = 0;
    for (int i = 0; i < size; i++) {
        sum += arr[i];
    }

    // Calculate the average
    float average = (float)sum / size;

    // Print the results
    printf("The sum of the elements is: %d\n", sum);
    printf("The average of the elements is: %.2f\n", average);

    return 0;
}

```

Q4. Write the program to sort an array using bubble sort.

```

#include <stdio.h>

void swap(int *a, int *b) {
    int temp = *a;
    *a = *b;
    *b = temp;
}

void bubbleSort(int arr[], int size) {
    for (int i = 0; i < size - 1; i++) {
        for (int j = 0; j < size - i - 1; j++) {
            if (arr[j] > arr[j + 1]) {
                // Swap elements if they are in the wrong order
                swap(&arr[j], &arr[j + 1]);
            }
        }
    }
}

int main() {
    int size;

    // Get the size of the array
    printf("Enter the size of the array: ");
    scanf("%d", &size);

    // Check for invalid input
    if (size <= 0) {
        printf("Invalid array size. Exiting...\n");
        return 1;
    }

    // Create an array of integers
    int arr[size];

    // Get elements from the user
    printf("Enter %d elements:\n", size);
    for (int i = 0; i < size; i++) {
        printf("Element %d: ", i + 1);
        scanf("%d", &arr[i]);
    }

    // Sort the array using bubble sort
    bubbleSort(arr, size);
}

```

```

        // Print the sorted array
        printf("\nSorted array using Bubble Sort:\n");
        for (int i = 0; i < size; i++) {
            printf("%d ", arr[i]);
        }
        printf("\n");

        return 0;
}

```

Q5. Write the program to search an element using linear search as well as binary search.

```

#include <stdio.h>

// Function for linear search
int linearSearch(int arr[], int size, int key) {
    for (int i = 0; i < size; i++) {
        if (arr[i] == key) {
            return i; // Return the index if the element is found
        }
    }
    return -1; // Return -1 if the element is not found
}

// Function for binary search (assuming the array is sorted)
int binarySearch(int arr[], int size, int key) {
    int low = 0, high = size - 1;

    while (low <= high) {
        int mid = low + (high - low) / 2;

        if (arr[mid] == key) {
            return mid; // Return the index if the element is found
        } else if (arr[mid] < key) {
            low = mid + 1;
        } else {
            high = mid - 1;
        }
    }

    return -1; // Return -1 if the element is not found
}

int main() {
    int size, key;

    // Get the size of the array
    printf("Enter the size of the array: ");
    scanf("%d", &size);

    // Check for invalid input
    if (size <= 0) {
        printf("Invalid array size. Exiting...\n");
        return 1;
    }

    // Create a sorted array of integers
    int arr[size];

    printf("Enter %d sorted elements:\n", size);
    for (int i = 0; i < size; i++) {
        printf("Element %d: ", i + 1);
        scanf("%d", &arr[i]);
    }
}

```

```

}

// Get the element to search
printf("Enter the element to search: ");
scanf("%d", &key);

// Perform linear search
int linearIndex = linearSearch(arr, size, key);
if (linearIndex != -1) {
    printf("Linear Search: Element found at index %d\n", linearIndex);
} else {
    printf("Linear Search: Element not found\n");
}

// Perform binary search
int binaryIndex = binarySearch(arr, size, key);
if (binaryIndex != -1) {
    printf("Binary Search: Element found at index %d\n", binaryIndex);
} else {
    printf("Binary Search: Element not found\n");
}

return 0;
}

```

Q6. Take an array of 20 integer inputs from user and print the following: a. number of positive numbers  
b. number of negative numbers c. number of odd numbers d. number of even numbers e. number of 0.

```

#include <stdio.h>

int main() {
    const int size = 20;
    int numbers[size];

    // Get 20 integer inputs from the user
    printf("Enter %d integer numbers:\n", size);
    for (int i = 0; i < size; i++) {
        printf("Element %d: ", i + 1);
        scanf("%d", &numbers[i]);
    }

    // Initialize counters for various statistics
    int positiveCount = 0, negativeCount = 0, oddCount = 0, evenCount = 0,
    zeroCount = 0;

    // Calculate statistics
    for (int i = 0; i < size; i++) {
        if (numbers[i] > 0) {
            positiveCount++;
        } else if (numbers[i] < 0) {
            negativeCount++;
        }

        if (numbers[i] % 2 == 0) {
            evenCount++;
        } else {
            oddCount++;
        }

        if (numbers[i] == 0) {
            zeroCount++;
        }
    }
}

```

```

// Print the statistics
printf("\nStatistics:\n");
printf("a. Number of positive numbers: %d\n", positiveCount);
printf("b. Number of negative numbers: %d\n", negativeCount);
printf("c. Number of odd numbers: %d\n", oddCount);
printf("d. Number of even numbers: %d\n", evenCount);
printf("e. Number of zeros: %d\n", zeroCount);

return 0;
}

```

Q7. Take an array of 10 elements. Split it into middle and store the elements in two different arrays.

```

#include <stdio.h>

int main() {
    const int size = 10;
    int originalArray[size];
    int firstArray[size / 2], secondArray[size / 2];

    // Get 10 integer inputs from the user
    printf("Enter %d integer numbers:\n", size);
    for (int i = 0; i < size; i++) {
        printf("Element %d: ", i + 1);
        scanf("%d", &originalArray[i]);
    }

    // Split the array into two parts
    for (int i = 0; i < size / 2; i++) {
        firstArray[i] = originalArray[i];
        secondArray[i] = originalArray[size / 2 + i];
    }

    // Print the original array
    printf("\nOriginal Array:\n");
    for (int i = 0; i < size; i++) {
        printf("%d ", originalArray[i]);
    }

    // Print the two split arrays
    printf("\n\nSplit Arrays:\n");
    printf("First Array:\n");
    for (int i = 0; i < size / 2; i++) {
        printf("%d ", firstArray[i]);
    }

    printf("\nSecond Array:\n");
    for (int i = 0; i < size / 2; i++) {
        printf("%d ", secondArray[i]);
    }

    return 0;
}

```

Q8. Write the program to count frequency of each element in an array.

```

#include <stdio.h>

int main() {
    const int size = 10;
    int arr[size];

    // Get 10 integer inputs from the user

```



```

printf("Enter %d integer numbers:\n", size);
for (int i = 0; i < size; i++) {
    printf("Element %d: ", i + 1);
    scanf("%d", &arr[i]);
}

// Initialize an array to store the frequency of each element
int frequency[size];

for (int i = 0; i < size; i++) {
    frequency[i] = -1; // Mark elements as not counted yet
}

// Count the frequency of each element
for (int i = 0; i < size; i++) {
    int count = 1;

    for (int j = i + 1; j < size; j++) {
        if (arr[i] == arr[j]) {
            count++;
            frequency[j] = 0; // Mark the element as counted
        }
    }

    if (frequency[i] != 0) {
        frequency[i] = count;
    }
}

// Print the frequency of each element
printf("\nFrequency of each element:\n");
for (int i = 0; i < size; i++) {
    if (frequency[i] != 0) {
        printf("%d occurs %d times\n", arr[i], frequency[i]);
    }
}

return 0;
}

```

## WEEK 7:

Q1. Write the program to print row major and column major matrix.

