LIST OF EXPERIMENTS

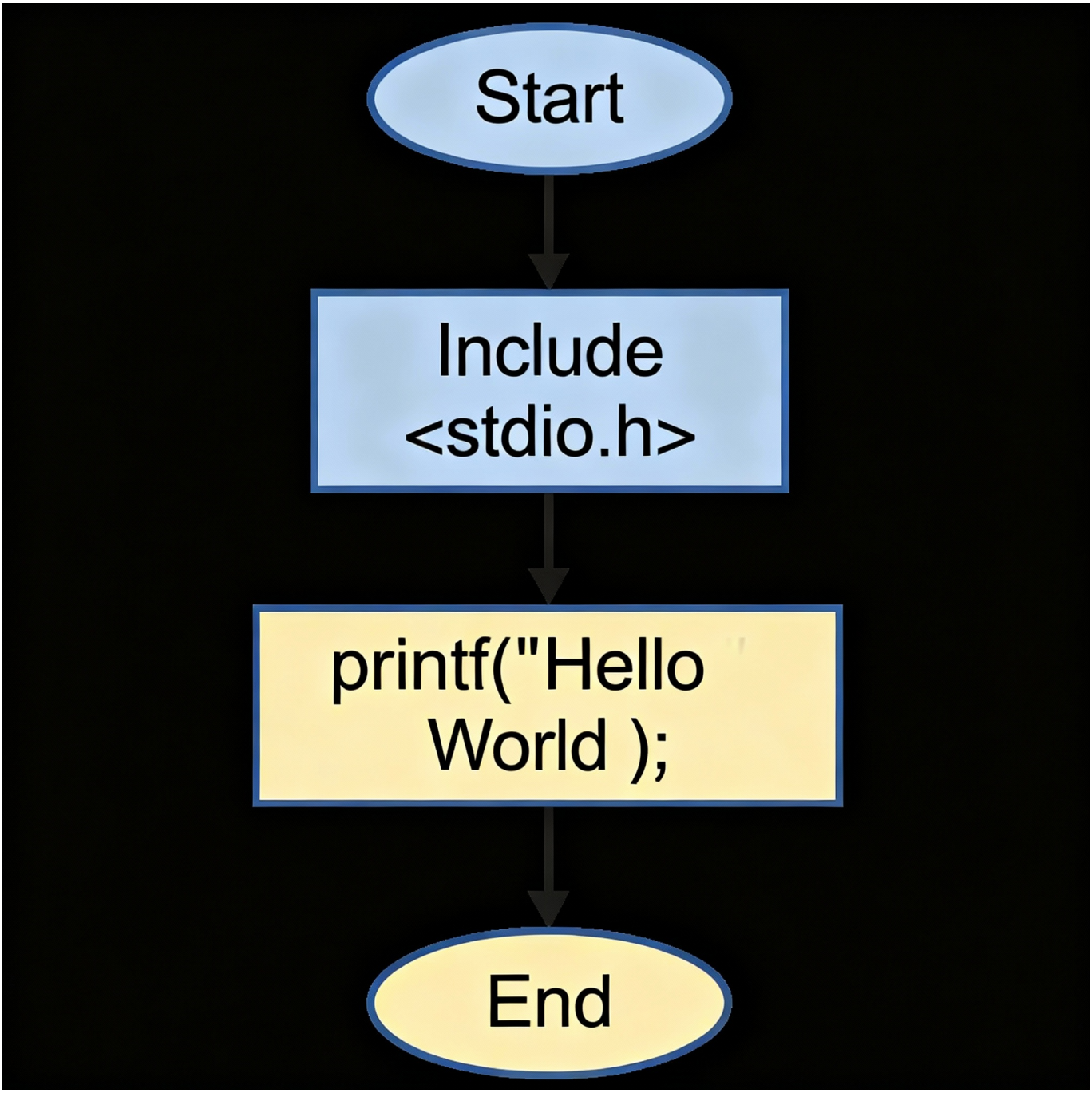
EXPERIMENT 1: Installation,Enviorment setup and starting with C language

1. Write a C program to print "helllo world".

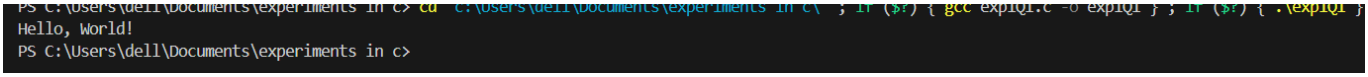
Algorithm

Step 1: Start Step 2: Include the standard input/output header file (stdio.h) Step 3: Define the main() function Step 4: Inside main(), use printf() to display "Hello, World!" Step 5: Return 0 to indicate successful program execution Step 6: Stop

Flowchart



Output

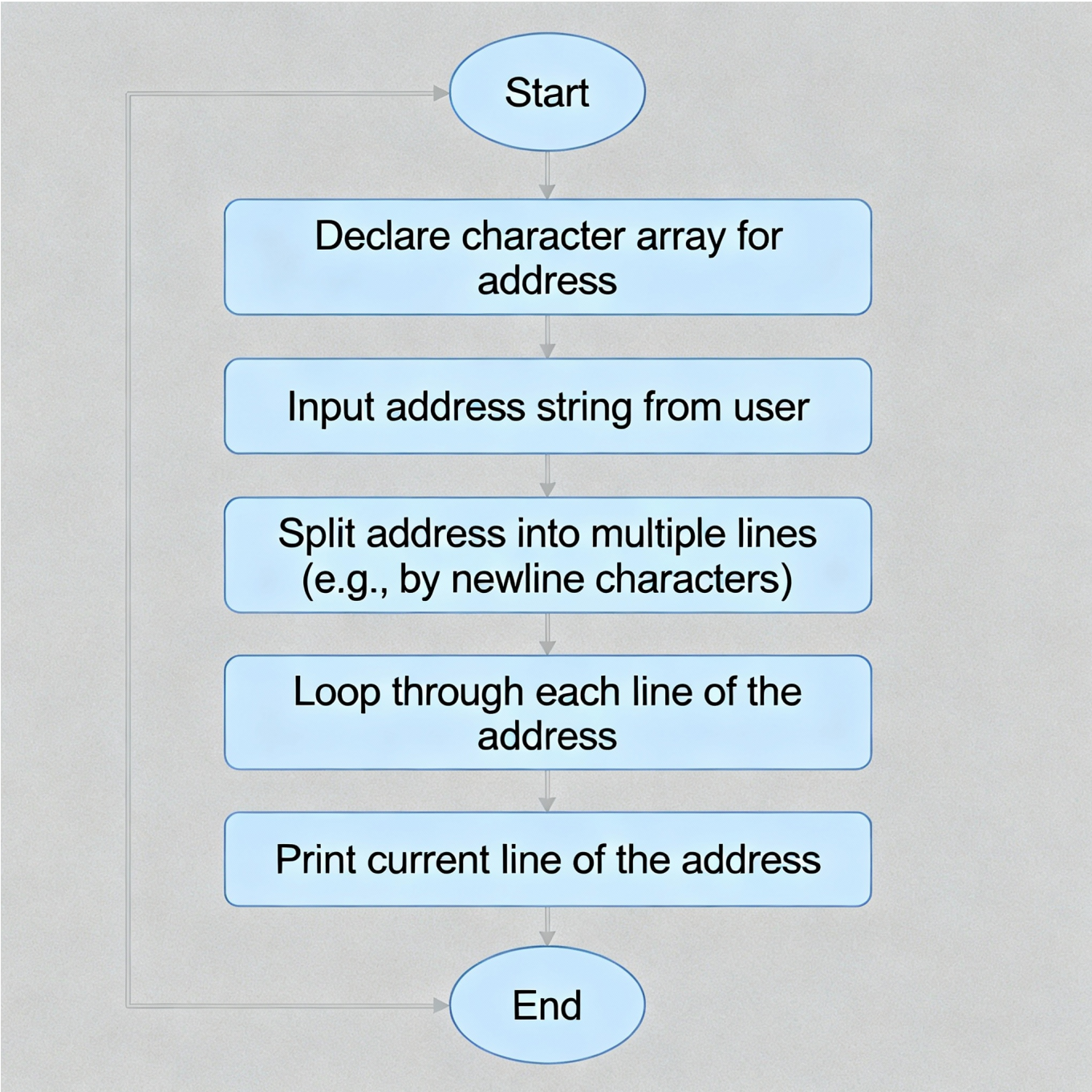


2.Write a C program to print adress in multiple lines

Algorithm

Step 1: Start
Step 2: Include the header file <stdio.h> for input and output functions
Step 3: Define the main() function
Step 4: Use printf() statements to print each line of the address
- Print the Name
- Print the Street
- Print the City and State
- Print the ZIP code (or Country)
Step 5: Return 0 to indicate successful execution
Step 6: Stop

Flowchart



Output

```
PS C:\Users\dell\Documents\experiments in c>
cd "c:\Users\dell\Documents\experiments in c\" ; if ($?) { gcc exp1Q2.c -o exp1Q2 } ; if ($?) { .\exp1Q2 }
Aditya Mishra
15A,
Kunraghat
Gorakhpur-273008
PS C:\Users\dell\Documents\experiments in c>
```

3:Write a C Program that prompts the user to enter their name and age.

Algorithm

Step 1: Start

Step 2: Include the header file <stdio.h>

Step 3: Declare a character array 'name' to store the user’s name

Step 4: Declare an integer variable 'age' to store the user’s age

Step 5: Display the message "Enter your name:"

Step 6: Read the user's name using scanf()

Step 7: Display the message "Enter your age:"

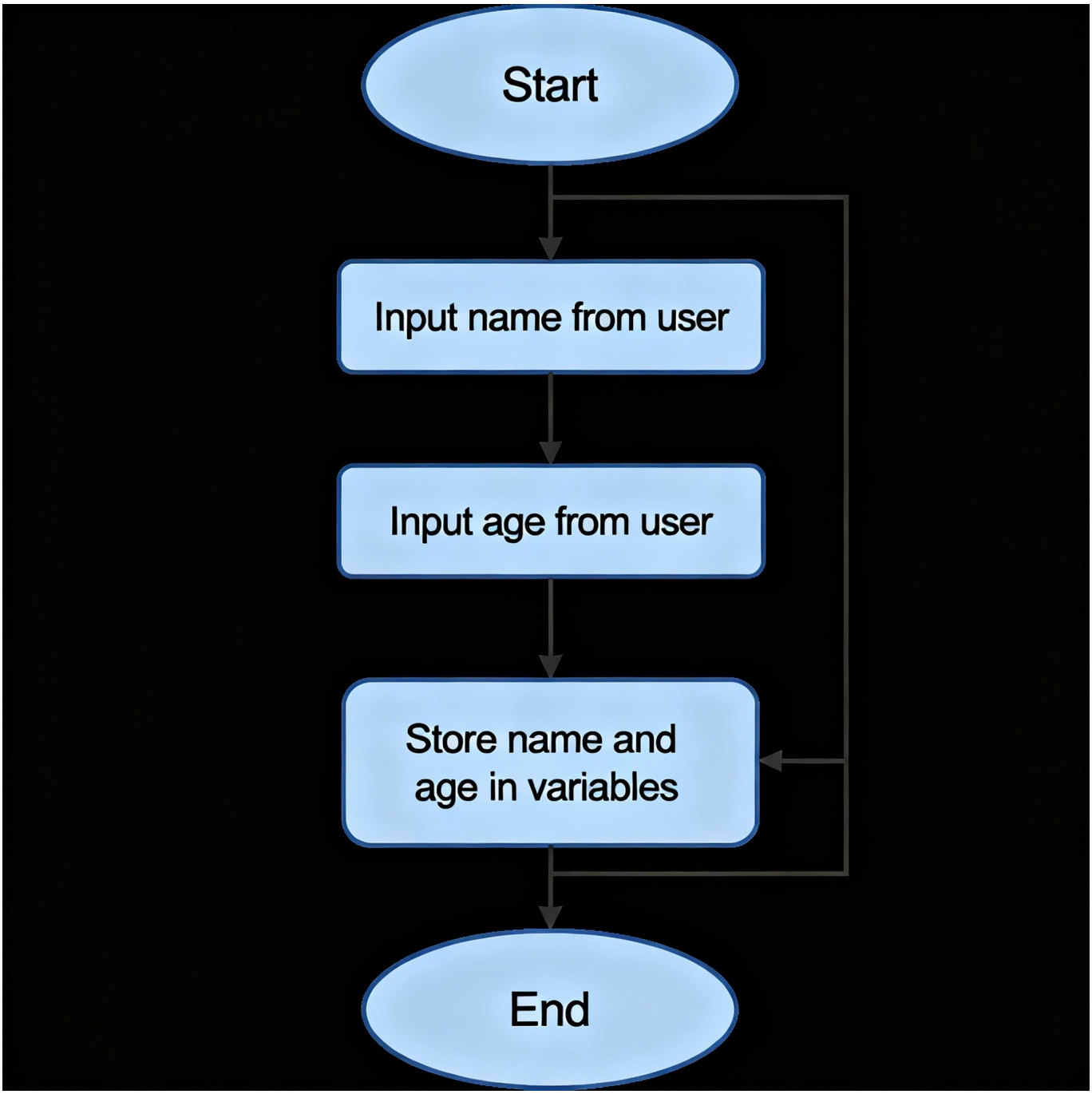
Step 8: Read the user's age using scanf()

Step 9: Display the output message "Hello ! You are years old."

Step 10: Return 0 to indicate successful program execution

Step 11: Stop

Flowchart



Output

```
PS C:\Users\dell\Documents\experiments in c>
cd "c:\Users\dell\Documents\experiments in c\" ; if ($?) { gcc exp1Q3.c -o exp1Q3 } ; if ($?) { .\exp1Q3 }
Enter your name: aditya
Enter your age: 19

Hello aditya! You are 19 years old.
PS C:\Users\dell\Documents\experiments in c>
```

4: Write a C program to add two numbers ,take numbers from users.

