Day 4	Numbering System and Introduction to C

Objectives	To develop the skills necessary to convert between these systems, perform arithmetic operations, and understand their applications in computer science and digital electronics.
Outcomes	Students will be able to 1.Convert numbers from one system to another, including binary to decimal, decimal to binary, octal to hexadecimal, and vice versa. 2.Design and implement algorithms for number system conversions and arithmetic operations 3. Design and implement C programs using different tokens including keywords and variables.

1.Decimal to Octal

Convert the decimal numbers in the test cases to octal.
Convert a decimal (base-10) number provided by the user into its octal (base-8) equivalent. It repeatedly divides the decimal number by 8, records the remainders, and constructs the octal number from these remainders. The final octal number is displayed as the result.

Constraints:
Input should be Decimal Number.
It should be Positive Number.

Test Case 1:
Input:127
Expected Output:177

Test Case 2:
Input: 17
Expected Output: 21

Test Case 3:
Input: 8826

Expected Output: 21172

Test Case 4:
Input: 94753
Expected Output: 271041

Test Case 5:
Input: 511
Expected Output: 777

2.Binary to Decimal

Convert the binary numbers in the test cases to decimal.
Convert a binary (base-2) number provided by the user into its decimal (base-10) equivalent. It multiplies each binary digit by 2 raised to the power of its position, sums these products, and outputs the resulting decimal number.

Constraints:
Input should be Binary Number.
It should be Positive Number.

Test Case 1:
Input: 1111
Expected Output: 16

Test Case 2:
Input: 1101
Expected Output: 13

Test Case 3:
Input: 111010
Expected Output: 58

Test Case 4:
Input: 110100
Expected Output: 52

Test Case 5:
Input: 1010
Expected Output: 10

3.Binary to Hexadecimal

Convert the binary number in the test cases to hexadecimal.
Convert a binary (base-2) number provided by the user into its hexadecimal (base-16) equivalent. It groups the binary digits into sets of four from the right most corner, converts each group to its corresponding hexadecimal digit, and outputs the final hexadecimal number.

Constraints:
Input should be Binary Number.
It should be Positive Number.

Test Case 1:
Input: 110101
Output: 35

Test Case 2:
Input: 1110
Expected Output: E

Test Case 3:
Input: 1010
Expected Output: A

Test Case 4:
Input: 100111011
Expected Output: 13B

Test Case 5:
Input: 110110010010
Expected Output: D92

4.Octal to Binary

Convert the octal numbers in the test cases to binary.
Convert an octal (base-8) number provided by the user into its binary (base-2) equivalent. It translates each octal digit into its corresponding 3-bit binary representation and concatenates these binary groups. The final result is the complete binary number.

Constraints:
Input should be Octal Number.
It should be Positive Number.

Test Case 1:
Input: 41
Output: 100001

Test Case 2:
Input: 23
Expected Output: 10011

Test Case 3:
Input: 57
Expected Output: 101111

Test Case 4:
Input: 10
Expected Output: 1000

Test Case 5:
Input: 77
Expected Output: 111111

5.Hexadecimal to Decimal
Convert the hexadecimal numbers in the test cases to decimal.
Convert a hexadecimal (base-16) number provided by the user into its decimal (base-10) equivalent. It processes each hexadecimal digit by multiplying it with 16 raised to the power of its position and summing the results. The final output is the decimal representation of the given hexadecimal number.

Constraints:
Input should be Hexadecimal Number.
It should be Positive Number.

Test Case 1:
Input: 1DA6
Output: 7590

Test Case 2:
Input: 2F3
Expected Output: 755

Test Case 3:
Input: A7
Expected Output: 167

Test Case 4:
Input: C4F
Expected Output: 3151

Test Case 5:
Input: FF
Expected Output: 255

6.Classroom Numbering System

Imagine a teacher is organizing a classroom activity where students are assigned numbers for various tasks. To keep things simple and organized, the teacher wants to determine if a student's assigned number is odd or even. This helps in splitting the class into two groups: one group with students having odd numbers and another with students having even numbers.

To facilitate this, the teacher needs a quick way to check whether a number is odd or even. The teacher decides to create a simple algorithm, pseudocode and flowchart to automate this task.

Constraints

Input Requirements:
The input must be a valid positive integer.
The input string should contain only digits ('0'-'9').
The input string must not be empty.
Output: The program should determine if the number is odd or even based on the provided input.

7.Perfect Square Checker for Student Numbers

A teacher wants to organize a classroom activity where each student is assigned a unique number. To introduce a mathematical challenge, the teacher wants to determine if a student's assigned number is a perfect square. A perfect square is a number that can be expressed as the square of an integer.

Write a program to check if a student's assigned number is a perfect square. This will help the teacher categorize students based on whether their number has this mathematical property.

Constraints:

The input must be a valid positive integer.
The input string should contain only digits ('0'-'9').
The input string must not be empty.
Output:

The program should determine if the number is a perfect square based on the provided input.

8.Calculate the Area of a Rectangle

A carpenter needs to determine the area of a rectangular piece of wood to ensure it fits in a given space. The user provides the width and height of the rectangle. The area helps in planning the amount of material needed.

Constraints

The width should be a positive number with up to two decimal places.
The height should be a positive number with up to two decimal places.
The input strings must not be empty.

Algorithm

1.Start
2.Prompt the user to enter the width and height of the rectangle.
3.Read and validate the inputs:

- Ensure the inputs are not empty.
- Ensure both inputs are positive numbers with up to two decimal places.
- Convert the inputs to floating-point numbers.

4.Calculate the area using the formula: Area = Width * Height.
5.Print the calculated area.
6.End

9.Sum of Natural Numbers

A candy shop is giving out free candies to its customers. Each customer is assigned a unique number. To determine the number of candies a customer gets, the shop uses the following formula:

*Number of candies = (customer's number * (customer's number + 1)) / 2*

Write a program to calculate and print the number of candies a customer will receive based on their assigned number.

Constraints:

The input must be a valid positive integer.
The input string should contain only digits ('0'-'9').
The input string must not be empty.