```

#include <stdio.h>

int main() {
    int rows, cols;

    // Get the number of rows and columns for the matrix
    printf("Enter the number of rows: ");
    scanf("%d", &rows);

    printf("Enter the number of columns: ");
    scanf("%d", &cols);

    // Check for invalid input
    if (rows <= 0 || cols <= 0) {
        printf("Invalid matrix size. Exiting...\n");
    }
}

```

```

        return 1;
    }

    int matrix[rows][cols];

    // Get matrix elements from the user
    printf("Enter the matrix elements:\n");
    for (int i = 0; i < rows; i++) {
        for (int j = 0; j < cols; j++) {
            printf("Element at (%d, %d): ", i + 1, j + 1);
            scanf("%d", &matrix[i][j]);
        }
    }

    // Print the matrix in row-major order
    printf("\nRow-Major Order:\n");
    for (int i = 0; i < rows; i++) {
        for (int j = 0; j < cols; j++) {
            printf("%d ", matrix[i][j]);
        }
        printf("\n");
    }

    // Print the matrix in column-major order
    printf("\nColumn-Major Order:\n");
    for (int j = 0; j < cols; j++) {
        for (int i = 0; i < rows; i++) {
            printf("%d ", matrix[i][j]);
        }
        printf("\n");
    }

    return 0;
}

```

Q2. Write the program to print sum of a whole matrix.

```

#include <stdio.h>

int main() {
    int rows, cols;

    // Get the number of rows and columns for the matrix
    printf("Enter the number of rows: ");
    scanf("%d", &rows);

    printf("Enter the number of columns: ");
    scanf("%d", &cols);

    // Check for invalid input
    if (rows <= 0 || cols <= 0) {
        printf("Invalid matrix size. Exiting...\n");
        return 1;
    }

    int matrix[rows][cols];

    // Get matrix elements from the user
    printf("Enter the matrix elements:\n");
    for (int i = 0; i < rows; i++) {
        for (int j = 0; j < cols; j++) {
            printf("Element at (%d, %d): ", i + 1, j + 1);
            scanf("%d", &matrix[i][j]);
        }
    }
}

```

```

    }

    // Calculate the sum of all elements in the matrix
    int sum = 0;
    for (int i = 0; i < rows; i++) {
        for (int j = 0; j < cols; j++) {
            sum += matrix[i][j];
        }
    }

    // Print the sum of the matrix
    printf("\nSum of the matrix: %d\n", sum);

    return 0;
}

```

Q3. Write a program to add and multiply two 3x3 matrices. You can use 2D array to create a matrix.

```

#include <stdio.h>

void addMatrices(int mat1[3][3], int mat2[3][3], int result[3][3]) {
    for (int i = 0; i < 3; i++) {
        for (int j = 0; j < 3; j++) {
            result[i][j] = mat1[i][j] + mat2[i][j];
        }
    }
}

void multiplyMatrices(int mat1[3][3], int mat2[3][3], int result[3][3]) {
    for (int i = 0; i < 3; i++) {
        for (int j = 0; j < 3; j++) {
            result[i][j] = 0;
            for (int k = 0; k < 3; k++) {
                result[i][j] += mat1[i][k] * mat2[k][j];
            }
        }
    }
}

void printMatrix(int mat[3][3]) {
    for (int i = 0; i < 3; i++) {
        for (int j = 0; j < 3; j++) {
            printf("%d ", mat[i][j]);
        }
        printf("\n");
    }
}

int main() {
    int matrix1[3][3], matrix2[3][3], resultAddition[3][3], resultMultiplication[3][3];

    // Get elements for the first matrix
    printf("Enter elements for the first matrix (3x3):\n");
    for (int i = 0; i < 3; i++) {
        for (int j = 0; j < 3; j++) {
            printf("Element at (%d, %d): ", i + 1, j + 1);
            scanf("%d", &matrix1[i][j]);
        }
    }

    // Get elements for the second matrix
    printf("\nEnter elements for the second matrix (3x3):\n");
    for (int i = 0; i < 3; i++) {
        for (int j = 0; j < 3; j++) {
            printf("Element at (%d, %d): ", i + 1, j + 1);
            scanf("%d", &matrix2[i][j]);
        }
    }

    // Perform matrix addition
    addMatrices(matrix1, matrix2, resultAddition);

    // Perform matrix multiplication
    multiplyMatrices(matrix1, matrix2, resultMultiplication);
}

```

```

    // Print the results
    printf("\nMatrix Addition:\n");
    printMatrix(resultAddition);

    printf("\nMatrix Multiplication:\n");
    printMatrix(resultMultiplication);

    return 0;
}

```

Q4. Write the program to print sum of all diagonal elements, upper triangular matrix and lower triangular matrix.

```

#include <stdio.h>

int main() {
    int size;

    // Get the size of the square matrix
    printf("Enter the size of the square matrix: ");
    scanf("%d", &size);

    // Check for invalid input
    if (size <= 0) {
        printf("Invalid matrix size. Exiting...\n");
        return 1;
    }

    int matrix[size][size];

    // Get matrix elements from the user
    printf("Enter the matrix elements (%dx%d):\n", size, size);
    for (int i = 0; i < size; i++) {
        for (int j = 0; j < size; j++) {
            printf("Element at (%d, %d): ", i + 1, j + 1);
            scanf("%d", &matrix[i][j]);
        }
    }

    // Calculate the sum of diagonal elements
    int diagonalSum = 0;
    for (int i = 0; i < size; i++) {
        diagonalSum += matrix[i][i];
    }

    // Calculate the sum of upper triangular matrix
    int upperTriangularSum = 0;
    for (int i = 0; i < size; i++) {
        for (int j = i + 1; j < size; j++) {
            upperTriangularSum += matrix[i][j];
        }
    }

    // Calculate the sum of lower triangular matrix
    int lowerTriangularSum = 0;
    for (int i = 0; i < size; i++) {
        for (int j = 0; j < i; j++) {
            lowerTriangularSum += matrix[i][j];
        }
    }

    // Print the results
    printf("\nSum of Diagonal Elements: %d\n", diagonalSum);
    printf("Sum of Upper Triangular Matrix: %d\n", upperTriangularSum);
    printf("Sum of Lower Triangular Matrix: %d\n", lowerTriangularSum);

    return 0;
}

```

Q5. Write the program to find the frequency of odd and even elements in matrix.