Algorithm

Step 1: Start

Step 2: Include the header file <stdio.h>

Step 3: Declare three integer variables: num1, num2, and sum

Step 4: Display the message "Enter first number:"

Step 5: Read the first number using scanf()

Step 6: Display the message "Enter second number:"

Step 7: Read the second number using scanf()

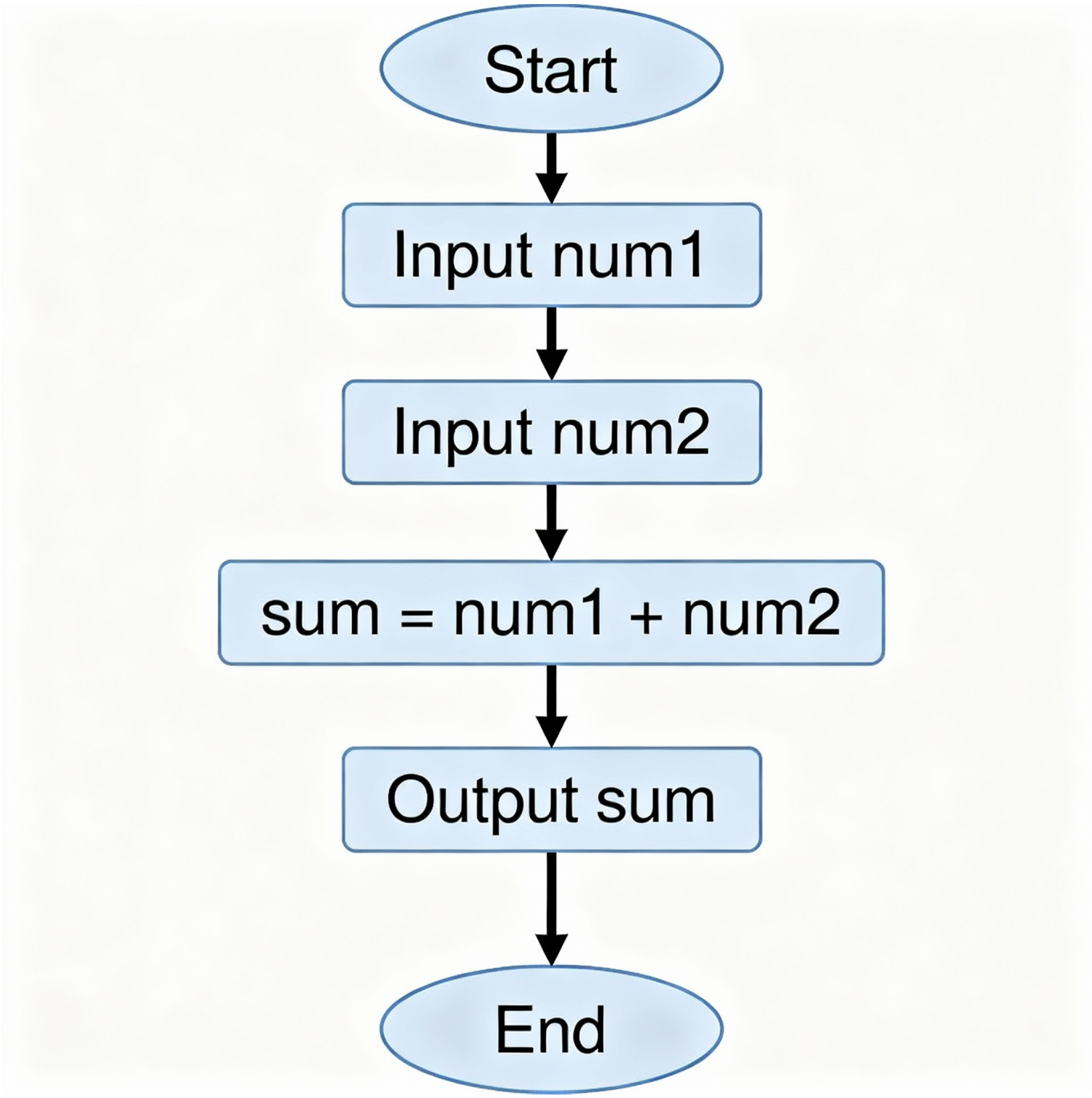
Step 8: Calculate the sum as sum = num1 + num2

Step 9: Display the result using printf()

Step 10: Return 0 to indicate successful execution

Step 11: Stop

Flowchart



Output

```
PS C:\Users\dell\Documents\experiments in c>
cd "c:\Users\dell\Documents\experiments in c\" ; if ($?) { gcc exp1Q4.c -o exp1Q4 } ; if ($?) { .\exp1Q4 }
Enter first number: 12
Enter second number: 13
The sum of 12 and 13 is 25.
PS C:\Users\dell\Documents\experiments in c>
```

EXPERIMENT 2: OPERATORS

1: Write a C program to Calculate the area and perimeter of a Rectangle based on its length and width.

Algorithm

Step 1: Start

Step 2: Include the header file <stdio.h>

Step 3: Declare three variables: length, width, area, and perimeter

Step 4: Display the message "Enter the length of the rectangle:"

Step 5: Read the length using scanf()

Step 6: Display the message "Enter the width of the rectangle:"

Step 7: Read the width using scanf()

Step 8: Calculate the area using the formula: area = length * width

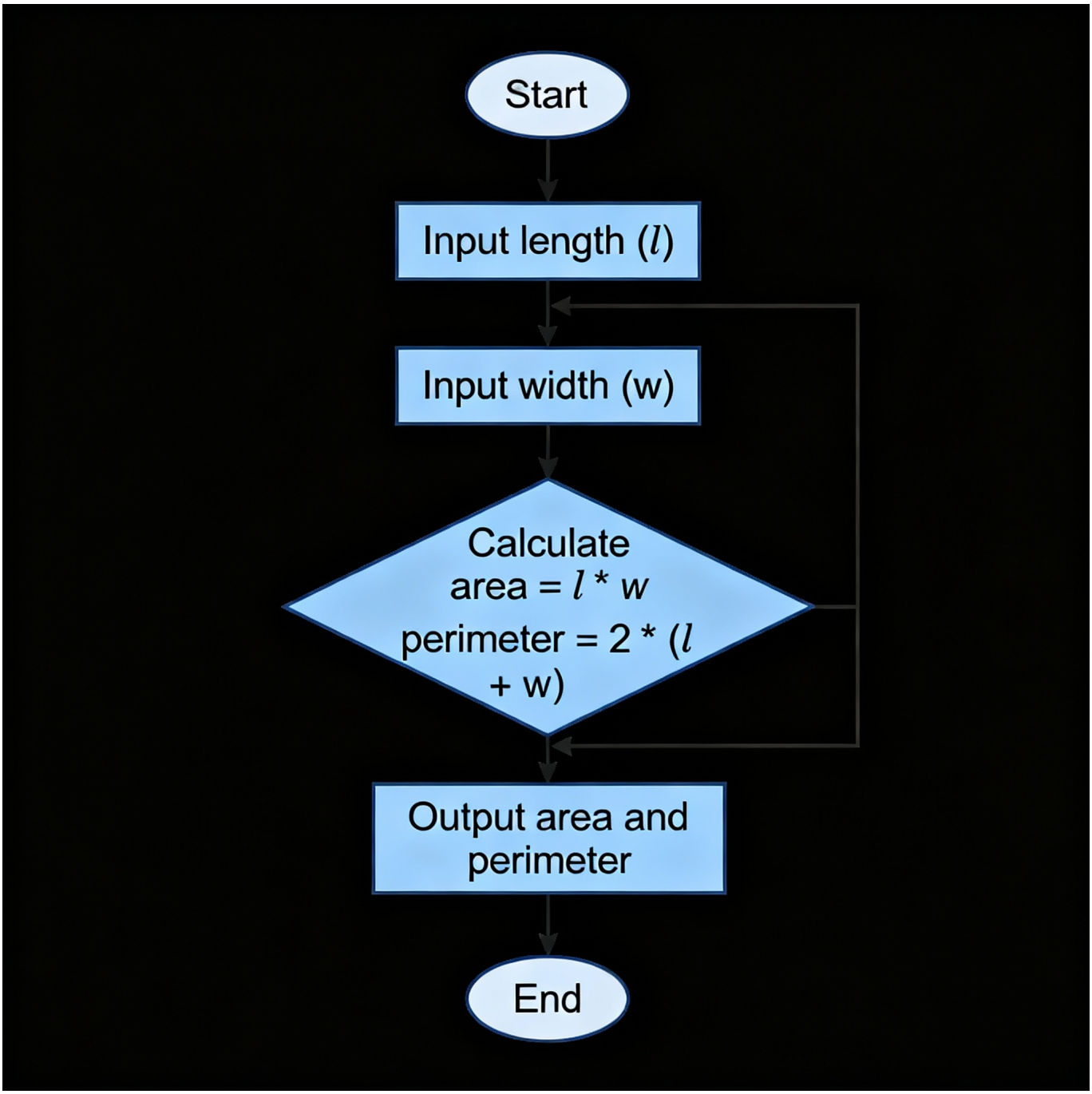
Step 9: Calculate the perimeter using the formula: perimeter = 2 * (length + width)

Step 10: Display the area and perimeter using printf()

Step 11: Return 0 to indicate successful execution

Step 12: Stop

Flowchart



Output

```
PS C:\Users\dell\Documents\experiments in c>
cd "c:\Users\dell\Documents\experiments in c\exp2\" ; if ($?) { gcc exp2Q1.c -o exp2Q1 } ; if ($?) { .\exp2Q1 }
Enter the length of the rectangle: 12
Enter the width of the rectangle: 13

Area of the rectangle = 156.00
Perimeter of the rectangle = 50.00
PS C:\Users\dell\Documents\experiments in c\exp2>
```

2:Write a c program to convert temperature from Celsius to Fahrenheit using the formula F= (C*9/5)+32

Algorithm

Step 1: Start

Step 2: Include the header file <stdio.h>

Step 3: Declare two variables: celsius and fahrenheit

Step 4: Display the message "Enter temperature in Celsius:"

Step 5: Read the Celsius temperature using scanf()

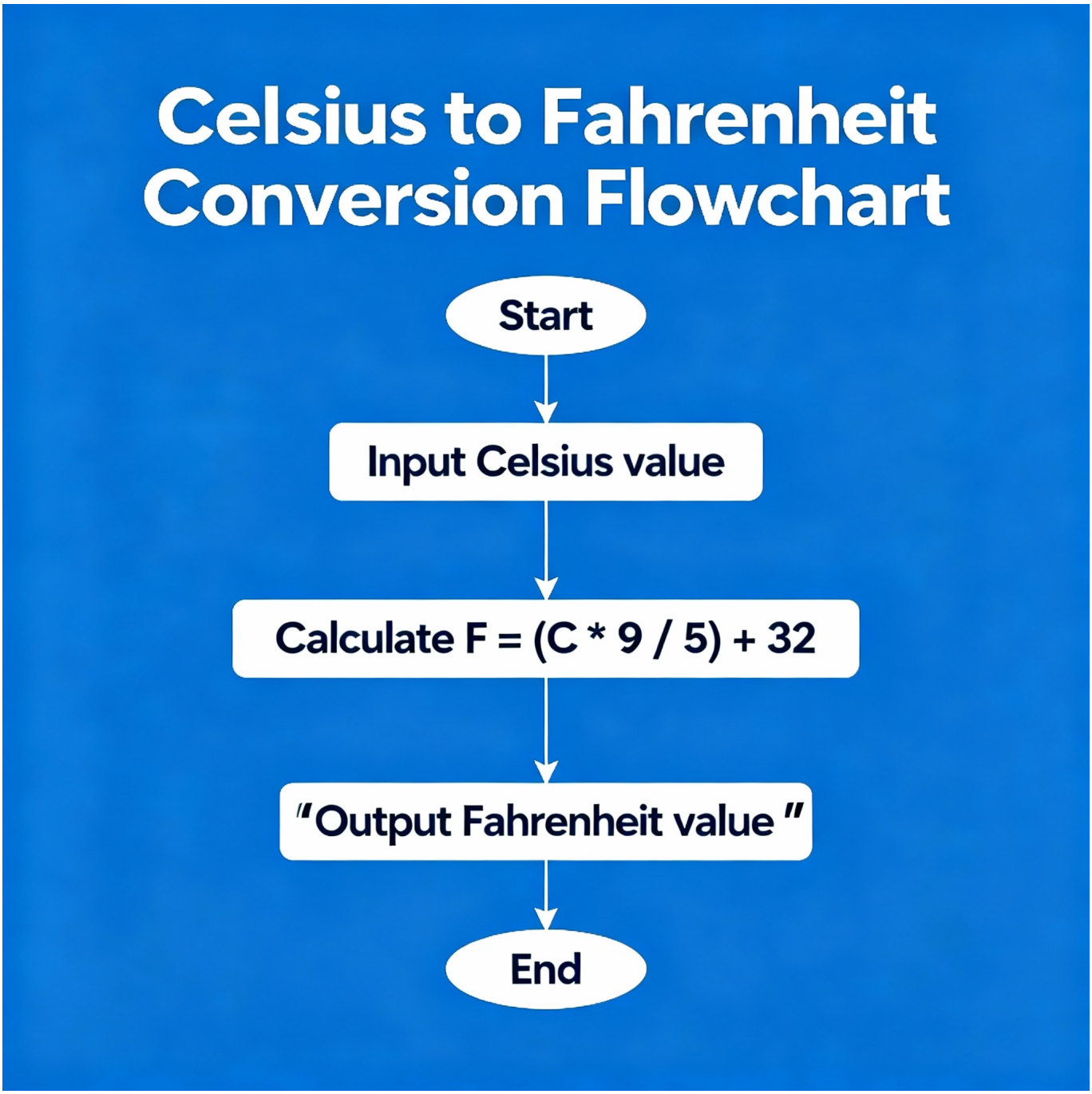
Step 6: Calculate Fahrenheit using the formula: fahrenheit = (celsius * 9 / 5) + 32

Step 7: Display the Fahrenheit temperature using printf()

Step 8: Return 0 to indicate successful execution

Step 9: Stop

Flowchart



Output

```
PS C:\Users\dell\Documents\experiments in c>
cd "c:\Users\dell\Documents\experiments in c\exp2\" ; if ($?) { gcc exp2Q2.c -o exp2Q2 } ; if ($?) { .\exp2Q2 }
Enter temperature in Celsius: 79
Temperature in Fahrenheit = 174.20
PS C:\Users\dell\Documents\experiments in c\exp2>
```

EXPERIMENT 3.1: Conditional Statements

1. WAP to take check if the triangle is valid or not.if the validity is established ,do check if the tringle is isosceles,equilateral,right angle or scalene. Take sides of the triangle as input from a user.

Algorithm

Start

Input three sides: a, b, c

Check triangle validity using the condition:

If (a + b > c) AND (a + c > b) AND (b + c > a) → Triangle is valid Else → Triangle is not valid and stop.

If valid:

If a == b == c → Triangle is Equilateral

Else If a == b OR b == c OR a == c → Triangle is Isosceles

Else → Triangle is Scalene

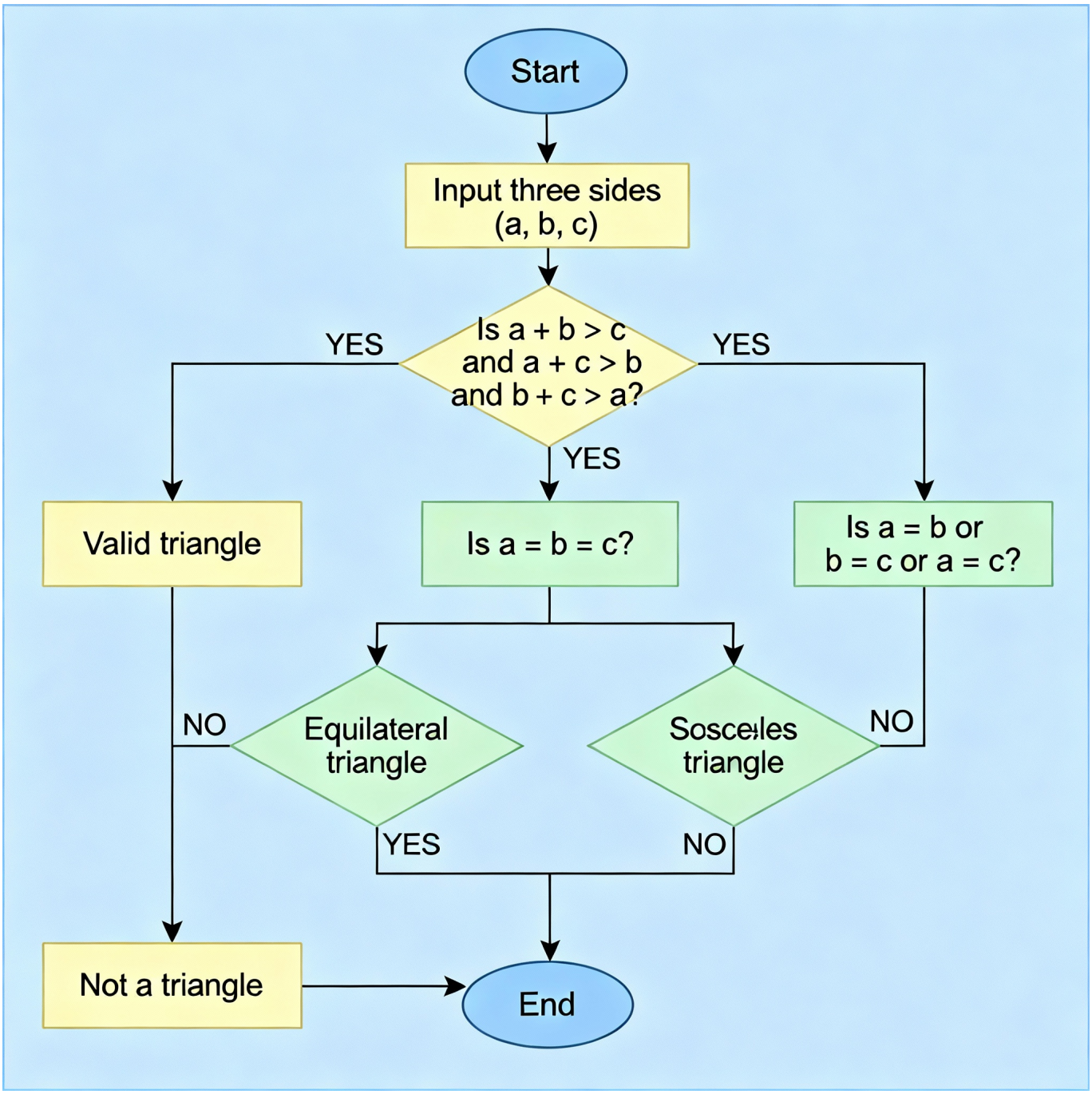
Check right-angle condition using Pythagoras theorem:

If $a^2 + b^2 == c^2$ OR $a^2 + c^2 == b^2$ OR $b^2 + c^2 == a^2$ → Triangle is Right-angled

Display result

Stop

Flowchart



Output

```
PS C:\Users\dell\Documents\experiments in c>
cd "c:\Users\dell\Documents\experiments in c\exp3_1\" ; if ($?) { gcc exp3Q1.c -o exp3Q1 } ; if ($?) { .\exp3Q1 }
Enter the three sides of the triangle: 2.0 3.0 2.0
Triangle is valid.
It is an Isosceles triangle.
PS C:\Users\dell\Documents\experiments in c\exp3_1>
```

2:WAP to compute the BMI index of the person and print the BMI values as per the following ranges. You can use the following formula to compute BMI=weight(kgs)Height(Mts)*Height(Mts).

	BMI
Starvation	<15
Anorexic	15.1 to 17.5
Underweight	17.6 to 18.5
Ideal	18.6 to 24.9
Overweight	25 to 25.9
Obese	30 to 39.9

Morbidity Obese	40.0 above

Algorithm

Step 1: Start

Step 2: Include the header file <stdio.h>

Step 3: Declare variables: weight, height, and bmi

Step 4: Prompt the user to enter weight (in kilograms)

Step 5: Read weight using scanf()

Step 6: Prompt the user to enter height (in meters)

Step 7: Read height using scanf()

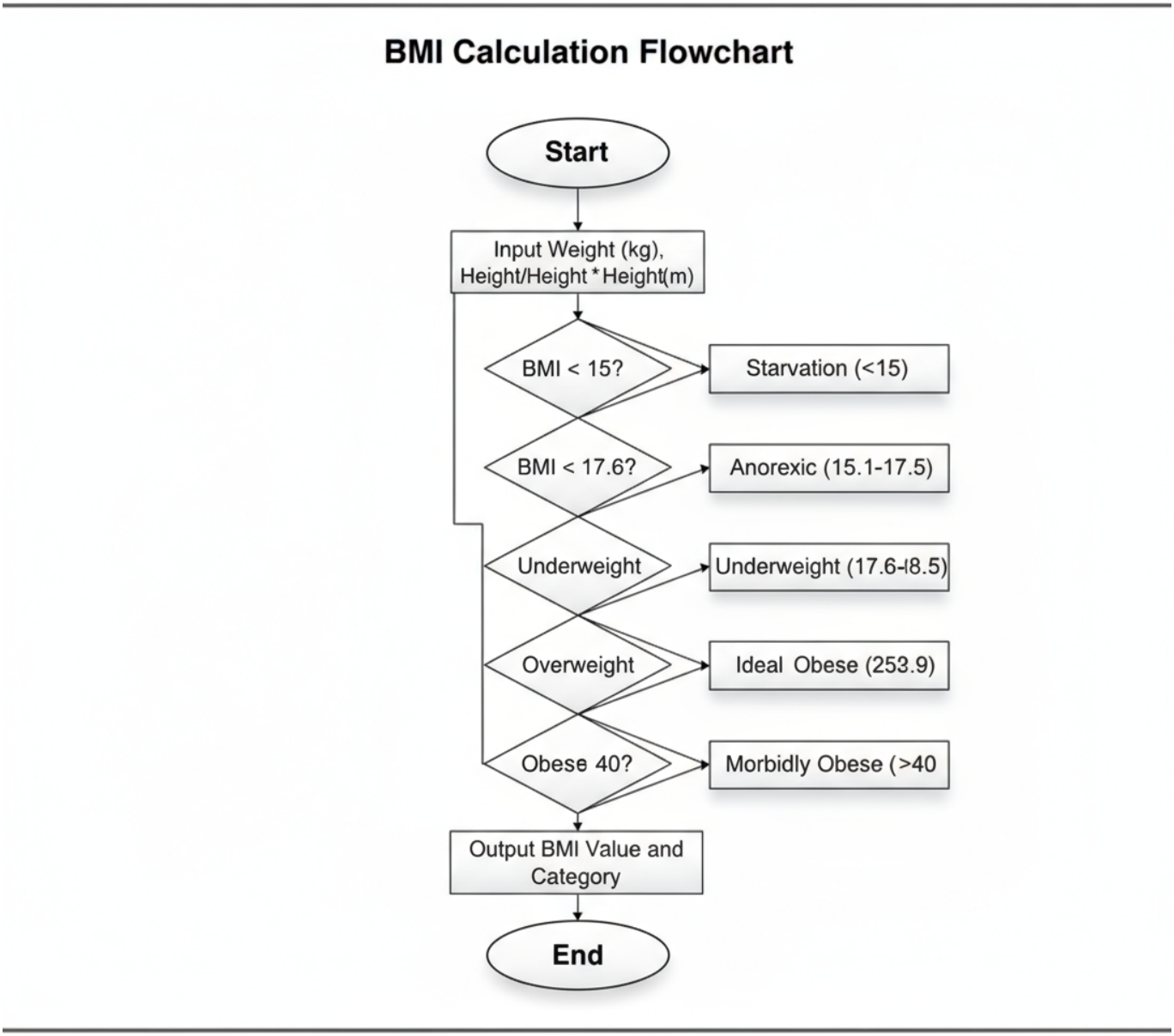
Step 8: Compute BMI using the formula:
bmi = weight / (height * height)

Step 9: Display the BMI value

Step 10: Determine the BMI category:
- If bmi < 15 → "Starvation"
- Else if bmi >= 15.1 and bmi <= 17.5 → "Anorexic"
- Else if bmi >= 17.6 and bmi <= 18.5 → "Underweight"
- Else if bmi >= 18.6 and bmi <= 24.9 → "Ideal"
- Else if bmi >= 25 and bmi <= 25.9 → "Overweight"
- Else if bmi >= 30 and bmi <= 39.9 → "Obese"
- Else if bmi >= 40 → "Morbidly Obese"

Step 11: Stop

Flowchart



Output

```
PS C:\Users\dell\Documents\experiments in c>
cd "c:\Users\dell\Documents\experiments in c\exp3_1\" ; if ($?) { gcc exp3Q2.c -o exp3Q2 } ; if ($?) { .\exp3Q2 }
Enter your weight (in kg): 60
Enter your height (in meters): 159

Your BMI is: 0.00
Category: Starvation
PS C:\Users\dell\Documents\experiments in c\exp3_1>
```

3: WAP to check if three points (x1,y1),(x2,y2)and(x3,y3)are collinear or not.

Algorithm

Step 1: Start

Step 2: Include the header file <stdio.h>

Step 3: Declare six variables: x1, y1, x2, y2, x3, y3

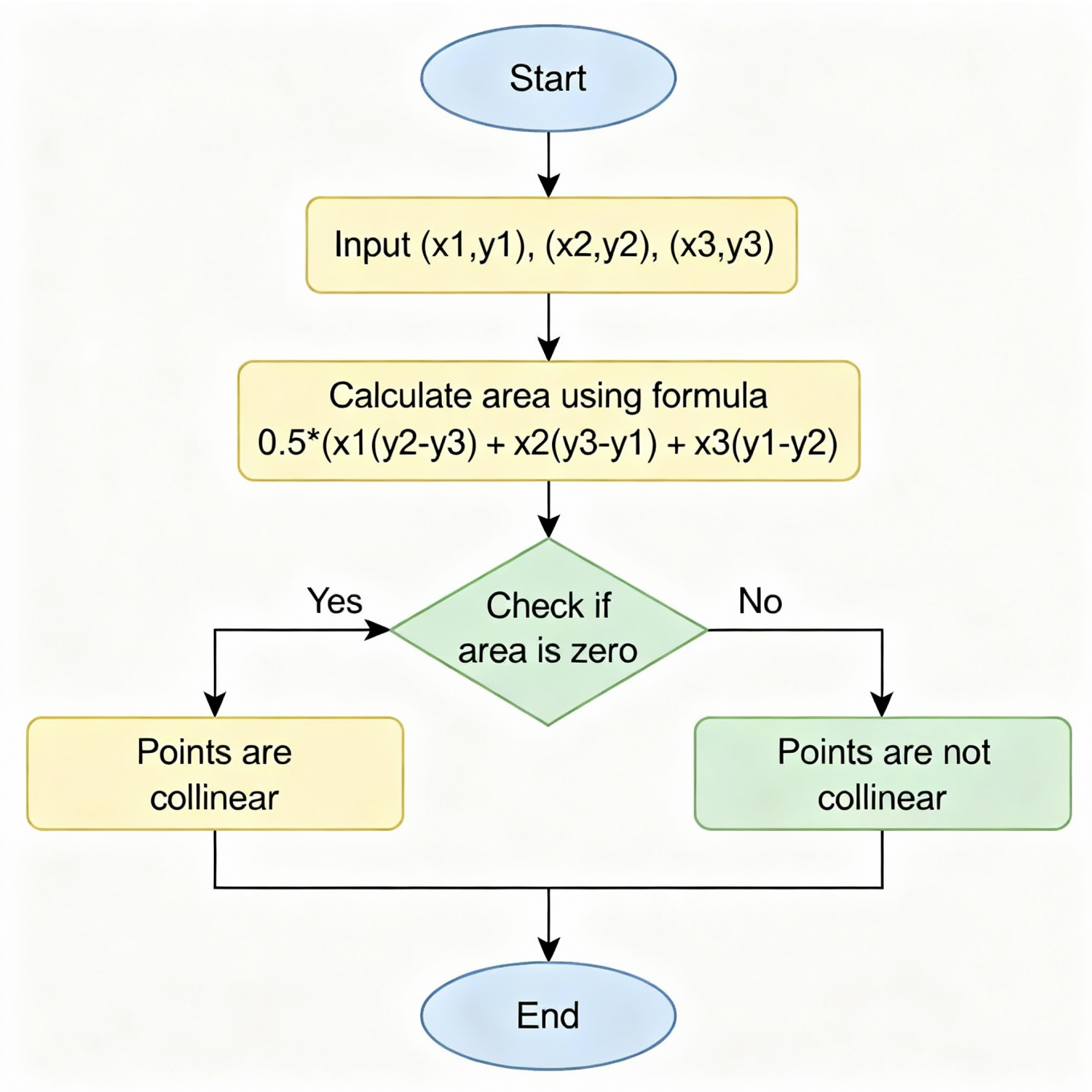
Step 4: Prompt the user to enter coordinates of the three points

Step 5: Read values of x1, y1, x2, y2, x3, y3 using scanf()

Step 6: Calculate the value: area = x1*(y2 - y3) + x2*(y3 - y1) + x3*(y1 - y2)

Step 7: If area == 0 Display "The points

Flowchart



Output

```
PS C:\Users\dell\Documents\experiments in c>
cd "c:\Users\dell\Documents\experiments in c\exp3_1\" ; if ($?) { gcc exp3Q3.c -o exp3Q3 } ; if ($?) { .\exp3Q3 }
Enter coordinates of first point (x1, y1): 2,2
Enter coordinates of second point (x2, y2): Enter coordinates of third point (x3, y3): The points are not collinear.
PS C:\Users\dell\Documents\experiments in c\exp3_1>
```

4:According to the gregorian calender,it was Monday on the date 01/01/01. if any year is input through the keyboard write a program to find out what is the day on 1st january of this year.

Algorithm

Step 1: Start

Step 2: Include the header file <stdio.h>

Step 3: Declare variables: - year (input year) - total_days (number of days between 01/01/01 and 01/01/year) - leap_years (number of leap years before the given year) - normal_years (number of normal years before the given year) - day (to store remainder when total_days is divided by 7)

Step 4: Input the year from the user.

Step 5: Calculate the number of leap years as: leap_years = (year - 1) / 4 - (year - 1) / 100 + (year - 1) / 400

Step 6: Calculate the number of normal years as: normal_years = (year - 1) - leap_years

Step 7: Compute total_days = (leap_years * 366) + (normal_years * 365)

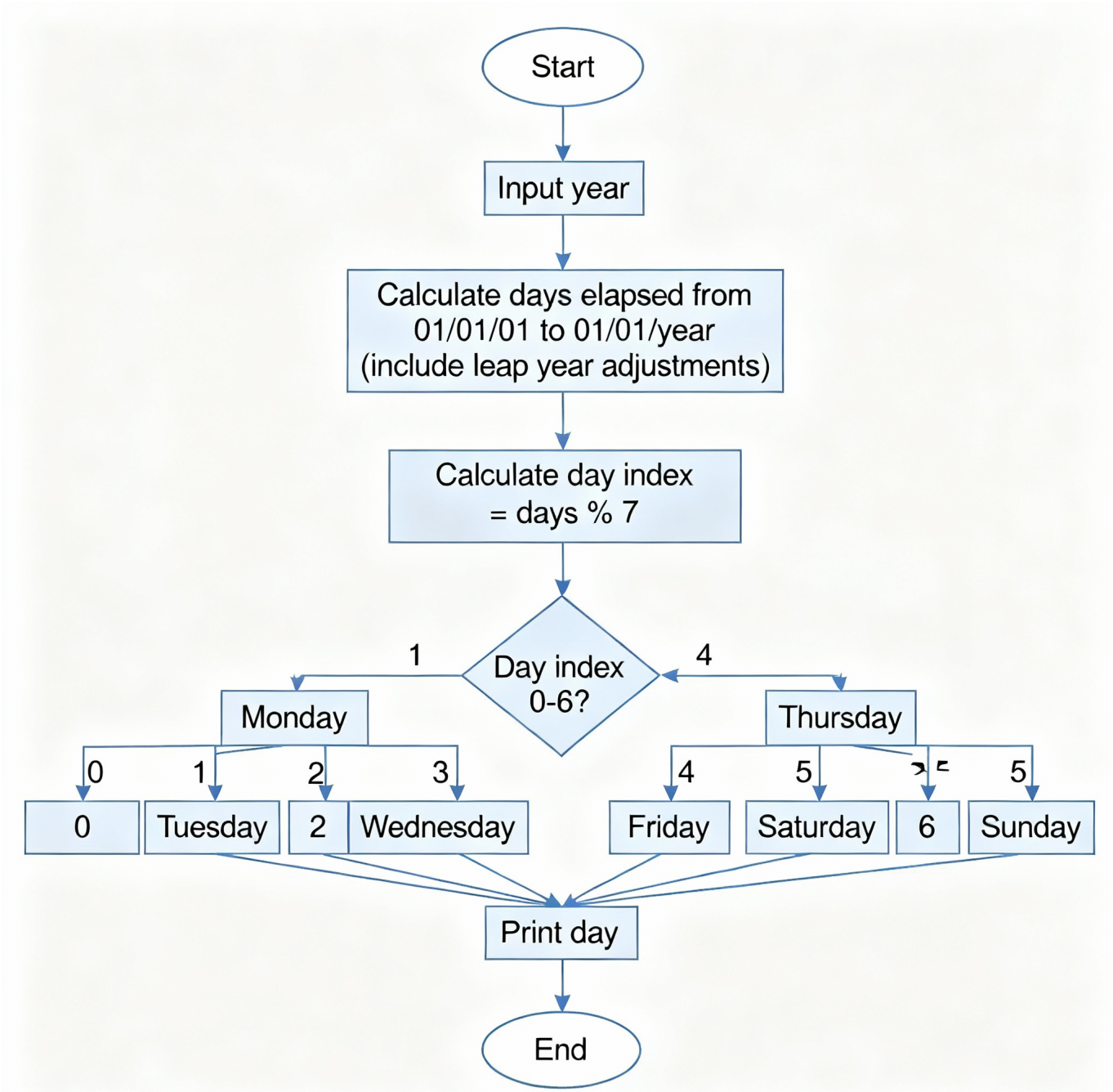
Step 8: Compute day = total_days % 7

Step 9: Since 01/01/01 was Monday, map the remainder to a day: 0 → Monday
1 → Tuesday
2 → Wednesday
3 → Thursday
4 → Friday
5 → Saturday
6 → Sunday

Step 10: Display the corresponding day of the week.

Step 11: Stop

Flowchart



Output

```
PS C:\Users\dell\Documents\experiments in c>
cd "c:\Users\dell\Documents\experiments in c\exp3_1\" ; if ($?) { gcc exp3Q4.c -o exp3Q4 } ; if ($?) { .\exp3Q4 }
Enter year: 2004
1st January 2004 is Thursday
PS C:\Users\dell\Documents\experiments in c\exp3_1>
```

5:WAP using ternary operator,the user should input the length and breadth of a rectangle,one has to find out which rectangle has the highest perimeter. The minimum no of rectangles should be three.

Algorithm

Step 1: Start

Step 2: Include the header file <stdio.h>

Step 3: Declare variables: l1, b1, l2, b2, l3, b3 for length and breadth p1, p2, p3 for perimeters max for the highest perimeter

Step 4: Input length and breadth for all three rectangles

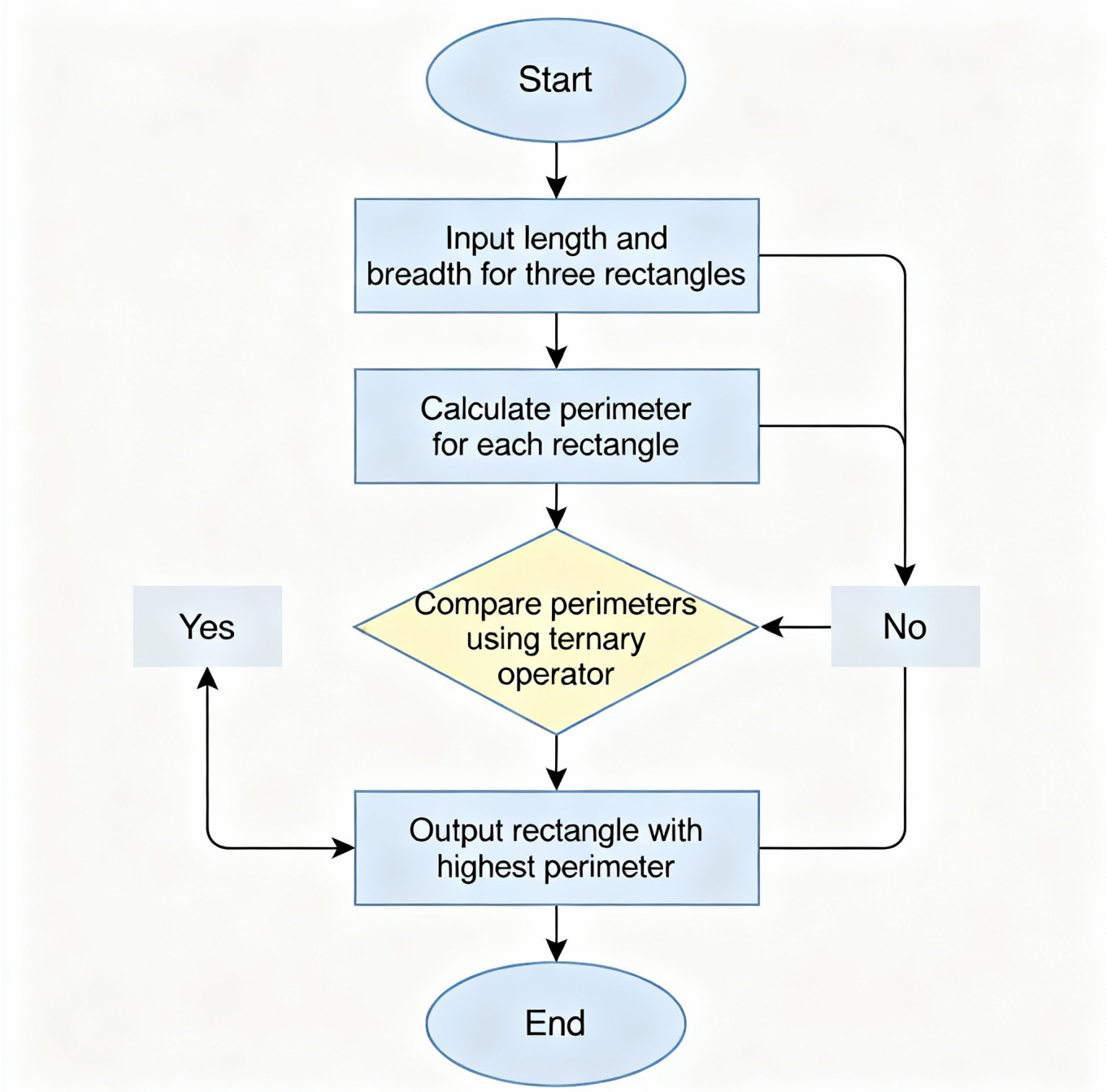
Step 5: Compute perimeters using: p1 = 2 * (l1 + b1) p2 = 2 * (l2 + b2) p3 = 2 * (l3 + b3)

Step 6: Find the highest perimeter using the ternary operator: max = (p1 > p2 && p1 > p3) ? p1 : (p2 > p3 ? p2 : p3)

Step 7: Display which rectangle has the highest perimeter

Step 8: Stop

Flowchart



Output

```
PS C:\Users\dell\Documents\experiments in c>
cd "c:\Users\dell\Documents\experiments in c\exp3_1\" ; if ($?) { gcc exp3Q5.c -o exp3Q5 } ; if ($?) { .\exp3Q5 }
Enter length and breadth of Rectangle 1: 12 5
Enter length and breadth of Rectangle 2: 13 5
Enter length and breadth of Rectangle 3: 5 6

Perimeter of Rectangle 1 = 34.00
Perimeter of Rectangle 2 = 36.00
Perimeter of Rectangle 3 = 22.00

Rectangle 2 has the highest perimeter: 36.00
PS C:\Users\dell\Documents\experiments in c\exp3_1>
```

EXPERIMENT 3.2: LOOPS

1:WAP to enter numbers till the user wants. At the end,it should display the count of positive,negative and zeroes entered.

Algorithm

Step 1: Start

Step 2: Include the header file <stdio.h>

Step 3: Declare integer variables: num (for input number), pos = 0, neg = 0, zero = 0 (for counting), choice (for user decision)

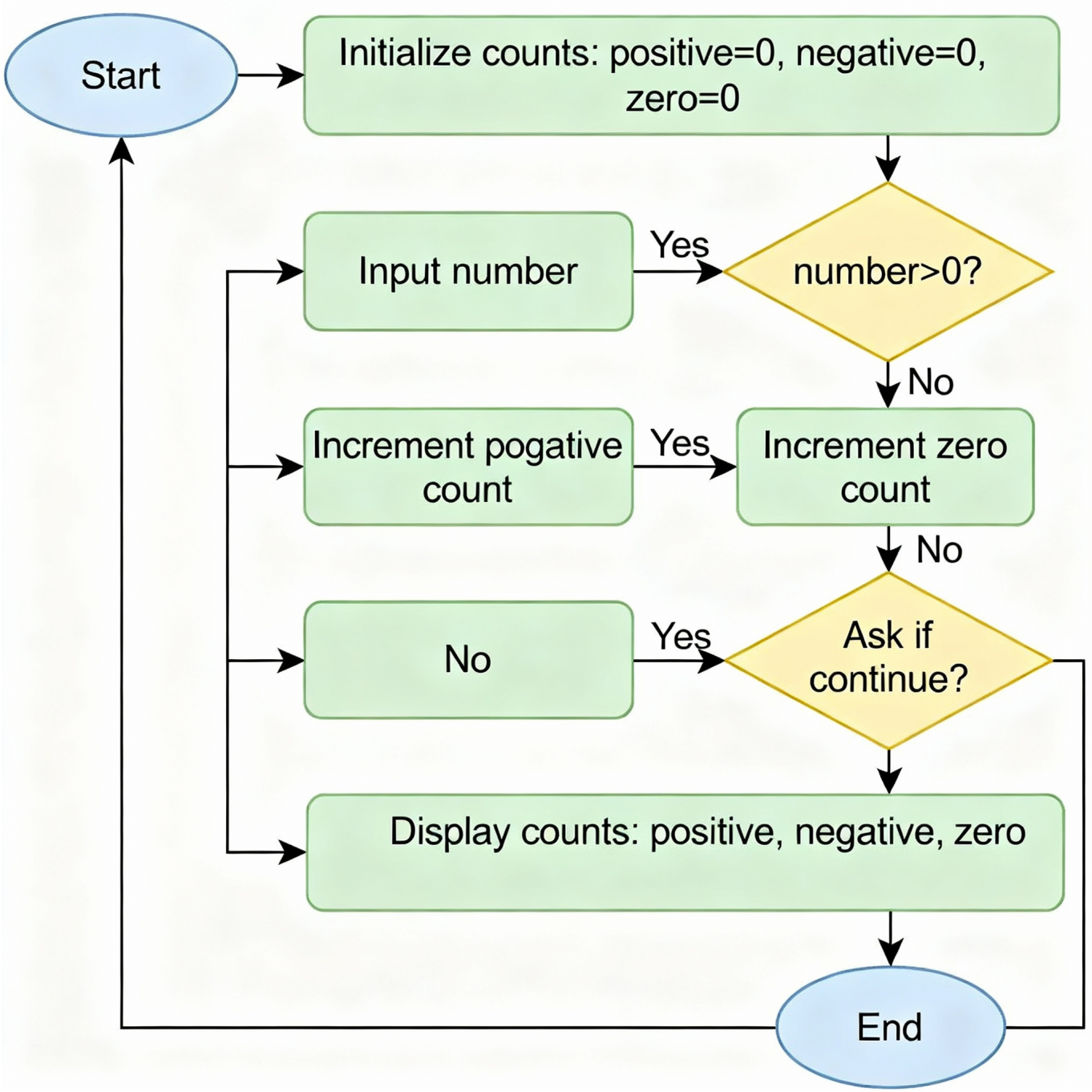
Step 4: Repeat the following steps using a loop: a) Prompt the user to enter a number b) Read the number using scanf() c) If number > 0 → increment pos Else if number < 0 → increment neg Else → increment zero d) Ask the user if they want to continue (1 = yes, 0 = no) e) Read the choice

Step 5: Continue the loop while choice == 1

Step 6: After the loop ends, display: - Total count of positive numbers - Total count of negative numbers - Total count of zeroes

Step 7: Stop

Flowchart



Output

```
PS C:\Users\dell\Documents\experiments in c>
cd "c:\Users\dell\Documents\experiments in c\exp3_1\exp3_2\" ; if ($?) { gcc exp3_2Q1.c -o exp3_2Q1 } ; if ($?) { .\exp3_2Q1 }
Enter a number: 12
Do you want to continue? (1 = Yes, 0 = No): 1
Enter a number: 13
Do you want to continue? (1 = Yes, 0 = No): 0

Count of Positive Numbers: 2
Count of Negative Numbers: 0
Count of Zeroes: 0
PS C:\Users\dell\Documents\experiments in c\exp3_1\exp3_2>
```

2:WAP to print the multiplication table of the numbers entered by the user.iy should be in the correct formatting. Num*1=Num.

Algorithm

Step 1: Start

Step 2: Include the header file <stdio.h>

Step 3: Declare integer variables: num and i

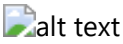
Step 4: Prompt the user to enter a number

Step 5: Read the number using scanf()

Step 6: Use a loop (for i = 1 to 10) - Calculate product = num * i - Display the result in the format: num * i = product

Step 7: Stop

Flowchart

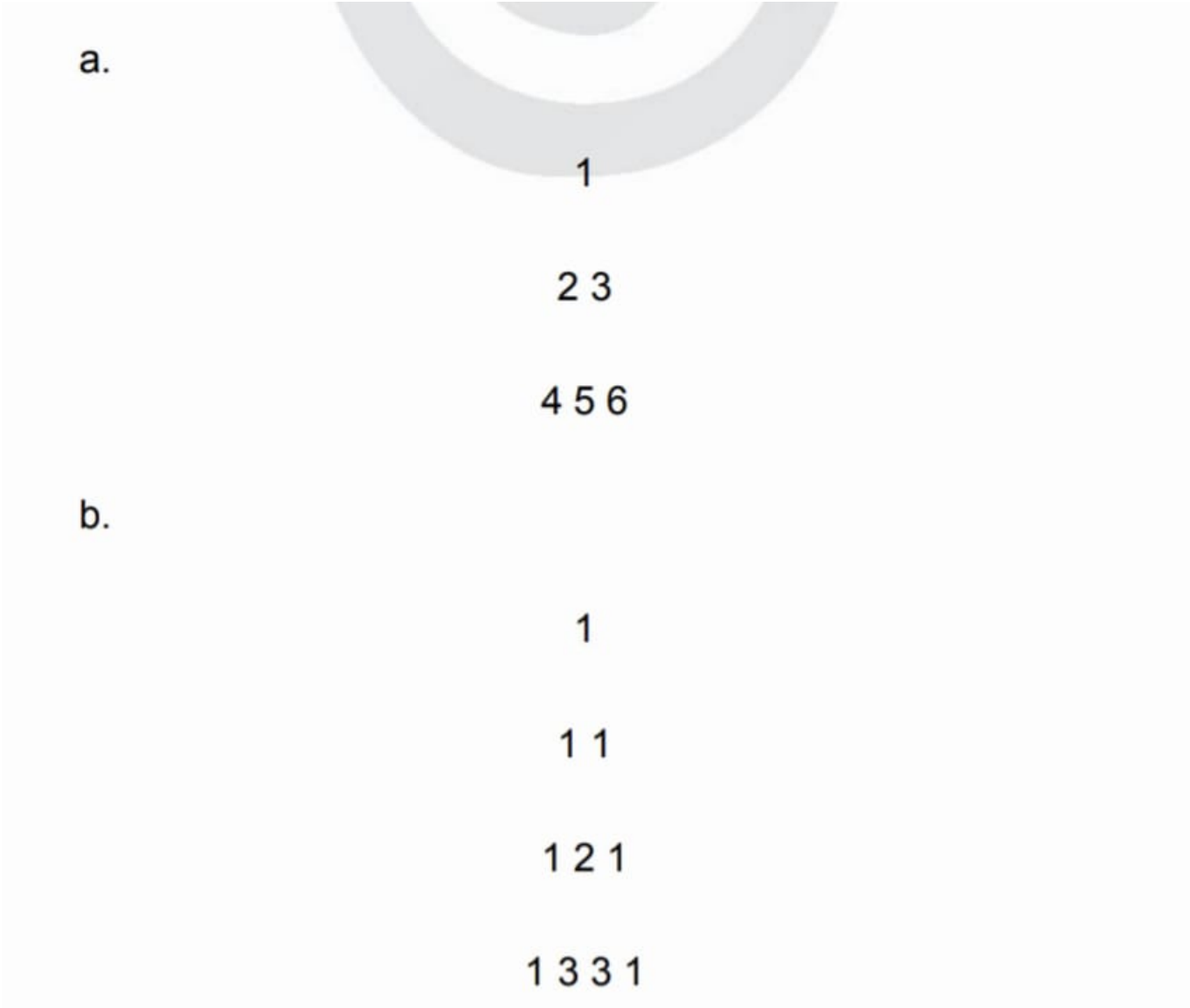


Output

```
PS C:\Users\dell\Documents\experiments in c>
cd "c:\Users\dell\Documents\experiments in c\exp3_1\exp3_2\" ; if ($?) { gcc exp3_2Q2.c -o exp3_2Q2 } ; if ($?) { .\exp3_2Q2 }
Enter a number: 12

Multiplication Table of 12:
12 * 1 = 12
12 * 2 = 24
12 * 3 = 36
12 * 4 = 48
12 * 5 = 60
12 * 6 = 72
12 * 7 = 84
12 * 8 = 96
12 * 9 = 108
12 * 10 = 120
PS C:\Users\dell\Documents\experiments in c\exp3_1\exp3_2>
```

3: WAP to generate the following set of output.