```

#include <stdio.h>

int main() {
    int rows, cols;

```

```

// Get the number of rows and columns for the matrix
printf("Enter the number of rows: ");
scanf("%d", &rows);

printf("Enter the number of columns: ");
scanf("%d", &cols);

// Check for invalid input
if (rows <= 0 || cols <= 0) {
    printf("Invalid matrix size. Exiting...\n");
    return 1;
}

int matrix[rows][cols];

// Get matrix elements from the user
printf("Enter the matrix elements:\n");
for (int i = 0; i < rows; i++) {
    for (int j = 0; j < cols; j++) {
        printf("Element at (%d, %d): ", i + 1, j + 1);
        scanf("%d", &matrix[i][j]);
    }
}

// Initialize counters for even and odd elements
int evenCount = 0, oddCount = 0;

// Calculate the frequency of even and odd elements
for (int i = 0; i < rows; i++) {
    for (int j = 0; j < cols; j++) {
        if (matrix[i][j] % 2 == 0) {
            evenCount++;
        } else {
            oddCount++;
        }
    }
}

// Print the results
printf("\nFrequency of even elements: %d\n", evenCount);
printf("Frequency of odd elements: %d\n", oddCount);

return 0;
}

```

Q6. Write the program to find sum of each row and sum of each column of matrix.

```

#include <stdio.h>

int main() {
    int rows, cols;

    // Get the number of rows and columns for the matrix
    printf("Enter the number of rows: ");
    scanf("%d", &rows);

    printf("Enter the number of columns: ");
    scanf("%d", &cols);

    // Check for invalid input
    if (rows <= 0 || cols <= 0) {
        printf("Invalid matrix size. Exiting...\n");
        return 1;
    }

    int matrix[rows][cols];

    // Get matrix elements from the user
    printf("Enter the matrix elements:\n");
    for (int i = 0; i < rows; i++) {
        for (int j = 0; j < cols; j++) {
            printf("Element at (%d, %d): ", i + 1, j + 1);
            scanf("%d", &matrix[i][j]);
        }
    }

    // Calculate the sum of each row

```

```

printf("\nSum of each row:\n");
for (int i = 0; i < rows; i++) {
    int rowSum = 0;
    for (int j = 0; j < cols; j++) {
        rowSum += matrix[i][j];
    }
    printf("Row %d: %d\n", i + 1, rowSum);
}

// Calculate the sum of each column
printf("\nSum of each column:\n");
for (int j = 0; j < cols; j++) {
    int colSum = 0;
    for (int i = 0; i < rows; i++) {
        colSum += matrix[i][j];
    }
    printf("Column %d: %d\n", j + 1, colSum);
}

return 0;
}

```

Q7. Initialize a 2D array of 3\*3 matrix.

```

#include <stdio.h>

int main() {
    // Initialize a 3x3 matrix
    int matrix[3][3] = {
        {1, 2, 3},
        {4, 5, 6},
        {7, 8, 9}
    };

    // Print the initialized matrix
    printf("Initialized 3x3 Matrix:\n");
    for (int i = 0; i < 3; i++) {
        for (int j = 0; j < 3; j++) {
            printf("%d ", matrix[i][j]);
        }
        printf("\n");
    }

    return 0;
}

```

Q8. A square matrix, one having the same number of rows and columns, is called a diagonal matrix if it's only non-zero elements are on the diagonal from upper left to lower right. It is called upper triangular matrix if all elements below the diagonal are zeroes, and lower triangular matrix, if all the elements above the diagonal are zeroes. Write a program that reads a matrix and determines if it is one of the above mentioned three special matrices.

```

#include <stdio.h>

int isDiagonalMatrix(int matrix[3][3], int size) {
    // Check if all elements outside the diagonal are zero
    for (int i = 0; i < size; i++) {
        for (int j = 0; j < size; j++) {
            if (i != j && matrix[i][j] != 0) {
                return 0; // Not a diagonal matrix
            }
        }
    }
    return 1; // Diagonal matrix
}

int isUpperTriangularMatrix(int matrix[3][3], int size) {
    // Check if all elements below the diagonal are zero
    for (int i = 0; i < size; i++) {
        for (int j = 0; j < i; j++) {
            if (matrix[i][j] != 0) {
                return 0; // Not an upper triangular matrix
            }
        }
    }
    return 1;
}

```

```

    return 1; // Upper triangular matrix
}

int isLowerTriangularMatrix(int matrix[3][3], int size) {
    // Check if all elements above the diagonal are zero
    for (int i = 0; i < size; i++) {
        for (int j = i + 1; j < size; j++) {
            if (matrix[i][j] != 0) {
                return 0; // Not a lower triangular matrix
            }
        }
    }
    return 1; // Lower triangular matrix
}

int main() {
    int size;

    // Get the size of the square matrix
    printf("Enter the size of the square matrix: ");
    scanf("%d", &size);

    // Check for invalid input
    if (size <= 0) {
        printf("Invalid matrix size. Exiting...\n");
        return 1;
    }

    int matrix[3][3];

    // Get matrix elements from the user
    printf("Enter the matrix elements (%dx%d):\n", size, size);
    for (int i = 0; i < size; i++) {
        for (int j = 0; j < size; j++) {
            printf("Element at (%d, %d): ", i + 1, j + 1);
            scanf("%d", &matrix[i][j]);
        }
    }

    // Check and print the type of matrix
    if (isDiagonalMatrix(matrix, size)) {
        printf("The matrix is a diagonal matrix.\n");
    } else if (isUpperTriangularMatrix(matrix, size)) {
        printf("The matrix is an upper triangular matrix.\n");
    } else if (isLowerTriangularMatrix(matrix, size)) {
        printf("The matrix is a lower triangular matrix.\n");
    } else {
        printf("The matrix is not one of the specified special matrices.\n");
    }

    return 0;
}

```

Q9. Write the program to check whether the matrix is sparse matrix or not.

```

#include <stdio.h>

#define MAX_SIZE 10

int isSparseMatrix(int matrix[MAX_SIZE][MAX_SIZE], int rows, int cols) {
    int zeroCount = 0;

    // Count the number of zero elements in the matrix
    for (int i = 0; i < rows; i++) {
        for (int j = 0; j < cols; j++) {
            if (matrix[i][j] == 0) {
                zeroCount++;
            }
        }
    }

    // Define a threshold (you can adjust this value as needed)
    double threshold = 0.6; // 60% of zero elements

    // Calculate the percentage of zero elements
    double zeroPercentage = (double)zeroCount / (rows * cols);
}

```

```

    // Check if the matrix is sparse based on the threshold
    if (zeroPercentage > threshold) {
        return 1; // Sparse matrix
    } else {
        return 0; // Not a sparse matrix
    }
}

int main() {
    int rows, cols;

    // Get the number of rows and columns for the matrix
    printf("Enter the number of rows: ");
    scanf("%d", &rows);

    printf("Enter the number of columns: ");
    scanf("%d", &cols);

    // Check for invalid input
    if (rows <= 0 || cols <= 0 || rows > MAX_SIZE || cols > MAX_SIZE) {
        printf("Invalid matrix size. Exiting...\n");
        return 1;
    }

    int matrix[MAX_SIZE][MAX_SIZE];

    // Get matrix elements from the user
    printf("Enter the matrix elements (%dx%d):\n", rows, cols);
    for (int i = 0; i < rows; i++) {
        for (int j = 0; j < cols; j++) {
            printf("Element at (%d, %d): ", i + 1, j + 1);
            scanf("%d", &matrix[i][j]);
        }
    }

    // Check and print whether the matrix is sparse
    if (isSparseMatrix(matrix, rows, cols)) {
        printf("The matrix is a sparse matrix.\n");
    } else {
        printf("The matrix is not a sparse matrix.\n");
    }

    return 0;
}

```

## WEEK 8:

Q1. Write a C program to create, initialize and use pointers.

```

#include <stdio.h>

int main() {
    // Declare variables
    int number = 42;
    float floatNumber = 3.14;
    char character = 'A';

    // Create pointers
    int *intPointer;
    float *floatPointer;
    char *charPointer;

    // Initialize pointers
    intPointer = &number;
    floatPointer = &floatNumber;
    charPointer = &character;

    // Use pointers to access and modify the values
    printf("Original values:\n");
    printf("Number: %d\n", number);
    printf("Float Number: %.2f\n", floatNumber);
    printf("Character: %c\n\n", character);
}

```



```

// Use pointers to modify the values
*intPointer = 100;
*floatPointer = 2.718;
*charPointer = 'B';

// Print modified values
printf("Modified values using pointers:\n");
printf("Number: %d\n", number);
printf("Float Number: %.2f\n", floatNumber);
printf("Character: %c\n\n", character);

// Use pointers to perform some arithmetic
int anotherNumber = 10;
int *resultPointer = &number;

*resultPointer += anotherNumber;

// Print the result
printf("Result of adding %d to the original number using pointers: %d\n", anotherNumber,
*resultPointer);

return 0;
}

```

Q2. Write a C program to add two numbers using pointers.

```

#include <stdio.h>

int main() {
    // Declare variables
    int num1, num2, sum;
    int *ptr1, *ptr2;

    // Initialize pointers
    ptr1 = &num1;
    ptr2 = &num2;

    // Get two numbers from the user
    printf("Enter the first number: ");
    scanf("%d", ptr1);

    printf("Enter the second number: ");
    scanf("%d", ptr2);

    // Add the numbers using pointers
    sum = *ptr1 + *ptr2;

    // Print the result
    printf("Sum of %d and %d is: %d\n", *ptr1, *ptr2, sum);

    return 0;
}

```

Q3. Write a C program to swap two numbers using pointers.

```

#include <stdio.h>

void swap(int *num1, int *num2) {
    int temp = *num1;
    *num1 = *num2;
    *num2 = temp;
}

int main() {
    // Declare variables
    int num1, num2;

    // Get two numbers from the user
    printf("Enter the first number: ");
    scanf("%d", &num1);

    printf("Enter the second number: ");
    scanf("%d", &num2);
}

```

```

// Print the original values
printf("\nOriginal values:\n");
printf("First number: %d\n", num1);
printf("Second number: %d\n", num2);

// Call the swap function to swap the numbers
swap(&num1, &num2);

// Print the swapped values
printf("\nSwapped values:\n");
printf("First number: %d\n", num1);
printf("Second number: %d\n", num2);

return 0;
}

```

Q. 4 Write a C program to input and print array elements using pointer.

```

#include <stdio.h>

int main() {
    int size;

    // Get the size of the array from the user
    printf("Enter the size of the array: ");
    scanf("%d", &size);

    // Check for invalid input
    if (size <= 0) {
        printf("Invalid array size. Exiting...\n");
        return 1;
    }

    int arr[size];

    // Get array elements from the user using pointers
    printf("Enter %d elements for the array:\n", size);
    for (int i = 0; i < size; i++) {
        printf("Element %d: ", i + 1);
        scanf("%d", &(arr + i)); // Using pointer notation to access array elements
    }

    // Print array elements using pointers
    printf("\nArray elements using pointers:\n");
    for (int i = 0; i < size; i++) {
        printf("%d ", *(arr + i)); // Using pointer notation to access array elements
    }

    return 0;
}

```

Q. 5 Write a C program to copy one array to another using pointer.

```

#include <stdio.h>

void copyArray(int *source, int *destination, int size) {
    for (int i = 0; i < size; i++) {
        *(destination + i) = *(source + i); // Copying elements using pointers
    }
}

int main() {
    int size;

    // Get the size of the array from the user
    printf("Enter the size of the array: ");
    scanf("%d", &size);

    // Check for invalid input
    if (size <= 0) {
        printf("Invalid array size. Exiting...\n");
        return 1;
    }

    int sourceArray[size];
    int destinationArray[size];

```

```

// Get source array elements from the user using pointers
printf("Enter %d elements for the source array:\n", size);
for (int i = 0; i < size; i++) {
    printf("Element %d: ", i + 1);
    scanf("%d", &*(sourceArray + i)); // Using pointer notation to access array elements
}

// Copy array elements using pointers
copyArray(sourceArray, destinationArray, size);

// Print source array elements
printf("\nSource Array elements:\n");
for (int i = 0; i < size; i++) {
    printf("%d ", *(sourceArray + i)); // Using pointer notation to access array elements
}

// Print destination array elements (copied array)
printf("\nDestination Array elements (copied from source):\n");
for (int i = 0; i < size; i++) {
    printf("%d ", *(destinationArray + i)); // Using pointer notation to access array elements
}

return 0;
}

```

Q. 6 Write a C program to swap two arrays using pointers.

```

#include <stdio.h>

void swapArrays(int *arr1, int *arr2, int size) {
    for (int i = 0; i < size; i++) {
        // Swap elements using pointers
        int temp = *(arr1 + i);
        *(arr1 + i) = *(arr2 + i);
        *(arr2 + i) = temp;
    }
}

void printArray(int *arr, int size) {
    for (int i = 0; i < size; i++) {
        printf("%d ", *(arr + i)); // Using pointer notation to access array elements
    }
    printf("\n");
}

int main() {
    int size;

    // Get the size of the arrays from the user
    printf("Enter the size of the arrays: ");
    scanf("%d", &size);

    // Check for invalid input
    if (size <= 0) {
        printf("Invalid array size. Exiting...\n");
        return 1;
    }

    int array1[size], array2[size];

    // Get elements for the first array from the user using pointers
    printf("Enter %d elements for the first array:\n", size);
    for (int i = 0; i < size; i++) {
        printf("Element %d: ", i + 1);
        scanf("%d", &*(array1 + i)); // Using pointer notation to access array elements
    }

    // Get elements for the second array from the user using pointers
    printf("\nEnter %d elements for the second array:\n", size);
    for (int i = 0; i < size; i++) {
        printf("Element %d: ", i + 1);
        scanf("%d", &*(array2 + i)); // Using pointer notation to access array elements
    }

    // Print original arrays
    printf("\nOriginal Arrays:\n");
    printf("Array 1: ");
}

```

```

    printArray(array1, size);
    printf("Array 2: ");
    printArray(array2, size);

    // Swap arrays using pointers
    swapArrays(array1, array2, size);

    // Print swapped arrays
    printf("\nSwapped Arrays:\n");
    printf("Array 1: ");
    printArray(array1, size);
    printf("Array 2: ");
    printArray(array2, size);

    return 0;
}

```

Q. 7 Write a C program to reverse an array using pointers.