Algorithm

a)

Step 1: Start

Step 2: Initialize variable num = 1 to keep track of current number

Step 3: Loop for rows from 1 to 3 (i)

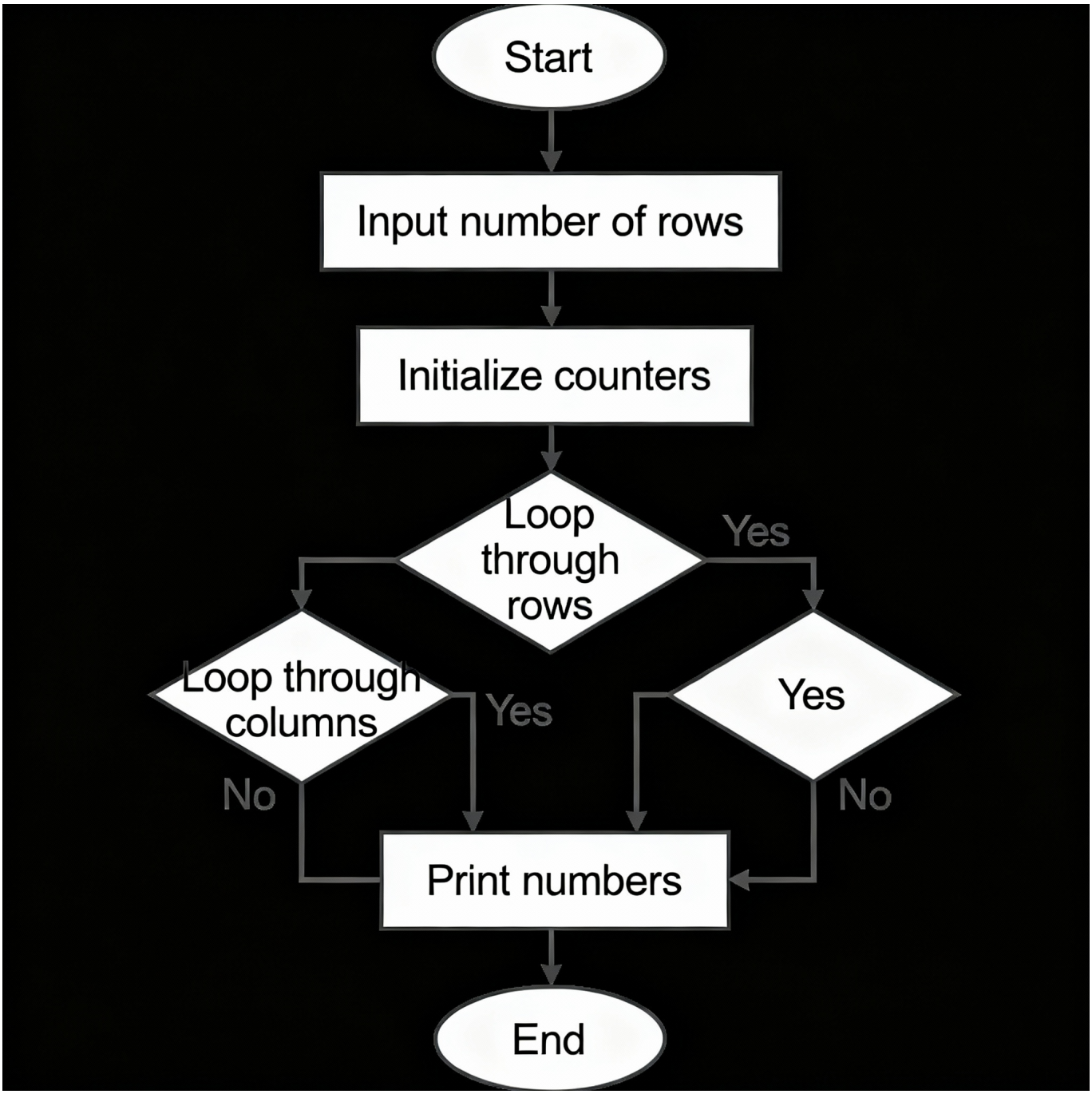
Step 4: For each row i, loop from j = 1 to i

Step 5: Print the current num and increment num by 1

Step 6: Move to the next line after each row

Step 7: Stop

Flowchart



Output

```
cd "c:\Users\dell\Documents\experiments in c\exp3_1\exp3_2\" ; if ($?) { gcc exp3_Q3a.c -o exp3_Q3a } ; if ($?) { .\exp3_Q3a }
1
2 3
4 5 6
PS C:\Users\dell\Documents\experiments in c\exp3_1\exp3_2>
```

b)

Algorithm

Step 1: Start

Step 2: Declare an array or variables to hold current and previous row values

Step 3: Loop through rows from 0 to 3 (4 rows total)

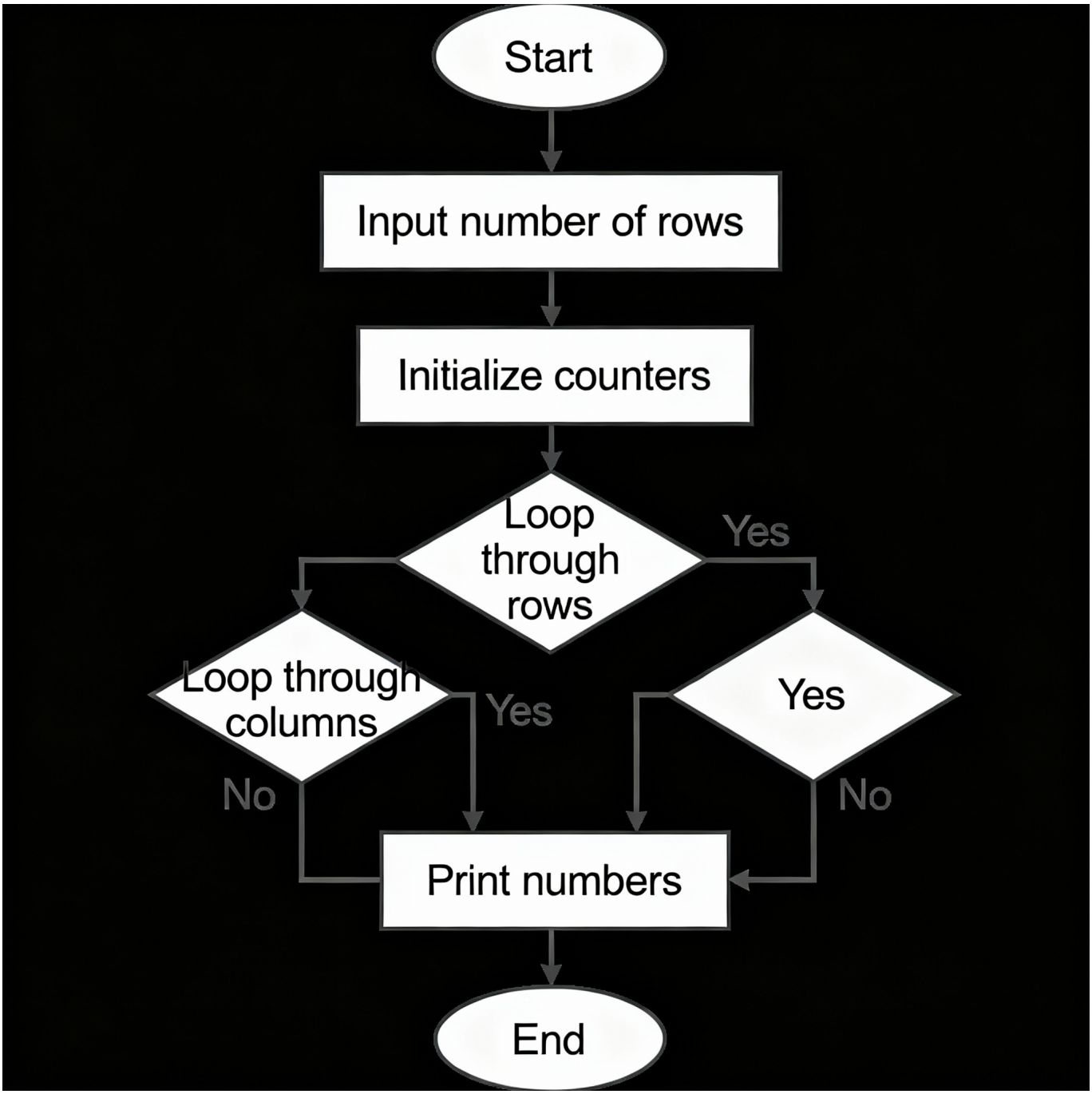
Step 4: For each row i, print the values using the formula for combinations or

```
    For j from 0 to i:
        If j == 0 or j == i, print 1
        Else print sum of previous row's j and j-1 elements
```

Step 5: Move to the next line after each row

Step 6: Stop

Flowchart



Output

```
PS C:\Users\dell\Documents\experiments in c>
cd "c:\Users\dell\Documents\experiments in c\exp3_1\exp3_2\" ; if ($?) { gcc exp3_Q3b.c -o exp3_Q3b } ; if ($?) { .\exp3_Q3b }
1
1 1
1 2 1
1 3 3 1
PS C:\Users\dell\Documents\experiments in c\exp3_1\exp3_2>
```

4: The population of a town is 100000. The population has increased steadily at the ratev of 10% per year for the last 10 years. Write a program to determine the population at the end of the year in the last decade.

Algorithm

Step 1: Start

Step 2: Include the header file <stdio.h>

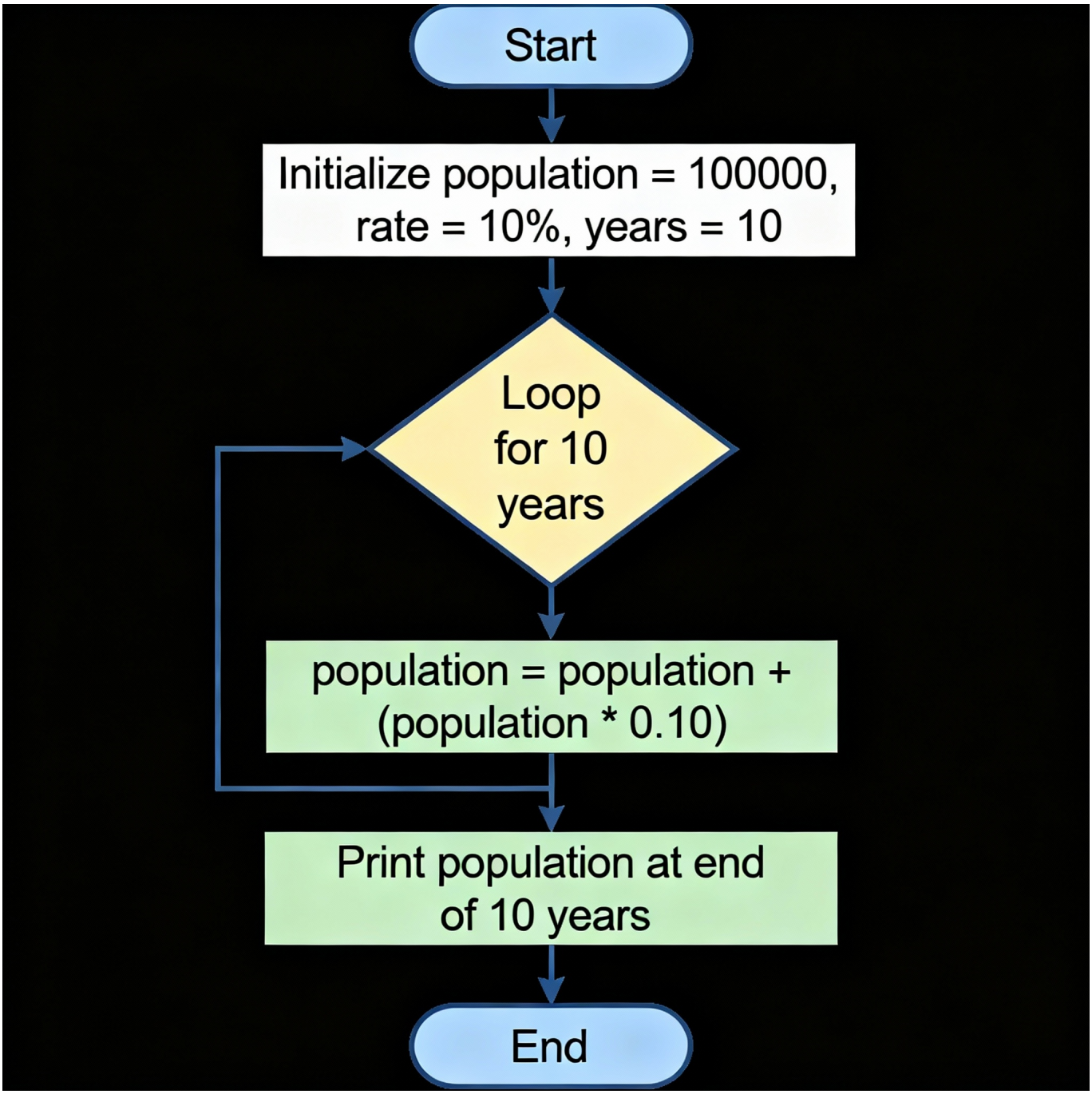
Step 3: Declare variables: - population (float) - year (int)

Step 4: Initialize population = 100000

Step 5: Loop through each year from 1 to 10 a) Compute new population: population = population + (population * 0.10) OR population *= 1.10 b) Display the population at the end of the current year

Step 6: Stop

Flowchart



Output

```
PS C:\Users\dell\Documents\experiments in c>
cd "c:\Users\dell\Documents\experiments in c\exp3_1\exp3_2\" ; if ($?) { gcc exp3_2Q4.c -o exp3_2Q4 } ; if ($?) { .\exp3_2Q4 }
Population at the end of year 1: 110000
Population at the end of year 2: 121000
Population at the end of year 3: 133100
Population at the end of year 4: 146410
Population at the end of year 5: 161051
Population at the end of year 6: 177156
Population at the end of year 7: 194872
Population at the end of year 8: 214359
Population at the end of year 9: 235795
Population at the end of year 10: 259374
PS C:\Users\dell\Documents\experiments in c\exp3_1\exp3_2>
```

5: Ramanujan Number is the smallest number that can be expressed as the sum of two cubes in two different ways.WAP to print all such numbers up to a resonable limit

Example of Ramanujan Number: 1729

12^3 + 1^3 and 10^3 + 9^3 for a number L=20(that is limit).