```

#include <stdio.h>

void reverseArray(int *arr, int size) {
    int *start = arr;
    int *end = arr + size - 1;

    // Swap elements from the start and end using pointers
    while (start < end) {
        // Swap elements using pointers
        int temp = *start;
        *start = *end;
        *end = temp;

        // Move pointers towards each other
        start++;
        end--;
    }
}

void printArray(int *arr, int size) {
    for (int i = 0; i < size; i++) {
        printf("%d ", *(arr + i)); // Using pointer notation to access array elements
    }
    printf("\n");
}

int main() {
    int size;

    // Get the size of the array from the user
    printf("Enter the size of the array: ");
    scanf("%d", &size);

    // Check for invalid input
    if (size <= 0) {
        printf("Invalid array size. Exiting...\n");
        return 1;
    }

    int array[size];

    // Get array elements from the user using pointers
    printf("Enter %d elements for the array:\n", size);
    for (int i = 0; i < size; i++) {
        printf("Element %d: ", i + 1);
        scanf("%d", &*(array + i)); // Using pointer notation to access array elements
    }

    // Print original array
    printf("\nOriginal Array:\n");
    printArray(array, size);

    // Reverse array using pointers
    reverseArray(array, size);

    // Print reversed array
    printf("\nReversed Array:\n");
    printArray(array, size);
}

```

```
    return 0;
}
```

Q. 8 Write a C program to add two matrix using pointers.

```
#include <stdio.h>

#define MAX_SIZE 10

void addMatrices(int *matrix1, int *matrix2, int *result, int rows, int cols) {
    for (int i = 0; i < rows; i++) {
        for (int j = 0; j < cols; j++) {
            // Add corresponding elements using pointers
            *(result + i * cols + j) = *(matrix1 + i * cols + j) + *(matrix2 + i * cols + j);
        }
    }
}

void printMatrix(int *matrix, int rows, int cols) {
    for (int i = 0; i < rows; i++) {
        for (int j = 0; j < cols; j++) {
            printf("%d ", *(matrix + i * cols + j)); // Using pointer notation to access matrix
elements
        }
        printf("\n");
    }
}

int main() {
    int rows, cols;

    // Get the number of rows and columns for the matrices
    printf("Enter the number of rows: ");
    scanf("%d", &rows);

    printf("Enter the number of columns: ");
    scanf("%d", &cols);

    // Check for invalid input
    if (rows <= 0 || cols <= 0 || rows > MAX_SIZE || cols > MAX_SIZE) {
        printf("Invalid matrix size. Exiting...\n");
        return 1;
    }

    int matrix1[MAX_SIZE][MAX_SIZE], matrix2[MAX_SIZE][MAX_SIZE], result[MAX_SIZE][MAX_SIZE];

    // Get elements for the first matrix from the user using pointers
    printf("Enter elements for the first matrix (%dx%d):\n", rows, cols);
    for (int i = 0; i < rows; i++) {
        for (int j = 0; j < cols; j++) {
            printf("Element at (%d, %d): ", i + 1, j + 1);
            scanf("%d", &*(matrix1 + i * cols + j)); // Using pointer notation to access matrix
elements
        }
    }

    // Get elements for the second matrix from the user using pointers
    printf("\nEnter elements for the second matrix (%dx%d):\n", rows, cols);
    for (int i = 0; i < rows; i++) {
        for (int j = 0; j < cols; j++) {
            printf("Element at (%d, %d): ", i + 1, j + 1);
            scanf("%d", &*(matrix2 + i * cols + j)); // Using pointer notation to access matrix
elements
        }
    }

    // Add matrices using pointers
    addMatrices((int *)matrix1, (int *)matrix2, (int *)result, rows, cols);

    // Print original matrices
    printf("\nOriginal Matrices:\n");
    printf("Matrix 1:\n");
    printMatrix((int *)matrix1, rows, cols);
    printf("Matrix 2:\n");
    printMatrix((int *)matrix2, rows, cols);
}
```

```

    // Print result matrix
    printf("\nResult Matrix (Sum of Matrix 1 and Matrix 2):\n");
    printMatrix((int *)result, rows, cols);

    return 0;
}

```

Q. 9 Write a C program to multiply two matrix using pointers.

```

#include <stdio.h>

#define MAX_SIZE 10

void multiplyMatrices(int *matrix1, int *matrix2, int *result, int rows1, int cols1, int cols2) {
    for (int i = 0; i < rows1; i++) {
        for (int j = 0; j < cols2; j++) {
            *(result + i * cols2 + j) = 0; // Initialize result matrix element to 0

            for (int k = 0; k < cols1; k++) {
                // Multiply and accumulate using pointers
                *(result + i * cols2 + j) += *(matrix1 + i * cols1 + k) * *(matrix2 + k * cols2 +
j);
            }
        }
    }
}

void printMatrix(int *matrix, int rows, int cols) {
    for (int i = 0; i < rows; i++) {
        for (int j = 0; j < cols; j++) {
            printf("%d ", *(matrix + i * cols + j)); // Using pointer notation to access matrix
elements
        }
        printf("\n");
    }
}

int main() {
    int rows1, cols1, rows2, cols2;

    // Get the number of rows and columns for the first matrix
    printf("Enter the number of rows for the first matrix: ");
    scanf("%d", &rows1);

    printf("Enter the number of columns for the first matrix: ");
    scanf("%d", &cols1);

    // Get the number of rows and columns for the second matrix
    printf("\nEnter the number of rows for the second matrix: ");
    scanf("%d", &rows2);

    printf("Enter the number of columns for the second matrix: ");
    scanf("%d", &cols2);

    // Check for valid matrix dimensions for multiplication
    if (cols1 != rows2) {
        printf("Invalid matrix dimensions for multiplication. Exiting...\n");
        return 1;
    }

    // Check for invalid input
    if (rows1 <= 0 || cols1 <= 0 || rows2 <= 0 || cols2 <= 0 ||
rows1 > MAX_SIZE || cols1 > MAX_SIZE || rows2 > MAX_SIZE || cols2 > MAX_SIZE) {
        printf("Invalid matrix size. Exiting...\n");
        return 1;
    }

    int matrix1[MAX_SIZE][MAX_SIZE], matrix2[MAX_SIZE][MAX_SIZE], result[MAX_SIZE][MAX_SIZE];