: Algorithm

Step 1: Start

Step 2: Include the header file <stdio.h>

Step 3: Declare variables: i, j, k, l → for looping limit → user input n1, n2 → for storing sums of cubes

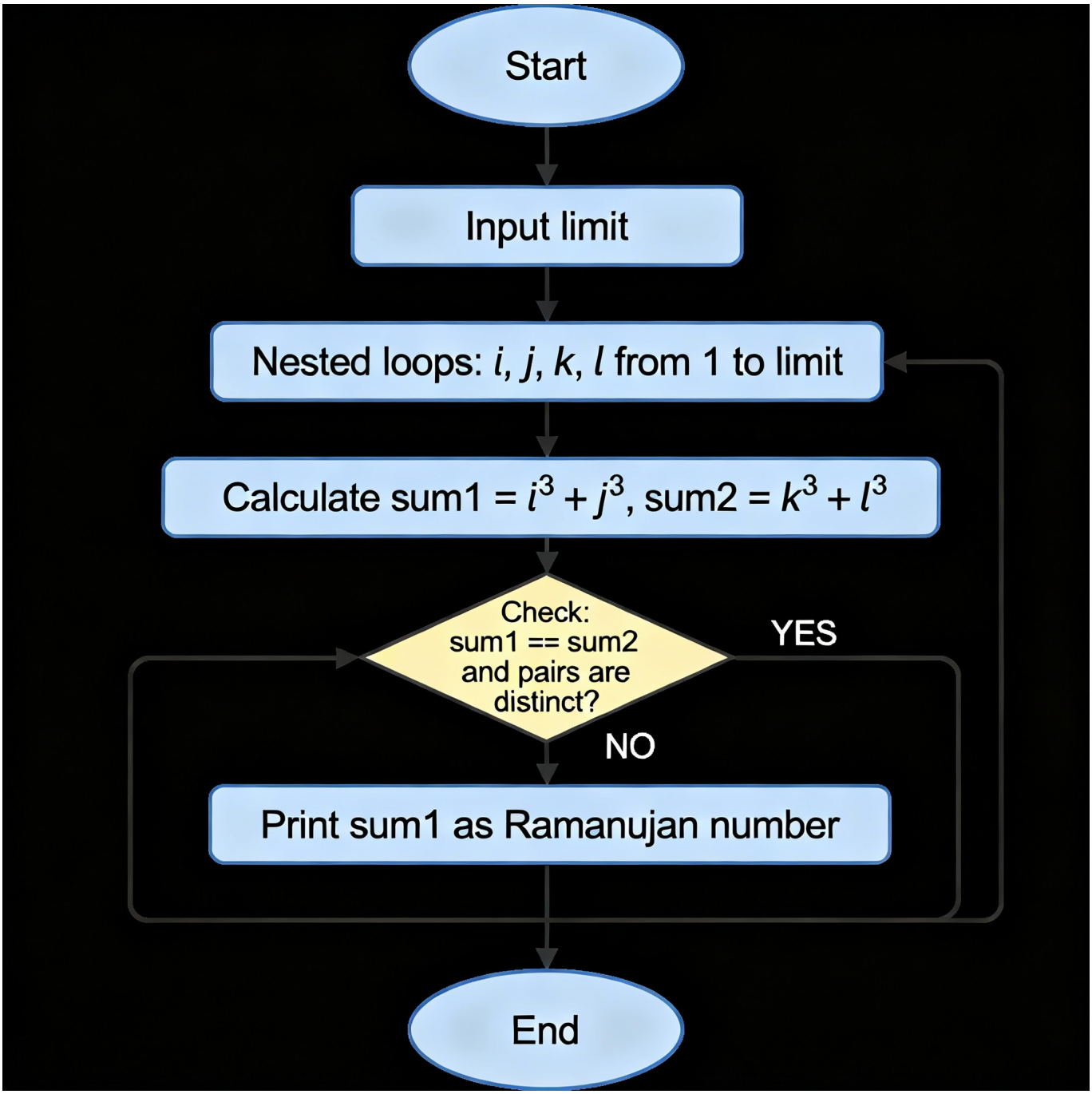
Step 4: Input limit from user

Step 5: Use four nested loops: For i = 1 to limit For j = i + 1 to limit Compute n1 = i^3 + j^3

```
            For k = i + 1 to limit
                For l = k + 1 to limit
                    Compute n2 = k^3 + l^3
                    If n1 == n2
                        Print the Ramanujan number and combinations
```

Step 6: Stop

: Flowchart



Output

```
cd "c:\Users\dell\Documents\experiments in c\exp3_1\exp3_2\" ; if ($?) { gcc exp3_2Q5.c -o exp3_2Q5 } ; if ($?) { .\exp3_2Q5 }
Enter the limit (e.g., 20): 50

Ramanujan Numbers up to 50:
1729 = 1^3 + 12^3 = 9^3 + 10^3
4104 = 2^3 + 16^3 = 9^3 + 15^3
13832 = 2^3 + 24^3 = 18^3 + 20^3
39312 = 2^3 + 34^3 = 15^3 + 33^3
46683 = 3^3 + 36^3 = 27^3 + 30^3
32832 = 4^3 + 32^3 = 18^3 + 30^3
110656 = 4^3 + 48^3 = 36^3 + 40^3
110808 = 6^3 + 48^3 = 27^3 + 45^3
40033 = 9^3 + 34^3 = 16^3 + 33^3
20683 = 10^3 + 27^3 = 19^3 + 24^3
65728 = 12^3 + 40^3 = 31^3 + 33^3
64232 = 17^3 + 39^3 = 26^3 + 36^3
PS C:\Users\dell\Documents\experiments in c\exp3_1\exp3_2>
```

EXPERIMENT 4: VARIABLE AND SCOPE OF VARIABLES

1: Declare a global variable outside all functions and use it inside various functions to understand its accessibility.

Algorithm

Step 1: Start

Step 2: Declare a global variable outside all functions (e.g., int count = 10)

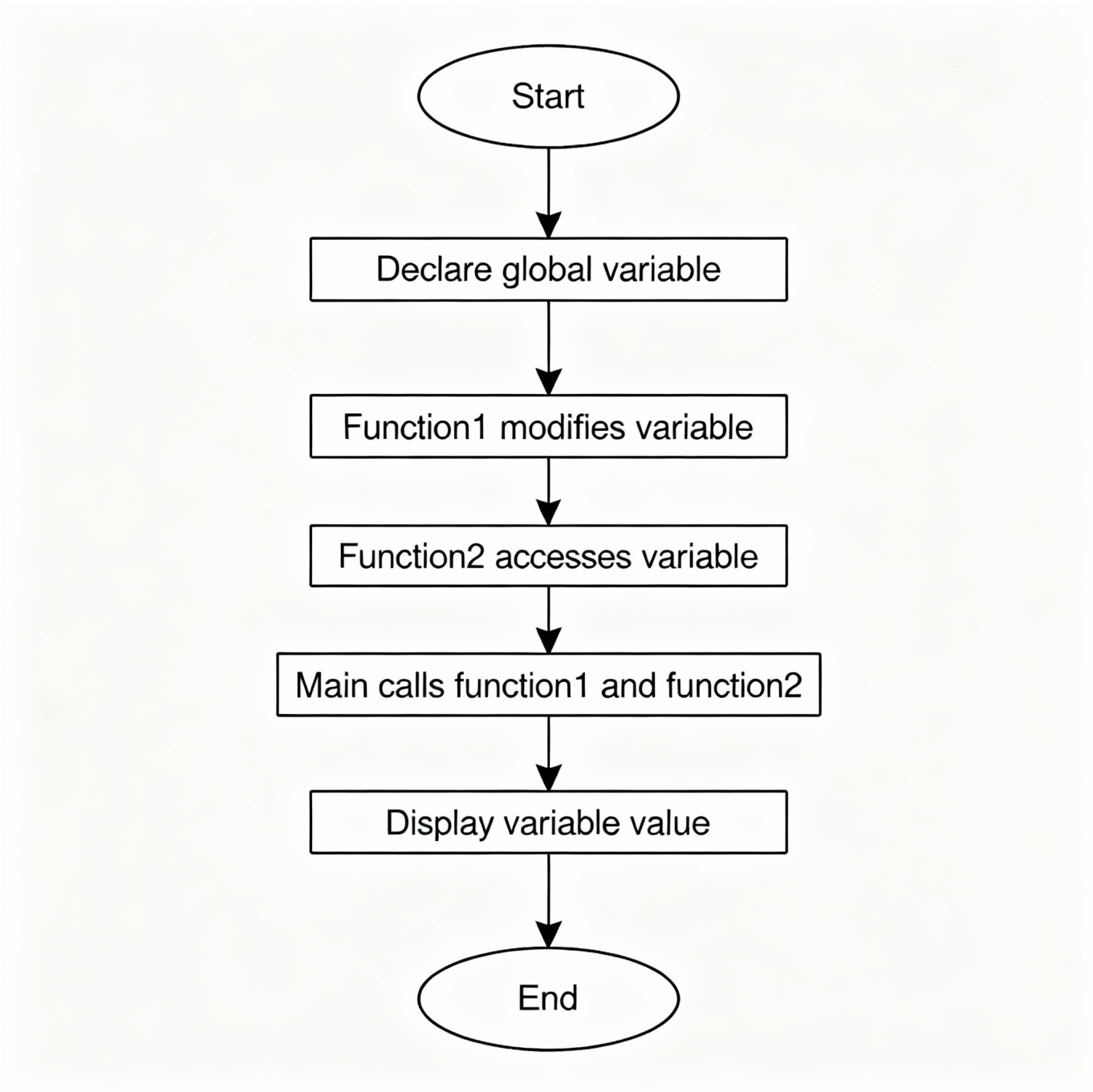
Step 3: Define a function display() that prints the value of the global variable

Step 4: Define another function modify() that changes the value of the global variable

Step 5: In main(): a) Print the initial value of the global variable b) Call display() to print the value c) Call modify() to change the value d) Call display() again to show the modified value

Step 6: Stop

Flowchart



Output

```
PS C:\Users\dell\Documents\experiments in c>
cd "c:\Users\dell\Documents\experiments in c\exp3_1\exp4\" ; if ($?) { gcc exp4Q1.c -o exp4Q1 } ; if ($?) { .\exp4Q1 }
Initial value of global variable (in main): 10
Value of global variable (in display): 10
Global variable modified (in modify): 15
Value of global variable (in display): 15
PS C:\Users\dell\Documents\experiments in c\exp3_1\exp4>
```

2: Declare a local variable inside a function and try to access it outside the function.Compare this with accessing the global variable from within the function.

Algorithm

Step 1: Start

Step 2: Declare a global variable outside all functions (e.g., int globalVar = 10)

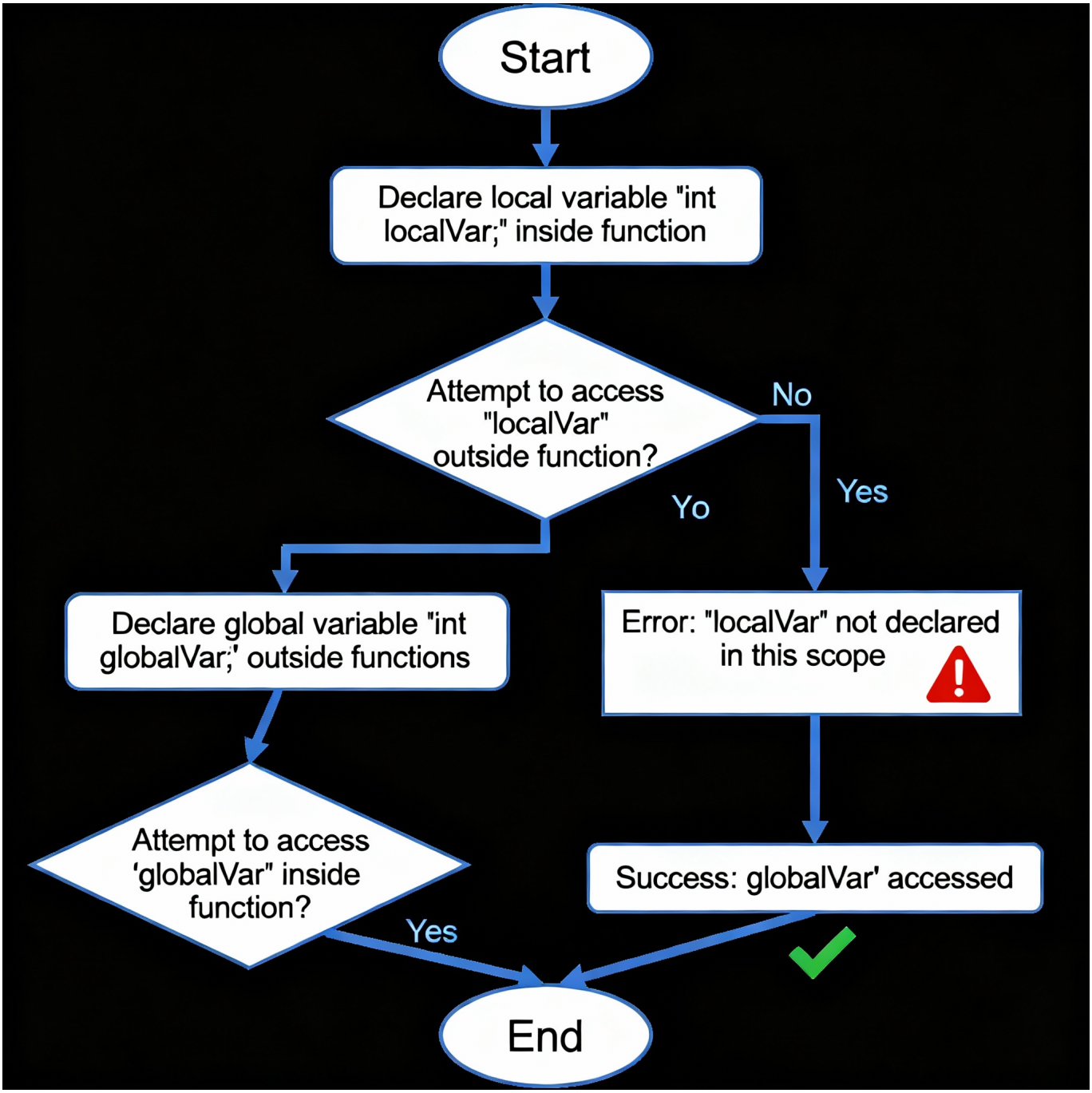
Step 3: Define a function testFunction(): a) Declare a local variable (e.g., int localVar = 5) b) Print both localVar and globalVar

Step 4: In main(): a) Print the global variable b) Call testFunction() c) Try to access localVar (which should give an error or be invalid)

Step 5: Observe that: - The global variable can be accessed anywhere. - The local variable is only accessible inside the function where it's declared.

Step 6: Stop

Flowchart



Output

```
PS C:\Users\dell\Documents\experiments in c>
cd "c:\Users\dell\Documents\experiments in c\exp3_1\exp4\" ; if ($?) { gcc exp4Q2.c -o exp4Q2 } ; if ($?) { .\exp4Q2 }
Inside main:
Global variable = 10
Inside testFunction:
Local variable = 5
Global variable = 10

Back in main, global variable still accessible = 10
PS C:\Users\dell\Documents\experiments in c\exp3_1\exp4>
```

3:Declare variables within different code blocks(enclosed by curly braces)and test their accessibility within and outside those blocks

Algorithm

Step 1: Start

Step 2: Include the header file <stdio.h>

Step 3: In main(), declare a variable 'a' and assign it a value (e.g., a = 10)

Step 4: Print the value of 'a'

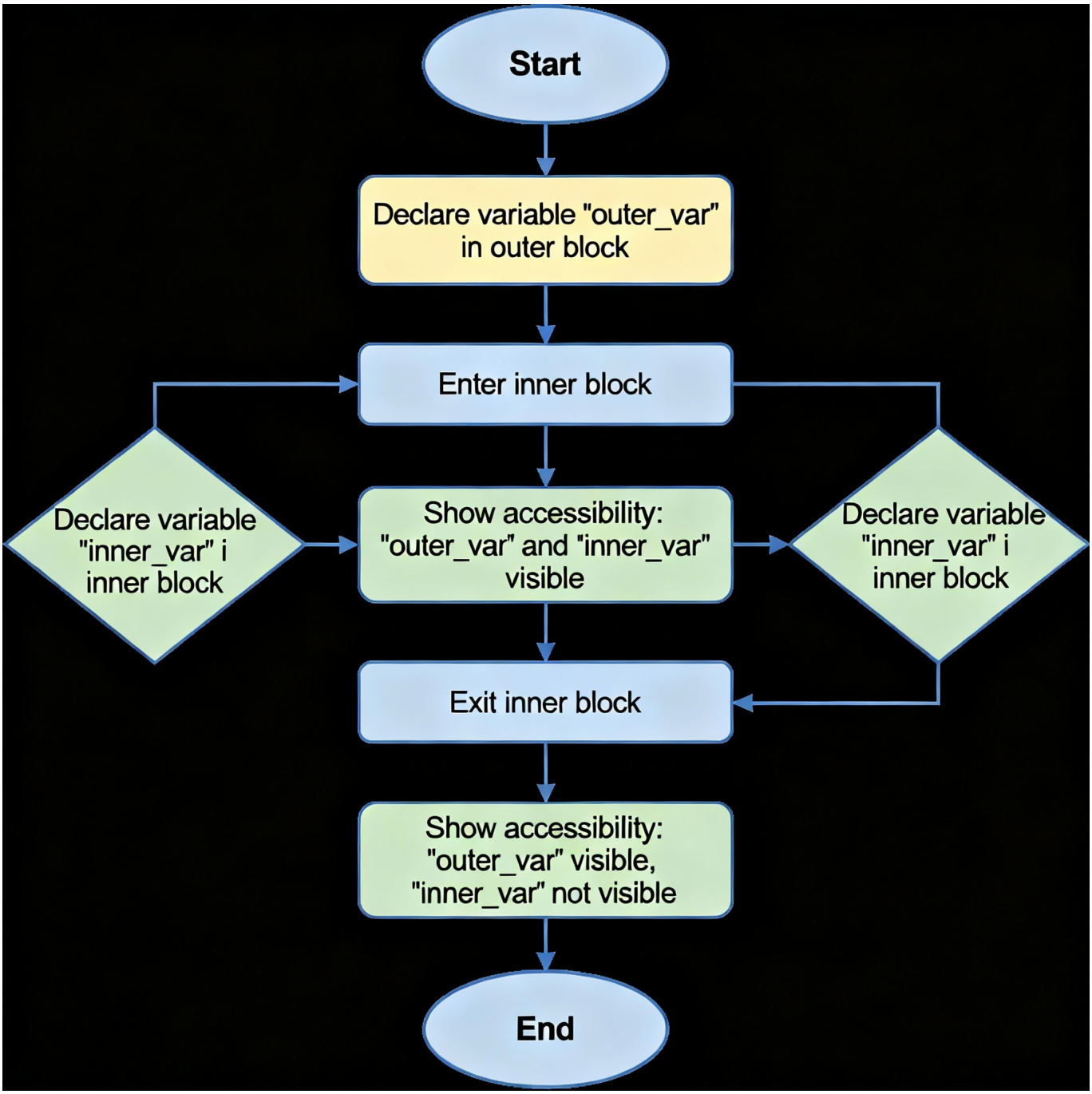
Step 5: Create a new block using { } a) Declare another variable 'b' inside this block b) Print 'a' (accessible from outer block) c) Print 'b' (accessible within this block)

Step 6: Outside the inner block: a) Print 'a' again (still accessible) b) Try to print 'b' (will cause an error if attempted, because 'b' is not in scope)

Step 7: End the program

Step 8: Stop

Flowchart



Output

```
PS C:\Users\dell\Documents\experiments in c>
cd "c:\Users\dell\Documents\experiments in c\exp3_1\exp4\" ; if ($?) { gcc exp4Q4.c -o exp4Q4 } ; if ($?) { .\exp4Q4 }
Inside main (outer block): a = 10
Inside inner block: a = 10, b = 20
Outside inner block: a = 10
PS C:\Users\dell\Documents\experiments in c\exp3_1\exp4>
```

4: Declare a static local variable inside a function.Observe how its value persists across function calls.

Algorithm

Step 1: Start

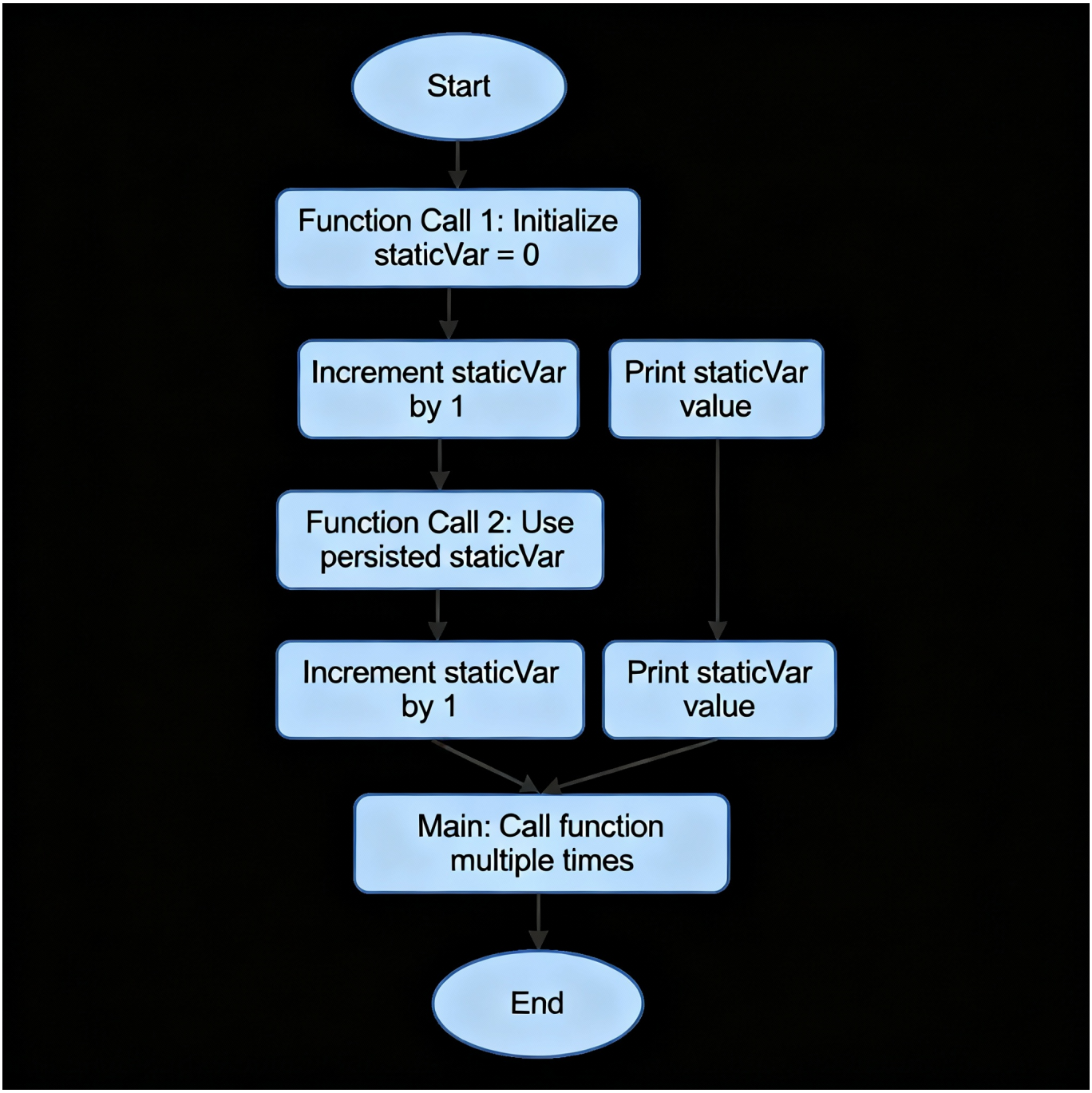
Step 2: Include the header file <stdio.h>

Step 3: Define a function `display()`: a) Declare a static local variable `count` and initialize it to 1 b) Print the value of `count` c) Increment `count` by 1

Step 4: In main(): a) Call `display()` multiple times (e.g., 3 times) b) Observe that the value of `count` persists between function calls

Step 5: Stop

Flowchart



Output

```
PS C:\Users\dell\Documents\experiments in c>
cd "c:\Users\dell\Documents\experiments in c\exp3_1\exp4\" ; if ($?) { gcc exp4Q4.c -o exp4Q4 } ; if ($?) { .\exp4Q4 }
First call:
Count = 1
Second call:
Count = 2
Third call:
Count = 3
PS C:\Users\dell\Documents\experiments in c\exp3_1\exp4>
```

EXPERIMENT 5: ARRAY

1:WAP to read a list of integers and store it in a single dimensional array.Write a C program to print the second largest integer in a list of integers.

Algorithm

Step 1: Start

Step 2: Include the header file <stdio.h>

Step 3: Declare variables: - arr[] → to store integers - n → number of elements - i → loop counter - first, second → to store the largest and second largest numbers

Step 4: Ask the user to enter the number of elements (n)

Step 5: Input all n integers into the array

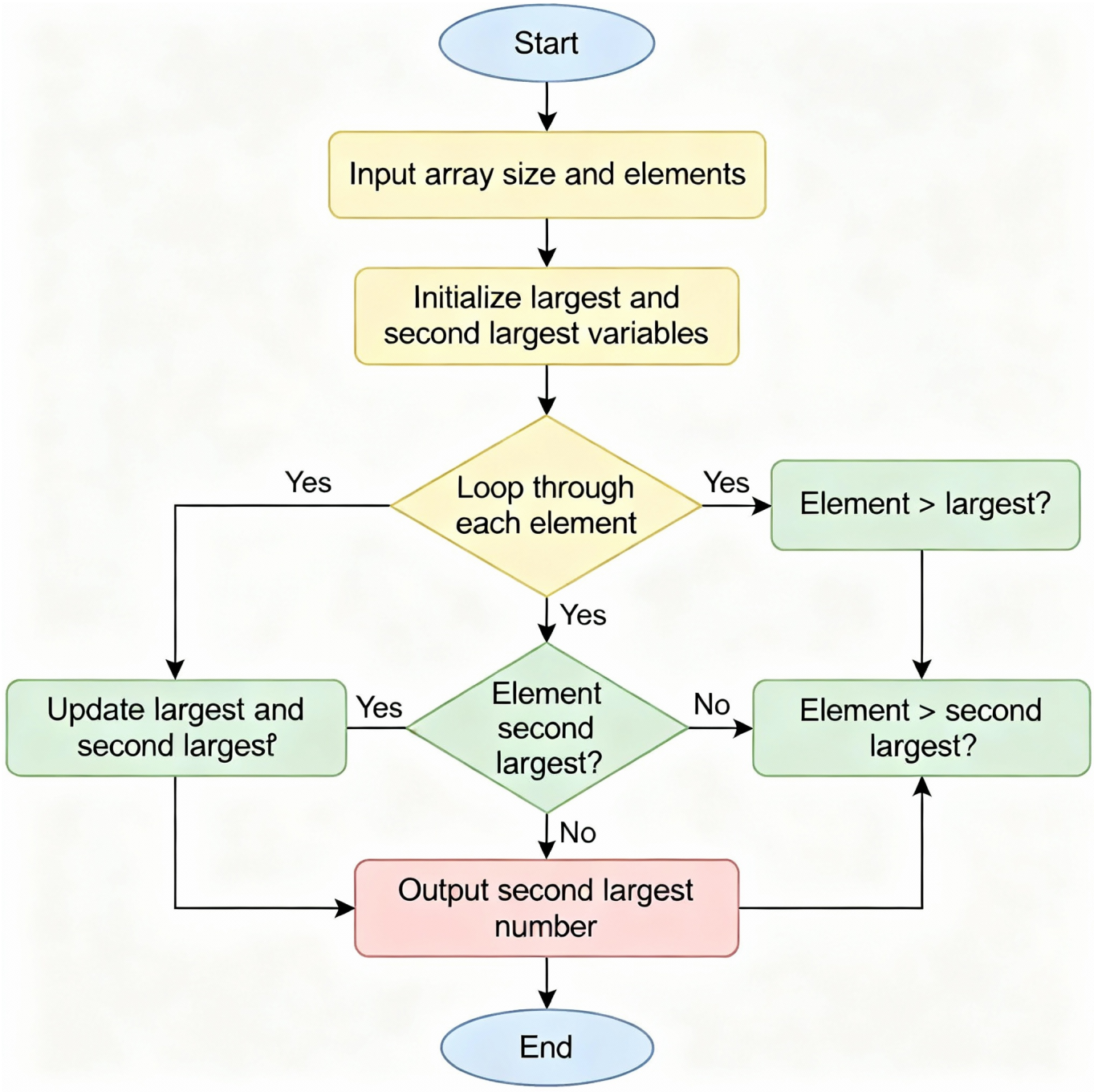
Step 6: Initialize: first = second = smallest possible value (e.g., -9999 or arr[0])

Step 7: Loop through the array: For i = 0 to n-1 - If arr[i] > first second = first first = arr[i] - Else if arr[i] > second and arr[i] != first second = arr[i]

Step 8: Print the value of 'second' as the second largest integer

Step 9: Stop

Flowchart



Output

```
PS C:\Users\dell\Documents\experiments in c>
cd "c:\Users\dell\Documents\experiments in c\exp3_1\exp4\exp5\" ; if ($?) { gcc exp5Q1.c -o exp5Q1 } ; if ($?) { .\exp5Q1 }
Enter number of elements: 20
Enter 20 integers:
1 2 3 4  5  6 7 8 9 10 11 12 13 14 15 16 17 18 19 20

The second largest number is: 19
PS C:\Users\dell\Documents\experiments in c\exp3_1\exp4\exp5>
```

2:WAP to read a list of integers and store it in a single dimensional array.Write a C program to count and display positive ,negative,odd and even numbers in a array.