    // Get elements for the first matrix from the user using pointers
    printf("\nEnter elements for the first matrix (%dx%d):\n", rows1, cols1);
    for (int i = 0; i < rows1; i++) {
        for (int j = 0; j < cols1; j++) {
            printf("Element at (%d, %d): ", i + 1, j + 1);
            scanf("%d", &*(matrix1 + i * cols1 + j)); // Using pointer notation to access matrix
elements

```

```

    }
}

// Get elements for the second matrix from the user using pointers
printf("\nEnter elements for the second matrix (%dx%d):\n", rows2, cols2);
for (int i = 0; i < rows2; i++) {
    for (int j = 0; j < cols2; j++) {
        printf("Element at (%d, %d): ", i + 1, j + 1);
        scanf("%d", &*(matrix2 + i * cols2 + j)); // Using pointer notation to access matrix
elements
    }
}

// Multiply matrices using pointers
multiplyMatrices((int *)matrix1, (int *)matrix2, (int *)result, rows1, cols1, cols2);

// Print original matrices
printf("\nOriginal Matrices:\n");
printf("Matrix 1:\n");
printMatrix((int *)matrix1, rows1, cols1);
printf("Matrix 2:\n");
printMatrix((int *)matrix2, rows2, cols2);

// Print result matrix
printf("\nResult Matrix (Product of Matrix 1 and Matrix 2):\n");
printMatrix((int *)result, rows1, cols2);

return 0;
}

```

## WEEK 9:

Q. 1 Write a C program to Search string.

```

#include <stdio.h>
#include <string.h>

int main() {
    char s1[] = "Beauty is in the eye of the beholder";
    char s2[] = "the";

    int n = 0;
    int m = 0;
    int times = 0;
    int len = strlen(s2); // contains the length of search string

    while(s1[n] != '\0') {

        if(s1[n] == s2[m]) { // if first character of search string matches

            // keep on searching

            while(s1[n] == s2[m] && s1[n] != '\0') {
                n++;
                m++;
            }

            // if we sequence of characters matching with the length of searched string
            if(m == len && (s1[n] == ' ' || s1[n] == '\0')) {

                // BINGO!! we find our search string.
                times++;
            }
        } else { // if first character of search string DOES NOT match
            while(s1[n] != ' ') { // Skip to next word
                n++;
                if(s1[n] == '\0')
                    break;
            }
        }
        n++;
        m=0; // reset the counter to start from first character of the search string.
    }
}

```

```

    if(times > 0) {
        printf("'s' appears %d time(s)\n", s2, times);
    } else {
        printf("'s' does not appear in the sentence.\n", s2);
    }

    return 0;
}

```

Q. 2 Write a C program to Reverse words in string.

```

/**
 * C program to reverse order of words in a string
 */
#include <stdio.h>
#include <string.h>
#define MAX_SIZE 100 // Maximum string size

int main()
{
    char str[100], reverse[100];
    int len, i, index, wordStart, wordEnd;

    printf("Enter any string: ");
    gets(str);

    len = strlen(str);
    index = 0;

    // Start checking of words from the end of string
    wordStart = len - 1;
    wordEnd = len - 1;

    while(wordStart > 0)
    {
        // If a word is found
        if(str[wordStart] == ' ')
        {
            // Add the word to the reverse string
            i = wordStart + 1;
            while(i <= wordEnd)
            {
                reverse[index] = str[i];

                i++;
                index++;
            }
            reverse[index++] = ' ';

            wordEnd = wordStart - 1;
        }

        wordStart--;
    }

    // Finally add the last word
    for(i=0; i<=wordEnd; i++)
    {
        reverse[index] = str[i];
        index++;
    }

    // Add NULL character at the end of reverse string
    reverse[index] = '\0';

    printf("Original string \n%s\n\n", str);
    printf("Reverse ordered words \n%s", reverse);

    return 0;
}

```

Q. 3 Write a C program to count vowels, consonants, etc.

```

#include <stdio.h>
int main() {

    char line[150];

```



```

int vowels, consonant, digit, space;

// initialize all variables to 0
vowels = consonant = digit = space = 0;

// get full line of string input
printf("Enter a line of string: ");
fgets(line, sizeof(line), stdin);

// loop through each character of the string
for (int i = 0; line[i] != '\0'; ++i) {

    // convert character to lowercase
    line[i] = tolower(line[i]);

    // check if the character is a vowel
    if (line[i] == 'a' || line[i] == 'e' || line[i] == 'i' ||
        line[i] == 'o' || line[i] == 'u') {

        // increment value of vowels by 1
        ++vowels;
    }

    // if it is not a vowel and if it is an alphabet, it is a consonant
    else if ((line[i] >= 'a' && line[i] <= 'z')) {
        ++consonant;
    }

    // check if the character is a digit
    else if (line[i] >= '0' && line[i] <= '9') {
        ++digit;
    }

    // check if the character is an empty space
    else if (line[i] == ' ') {
        ++space;
    }
}

printf("Vowels: %d", vowels);
printf("\nConsonants: %d", consonant);
printf("\nDigits: %d", digit);
printf("\nWhite spaces: %d", space);

return 0;
}

```

Q. 4 Create a program to separate characters in a given string?

```

#include <stdio.h>
#include <stdlib.h>

void main()
{
    char str[100]; /* Declares a string of size 100 */
    int l= 0;

    printf("\n\separate the individual characters from a string :\n");
    printf("-----\n");
    printf("Input the string : ");
    fgets(str, sizeof str, stdin);
    printf("The characters of the string are : \n");
    while(str[l]!='\0')
    {
        printf("%c  ", str[l]);
        l++;
    }
    printf("\n");
}

```

Q. 5 Write a program to take two strings from user and concatenate them also add a space between them using strcat() function.

Sample input:

JAI

GLA

Sample output: JAI GLA

```
#include <stdio.h>
#include <string.h>

int main()
{
    char a[100], b[100];

    printf("Enter the first string\n");
    gets(a);

    printf("Enter the second string\n");
    gets(b);

    strcat(a,b);

    printf("String obtained on concatenation is %s\n",a);

    return 0;
}
```

Q. 6 Write a C program to take a string from user and make it toggle its case i.e. lower case to upper case and upper case to lower case.

Sample Input: HEllO wOrLD

Sample output: heLlO WoRLd

```
#include <stdio.h>

void toggleChars(char str[])
{
    for (int i = 0; str[i] != '\0'; i++) {
        if (str[i] >= 'A' && str[i] <= 'Z')
            str[i] = str[i] + 'a' - 'A';
        else if (str[i] >= 'a' && str[i] <= 'z')
            str[i] = str[i] + 'A' - 'a';
    }
}