Algorithm

Algorithm: Count and Display Positive, Negative, Odd, and Even Numbers

Step 1: Start

Step 2: Include the header file <stdio.h>

Step 3: Declare variables: - arr[] → array to store integers - n → number of elements - i → loop counter - pos, neg, odd, even → counters initialized to 0

Step 4: Ask the user to enter the number of elements (n)

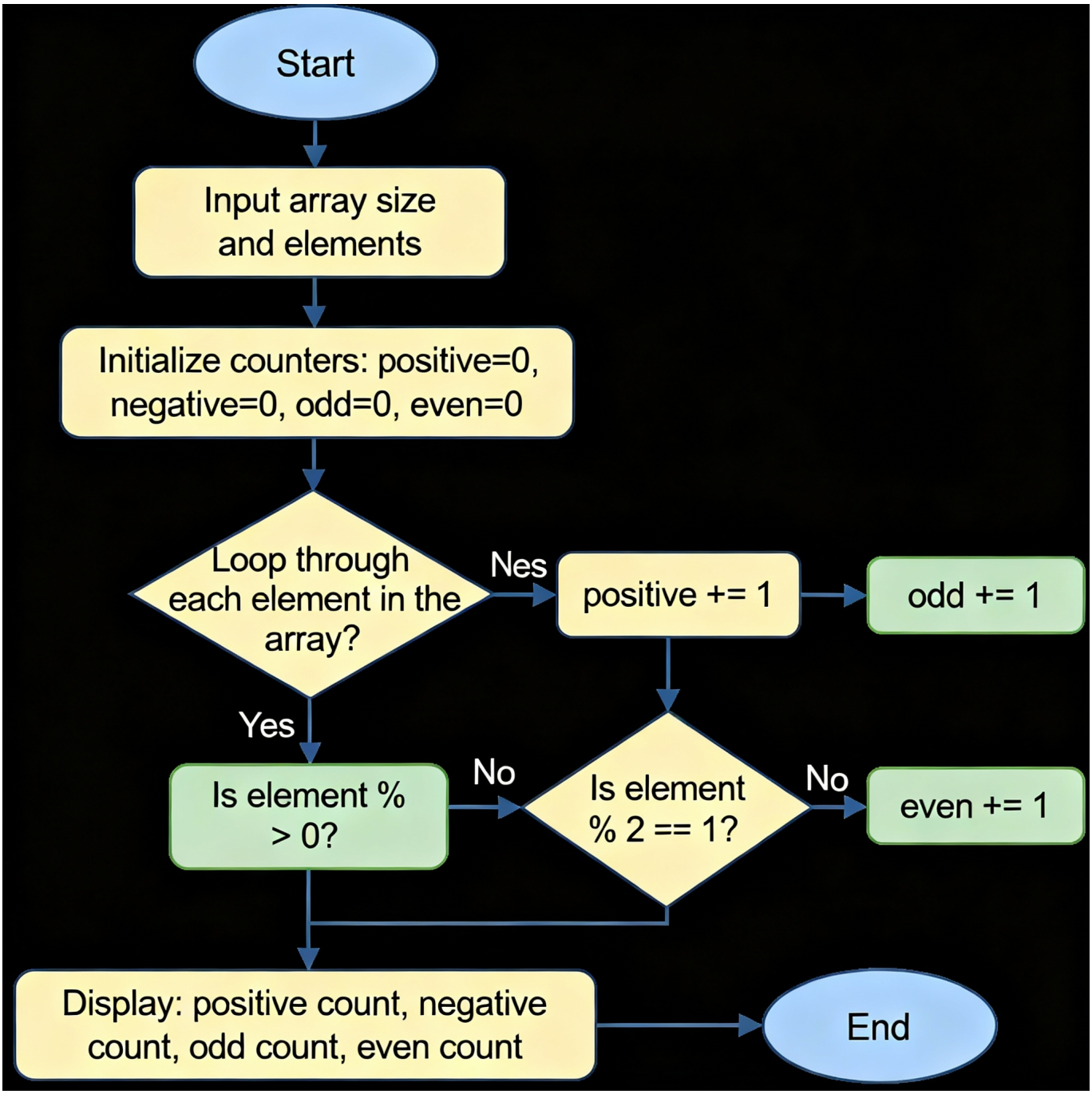
Step 5: Input n integers into the array using a loop

Step 6: Loop through the array from i = 0 to n-1: a) If arr[i] > 0, increment pos b) Else if arr[i] < 0, increment neg c) If arr[i] % 2 == 0, increment even Else increment odd

Step 7: Display the counts of: - Positive numbers - Negative numbers - Even numbers - Odd numbers

Step 8: Stop

Flowchart



Output

```
PS C:\Users\dell\Documents\experiments in c>
cd "c:\Users\dell\Documents\experiments in c\exp3_1\exp4\exp5\" ; if ($?) { gcc exp5Q2.c -o exp5Q2 } ; if ($?) { .\exp5Q2 }
Enter number of elements: 4
Enter 4 integers:
1 2 3 4

Count of Positive numbers: 4
Count of Negative numbers: 0
Count of Even numbers: 2
Count of Odd numbers: 2
PS C:\Users\dell\Documents\experiments in c\exp3_1\exp4\exp5>
```

3:WAP to read a list of integers and store it in a single dimensional array. Write a C program to find the frequency of a particular numbers in list of integers

Algorithm

Algorithm: Find Frequency of a Given Number in an Array

Step 1: Start

Step 2: Include the header file <stdio.h>

Step 3: Declare variables: - arr[] → array to store integers - n → number of elements - i → loop counter - num → number whose frequency is to be found - count → variable to count frequency (initialize to 0)

Step 4: Ask the user to enter the number of elements (n)

Step 5: Input n integers into the array using a loop

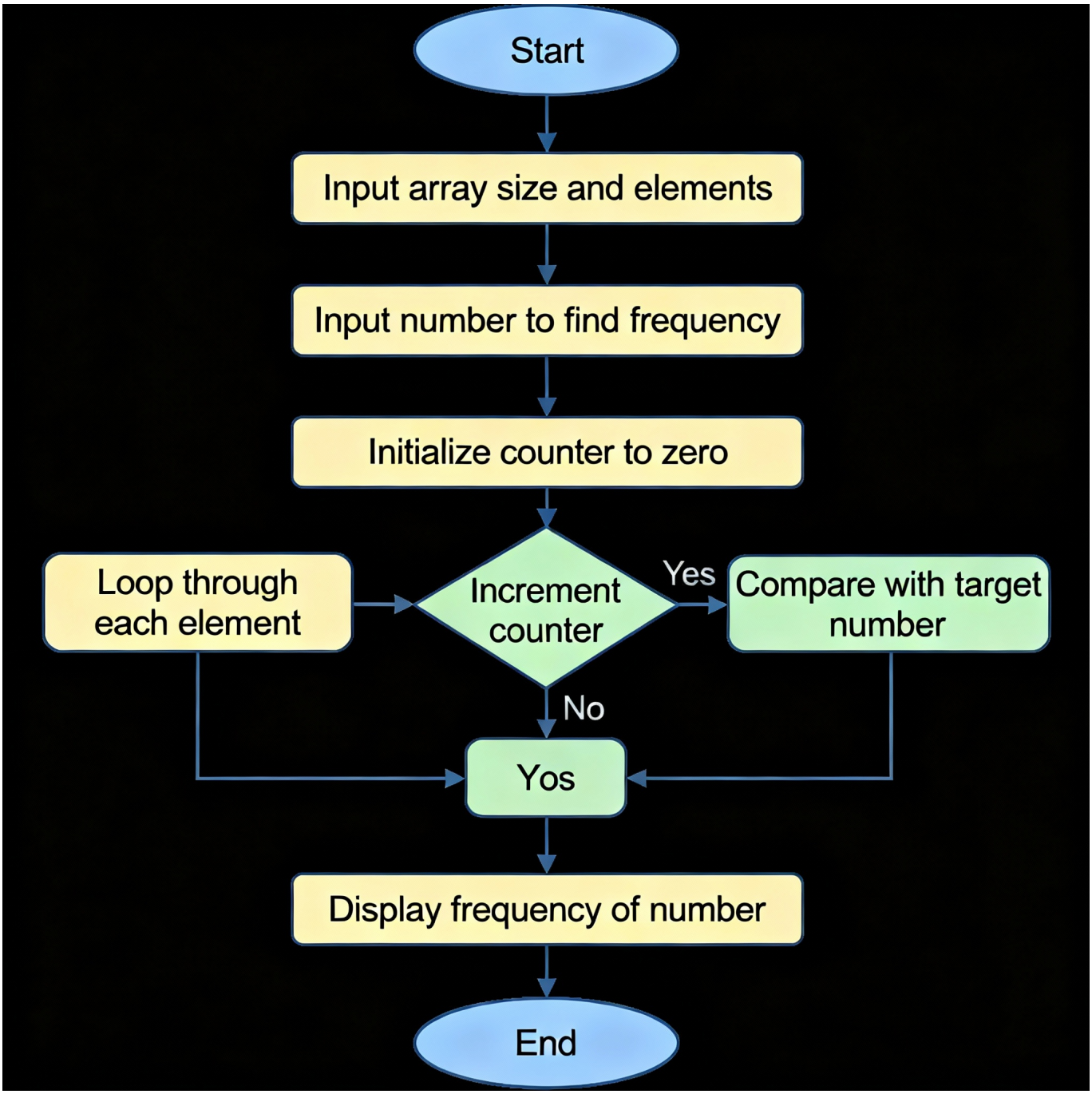
Step 6: Ask the user to enter the number (num) whose frequency is to be found

Step 7: Loop through the array from i = 0 to n - 1: - If arr[i] == num increment count by 1

Step 8: Display the frequency (count) of the number

Step 9: Stop

Flowchart



Output

```
PS C:\Users\dell\Documents\experiments in c>
cd "c:\Users\dell\Documents\experiments in c\exp3_1\exp4\exp5\" ; if ($?) { gcc exp5Q3.c -o exp5Q3 } ; if ($?) { .\exp5Q3 }
Enter number of elements: 5
Enter 5 integers:
1 20 37 34 7
Enter the number to find its frequency: 34

The frequency of 34 is: 1
PS C:\Users\dell\Documents\experiments in c\exp3_1\exp4\exp5>
```

4: WAP a program that reads two matrices A(m X n)and B(p X q)and computes the product A and B . Read matrix A and B in a row major order respectively.Print both the input matrices and resultant matrix with suitable headings and output should be in matrix form only.

Program must check the compatibility of orders of the matrices for multiplication .Report appropriate message in case of incompatibility.

Algorithm

Algorithm: Matrix Multiplication with Compatibility Check

Step 1: Start

Step 2: Include the header file <stdio.h>

Step 3: Declare matrices: A[m][n], B[p][q], C[m][q] and variables m, n, p, q, i, j, k

Step 4: Input order of matrix A (m × n) Input elements of matrix A (row-wise)

Step 5: Input order of matrix B (p × q) Input elements of matrix B (row-wise)

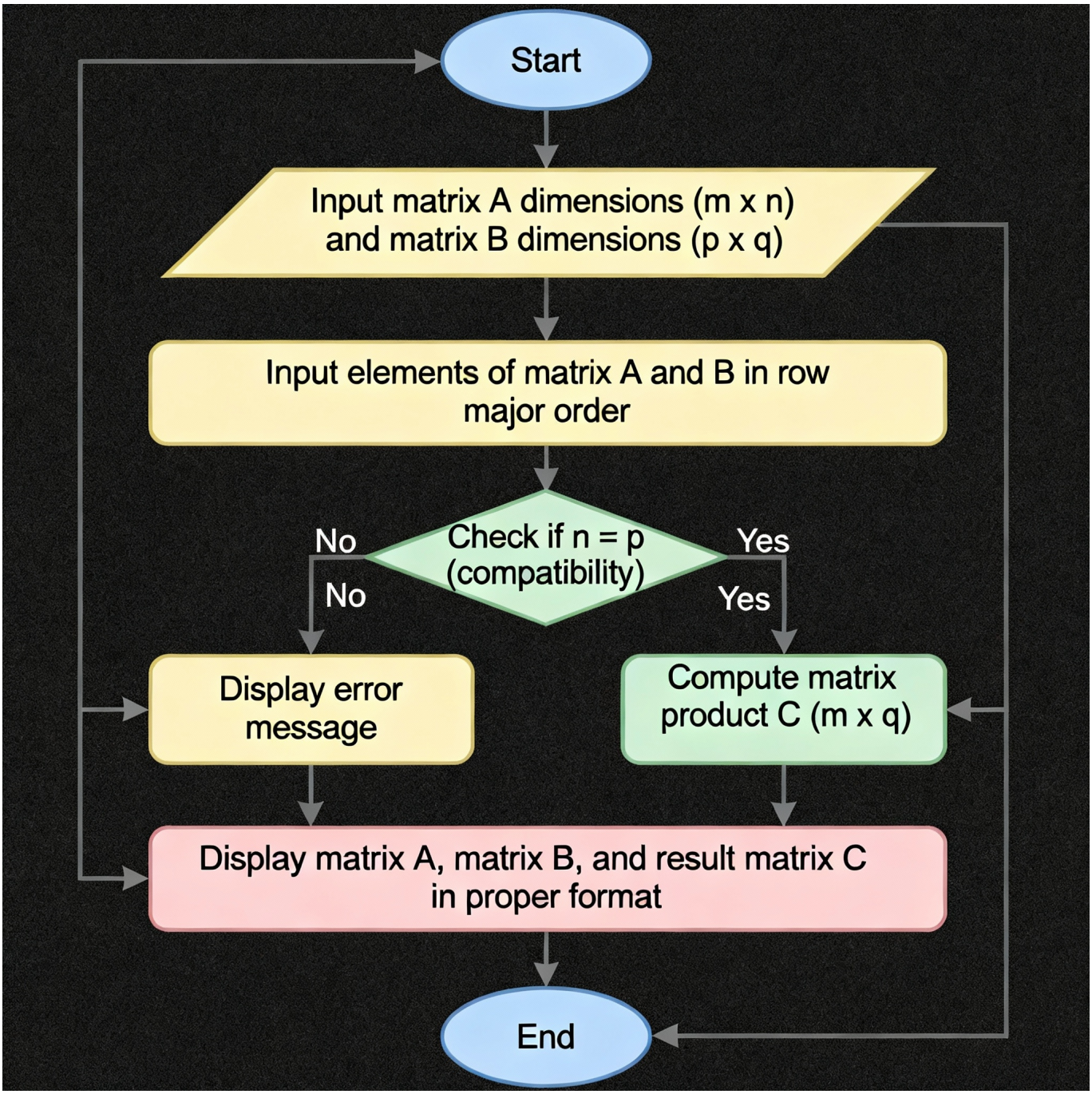
Step 6: Check compatibility for multiplication: If n != p Print "Matrix multiplication not possible!" Stop the program

Step 7: Else Initialize all elements of C to 0 For i = 0 to m-1 For j = 0 to q-1 For k = 0 to n-1 C[i][j] = C[i][j] + A[i][k] * B[k][j]

Step 8: Print Matrix A Step 9: Print Matrix B Step 10: Print Resultant Matrix C

Step 11: Stop

Flowchart



Output

```
PS C:\Users\dell\Documents\experiments in c>
cd "c:\Users\dell\Documents\experiments in c\exp3_1\exp4\exp5\" ; if ($?) { gcc exp5Q4.c -o exp5Q4 } ; if ($?) { .\exp5Q4 }
Enter rows and cols of Matrix A: 2
4
Enter Matrix A elements:
1 2
3 4
4 5
2 3
Enter rows and cols of Matrix B: 2 2
Enter Matrix B elements:
2 3
3 4

Matrix multiplication not possible!
PS C:\Users\dell\Documents\experiments in c\exp3_1\exp4\exp5>
```