// Driver code
int main()
{
    char str[] = "GeKf@rGeek$";
    toggleChars(str);
    printf("String after toggle \n");
    printf("%s\n", str);
    return 0;
}
```

Q. 7 Write a C program to take two strings as input from user and check they are identical or not without using string functions.

Sample input:

Jai Gla

Jai Gla

Sample output: Identical

```
#include <stdio.h>
#include <string.h>

int main()
{
    char Str1[100], Str2[100];
    int result, i;

    printf("\n Please Enter the First String : ");
```

```

    gets(Str1);

    printf("\n Please Enter the Second String :  ");
    gets(Str2);

    for(i = 0; Str1[i] == Str2[i] && Str1[i] == '\0'; i++);

    if(Str1[i] < Str2[i])
    {
        printf("\n str1 is Less than str2");
    }
    else if(Str1[i] > Str2[i])
    {
        printf("\n str2 is Less than str1");
    }
    else
    {
        printf("\n str1 is Equal to str2");
    }

    return 0;
}

```

Q. 8 Write a C program to take a list of a student's names from user by asking number of students and sort them alphabetical order.

Sample Input:

Bhisham

Jayant

Abhishek

Dhruv

Sample Output:

Abhishek

Bhisham

Dhruv

Jayant

```

#include <stdio.h>
#include <stdlib.h>
#include <string.h>

int main() {
    int numStudents;

    // Get the number of students
    printf("Enter the number of students: ");
    scanf("%d", &numStudents);

    // Check for invalid input
    if (numStudents <= 0) {
        printf("Invalid number of students. Exiting...\n");
        return 1;
    }

    // Create an array of strings to store student names
    char **studentNames = (char **)malloc(numStudents * sizeof(char *));

    // Get names from the user
    for (int i = 0; i < numStudents; i++) {
        printf("Enter the name of student %d: ", i + 1);

        // Allocate memory for each name
        studentNames[i] = (char *)malloc(100 * sizeof(char));
        scanf("%s", studentNames[i]);
    }

    // Sort the names using selection sort
    for (int i = 0; i < numStudents - 1; i++) {
        for (int j = i + 1; j < numStudents; j++) {

```

```

        if (strcmp(studentNames[i], studentNames[j]) > 0) {
            // Swap names
            char *temp = studentNames[i];
            studentNames[i] = studentNames[j];
            studentNames[j] = temp;
        }
    }
}

// Display the sorted names
printf("\nSorted names in alphabetical order:\n");
for (int i = 0; i < numStudents; i++) {
    printf("%d. %s\n", i + 1, studentNames[i]);
}

// Free allocated memory
for (int i = 0; i < numStudents; i++) {
    free(studentNames[i]);
}
free(studentNames);

return 0;
}

```

## WEEK 10:

Q. 1 Write a C program to find length of string using pointers.

```

#include <stdio.h>
int main() {
    char str[100], * ptr;
    int count;
    printf("Enter any string: ");
    gets(str);
    // ptr pointing to first char of string
    ptr = str;
    // Initialize count to zero
    count = 0;
    // Run until null character is reached
    while ( *ptr != '\0') {
        count++;
        ptr++;
    }
    printf("The length of the string is: %d", count);
    return 0;
}

```

Q. 2 Write a C program to copy one string to another using pointer.

```

#include<stdio.h>

void copy_string(char*, char*);

main()
{
    char source[100], target[100];
    printf("Enter source string\n");
    gets(source);
    copy_string(target, source);
    printf("Target string is \"%s\"\n", target);
    return 0;
}

void copy_string(char *target, char *source)
{
    while(*source)
    {
        *target = *source;
        source++;
        target++;
    }
    *target = '\0';
}

```

Q. 3 Write a C program to concatenate two strings using pointers.

```

#include <stdio.h>

void concatenate(char *str1, char *str2) {
    // Move pointer to the end of the first string
    while (*str1) {
        str1++;
    }

    // Copy characters of the second string to the end of the first string
    while (*str2) {
        *str1 = *str2;
        str1++;
        str2++;
    }
    *str1 = '\0'; // Terminate the concatenated string
}

int main() {
    char string1[100], string2[50];

    // Input the strings
    printf("Enter the first string:\n");
    gets(string1);
    printf("Enter the second string:\n");
    gets(string2);

    // Concatenate the strings
    concatenate(string1, string2);

    // Display the concatenated string
    printf("Concatenated string: %s\n", string1);

    return 0;
}

```

Q. 4 Write a C program to compare two strings using pointers.

```

#include <iostream>
using namespace std;

int main()
{
    char string1[50], string2[50], *str1, *str2;
    int i, equal = 0;

    printf("Enter The First String: ");
    scanf("%s", string1);

    printf("Enter The Second String: ");
    scanf("%s", string2);

    str1 = string1;
    str2 = string2;

    while(*str1 == *str2)
    {
        if ( *str1 == '\0' || *str2 == '\0' )
            break;

        str1++;
        str2++;
    }

    if( *str1 == '\0' && *str2 == '\0' )
        printf("\n\nBoth Strings Are Equal.");
    else
        printf("\n\nBoth Strings Are Not Equal.");
}

```

Q. 5 WAP to find largest among three numbers using pointer

Q. 6 WAP to find largest among three numbers using pointer.

```

#include<stdio.h>

int main()
{
    int a,b,c,*pa, *pb, *pc;

```

```

printf("Enter three numbers:\n");
scanf("%d%d%d", &a,&b,&c);

/* Referencing */
pa= &a;
pb= &b;
pc= &c;
if(*pa > *pb && *pa > *pc)
{
    printf("Largest is: %d", *pa);
}
else if(*pb > *pc && *pb > *pc)
{
    printf("Largest is : %d", *pb);
}
else
{
    printf("Largest = %d", *pc);
}
return 0;
}

```

Q. 7 WAP to find factorial of a number using pointer.

```

#include<stdio.h>

void findFactorial(int,int *); //function
int main(){
    int i,factorial,n;

    printf("Enter a number: ");
    scanf("%d",&n);

    findFactorial(n,&factorial);
    printf("Factorial of %d is: %d",n,*factorial);

    return 0;
}

void findFactorial(int n,int *factorial){
    int i;

    *factorial =1;

    for(i=1;i<=n;i++)
        *factorial=*factorial*i;
}

```

Q. 8 Write a program to print largest even number present in an array using pointer to an array.

```

#include <stdio.h>

int findLargestEven(int *arr, int size) {
    int largestEven = -1; // Assuming all elements are non-negative

    for (int i = 0; i < size; i++) {
        if (arr[i] % 2 == 0 && arr[i] > largestEven) {
            largestEven = arr[i];
        }
    }

    return largestEven;
}

int main() {
    int size;

    // Input: Size of the array
    printf("Enter the size of the array: ");
    scanf("%d", &size);

    int arr[size];

    // Input: Elements of the array
    printf("Enter %d elements:\n", size);
    for (int i = 0; i < size; i++) {
        printf("Element %d: ", i + 1);
    }
}

```

```

        scanf("%d", &arr[i]);
    }

    // Finding the largest even number using pointer
    int *ptr = arr;
    int largestEven = findLargestEven(ptr, size);

    if (largestEven != -1) {
        printf("The largest even number in the array is: %d\n", largestEven);
    } else {
        printf("No even numbers found in the array.\n");
    }

    return 0;
}

```

Q. 9 WAP to find sum of elements of an array using array of pointer.

```

#include <stdio.h>
#include <malloc.h>

void main()
{
    int i, n, sum = 0;
    int *a;

    printf("Enter the size of array A \n");
    scanf("%d", &n);

    a = (int *) malloc(n * sizeof(int));

    printf("Enter Elements of the List \n");
    for (i = 0; i < n; i++)
    {
        scanf("%d", a + i);
    }

    /* Compute the sum of all elements in the given array */

    for (i = 0; i < n; i++)
    {
        sum = sum + *(a + i);
        /* this (a+i) is used to access the value stored at the address/
    }

    printf("Sum of all elements in array = %d\n", sum);
    return 0;
}

```

Q. 10 WAP to compute simple interest using pointers.

```

#include<stdio.h>
int main() {
    float p, t, r, SI;
    // p = principal, t = time, and r = rate
    // SI = value of the simple interest

    float *x, *y, *z; // These are the pointer variables

    printf("Enter the principal (amount), time, and rate::\n");
    scanf("%f%f%f", &p, &t, &r);

    x = &p;
    y = &t;
    z = &r;

    // It will calculate the value of simple interest
    SI = (*x * *y * *z) / 100;

    // It will produce the final output
    printf("\nSimple Interest = %.2f\n", SI);
    return 0;
}

```

Q. 11 Write a program to print largest even number present in an array using pointer to an array.

```

#include <stdio.h>

int findLargestEven(int *arr, int size) {
    int largestEven = -1; // Assuming all elements are non-negative

    for (int i = 0; i < size; i++) {
        if (arr[i] % 2 == 0 && arr[i] > largestEven) {
            largestEven = arr[i];
        }
    }

    return largestEven;
}

int main() {
    int size;

    // Input: Size of the array
    printf("Enter the size of the array: ");
    scanf("%d", &size);

    int arr[size];

    // Input: Elements of the array
    printf("Enter %d elements:\n", size);
    for (int i = 0; i < size; i++) {
        printf("Element %d: ", i + 1);
        scanf("%d", &arr[i]);
    }

    // Finding the largest even number using pointer
    int *ptr = arr;
    int largestEven = findLargestEven(ptr, size);

    if (largestEven != -1) {
        printf("The largest even number in the array is: %d\n", largestEven);
    } else {
        printf("No even numbers found in the array.\n");
    }

    return 0;
}

```

## WEEK 6:

Q. 1 Write a C function to return the maximum of three integers.

```

#include<stdio.h>
double max3(double x,double y,double z);
void main () {
    double i;
    double a,b,c;
    clrscr();
    printf("Enter the value of x,y,z:\n");
    scanf("%lf%lf%lf",&a,&b,&c);
    i= max3(a,b,c) ;
    printf("%lf",i);
    getch();
}
double max3(double x,double y,double z) {
    double max;
    if (x > y)
        max = x;
    else max = y;
    if (z > max)
        max = z;
    return max;
}

```

Q. 2 Write a C function to check if a given number is prime or not.

```

#include <stdio.h>

int main() {

```



```

int n, i, flag = 0;
printf("Enter a positive integer: ");
scanf("%d", &n);

if (n == 0 || n == 1)
    flag = 1;

for (i = 2; i <= n / 2; ++i) {

    if (n % i == 0) {
        flag = 1;
        break;
    }
}

if (flag == 0)
    printf("%d is a prime number.", n);
else
    printf("%d is not a prime number.", n);

return 0;
}

```

Q. 3 Write a C function to compute the factorial of a non-negative integer.

```

#include <stdio.h>

int main() {
    int n, i;
    unsigned long long fact = 1;
    printf("Enter an integer: ");
    scanf("%d", &n);

    if (n < 0)
        printf("Error! Factorial of a negative number doesn't exist.");
    else {
        for (i = 1; i <= n; ++i) {
            fact = i;
        }
        printf("Factorial of %d = %llu", n, fact);
    }

    return 0;
}

```

Q. 4 Write a C function to swap the values of two integers in actual arguments.

```

#include <stdio.h>

void swap(int,int );

int main ()
{
    int a, b;

    printf("Enter two numbers: ");
    scanf("%d%d", &a, &b);

    printf("Before Swapping : a=%d,b=%d\n",a,b);

    swap(&a,&b);

    printf("After Swapping : a=%d,b=%d\n",a,b);
    return 0;
}

void swap(inta,int b){
    int tmp;
    tmp =a;
    a=b;
    b=tmp;
}

```

```
    *b=tmp;
}
```

Q. 5 Write a C function to compute the sum and average of an array of integers.

```
#include <stdio.h>

int main(){

    int arr[100], size, sum;
    float avg;

    printf("Enter the size of the array: ");
    scanf("%d", &size);

    printf("Enter the array elements: ");
    for(int i = 0; i < size; i++){
        scanf("%d", &arr[i]);
    }

    sum = 0;

    for(int i = 0; i < size; i++){
        sum = sum + arr[i];
    }

    avg = sum / size;

    printf("Sum of array elements is: %d", sum);
    printf("\nAvg. of arrays elements is: %.2f", avg);

    return 0;
}
```

Q. 6 Write a C function to find the GCD (Greatest Common Divisor) of two non negative integers using Euclid's algorithm.

```
#include <stdio.h>
int main()
{
    int n1, n2, i, gcd;

    printf("Enter two integers: ");
    scanf("%d %d", &n1, &n2);

    for(i=1; i <= n1 && i <= n2; ++i)
    {
        if(n1%i==0 && n2%i==0)
            gcd = i;
    }

    printf("G.C.D of %d and %d is %d", n1, n2, gcd);

    return 0;
}
```

Q. 7 Write a C function to check if a given string is a valid palindrome, considering only alphanumeric characters and ignoring cases.

```
#include <stdio.h>
#include <string.h>

int main()
{
    char str[] = { "abbba" };

    int l = 0;
    int h = strlen(str) - 1;
    while (h > l) {
        if (str[l++] != str[h--]) {
```

```
        printf("%s is not a palindrome\n", str);
        return 0;

    }

}

printf("%s is a palindrome\n", str);

return 0;
}
```

Q. 8 Write a C function to calculate the sum and difference of two complex numbers.

```
#include <stdio.h>
int solve(int a, int b){
    int temp = a + b;
    b = a - *b;
    *a = temp;
}
int main(){
    int a
= 5, b = 8;
    solve(&a, &b);
    printf("a + b = %d and a - b = %d", a, b);
}
```