**SRI CHANDRASEKHARENDRA SARASWATHI VISWA MAHAVIDYALAYA**

**(UNIVERSITY ESTABLISHED under section 3 of UGC Act 1956)**

**ENATHUR,** **KANCHIPURAM – 631 561**



**PROBLEM SOLVING TECHNIQUES LAB**

**LABORATORY RECORD**

**Name :** T. ROHITH

**Reg. No :** 112534041

**Class :** I YEAR BCA

**Subject :** UCAF251T50 - PST IN C LAB

**SRI CHANDRASEKHARENDRA SARASWATHI**

**VISWA MAHAVIDYALAYA**

**(University Established under section 3 of UGC Act 1956)**

****

**BONAFIDE CERTIFICATE**

**This is to Certify that this is the bonafide record of work done by Mr. T. ROHITH**, **with Reg. No 112534041 of I Year BCA in the Problem Solving Techniques Lab during the year 2025.**

**Staff-in-charge** **Head of the Department**

**Submitted for the Practical Examination held on \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Internal Examiner** **External Examiner.**

**INDEX**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **S.No.** | **Date** | **Title** | **Page No.** | **Signature** |
| 1 | 24 Jul 2025 | [CELSIUS TO FAHRENHEIT](#CELSIUS) | 1 |  |
| 2 | 31 Jul 2025 | [FIBONNACI SERIES](#SERIES) | 3 |  |
| 3 | 07 Aug 2025 | [SQUARE AND CUBE OF NUMBERS](#squarecube) | 5 |  |
| 4 | 14 Aug 2025 | [GENERATE ODD NUMBERS](#Odd) | 7 |  |
| 5 | 21 Aug 2025 | [GENERATE GRADE](#grade) | 9 |  |
| 6 | 28 Aug 2025 | [GENERATE TOWER](#tower) | 11 |  |
| 7 | 04 Sep 2025 | PALINDROME OR NOT | 14 |  |
| 8 | 25 Sep 2025 | STRING HANDLING | 17 |  |
| 9 | 25 Sep 2025 | ARRAY SORTING | 20 |  |
| 10 | 09 Oct 2025 | [FACTORIAL](#factorial) | 23 |  |
| 11 | 09 Oct 2025 | SWAPPING OF NUMBER | 25 |  |
| 12 | 23 Oct 2025 | USING STRUCTURES | 27 |  |
| 13 | 23 Oct 2025 | FILE PROGRAMMING | 29 |  |

|  |  |  |
| --- | --- | --- |
| **1** | **CELSIUS TO FAHRENHEIT** | **DATE: 24 Jul 2025** |

**AIM:**

**Write a C program to Convert temperature from Celsius to Fahrenheit and vice versa.**

**ALGORITHM:**

**Step 1:** Start the program.

**Step 2:** Display the menu options:

* Option 1 → Convert Celsius to Fahrenheit
* Option 2 → Convert Fahrenheit to Celsius

**Step 3:** Read the user’s choice.

**Step 4: if** the choice = 1 goto step 5, else goto step 9

**Step 5:** Read the Celsius value.

**Step 6:** Convert Celsius to Fahrenheit using the formula: Fahrenheit=(Celsius×9/5) +32

**Step 7:** display the converted Fahrenheit value.

**Step 8:** Goto step 15.

**Step 9: if** the choice is 2: goto step 11. Else goto step 14.

**Step 11:** Read the Fahrenheit value.

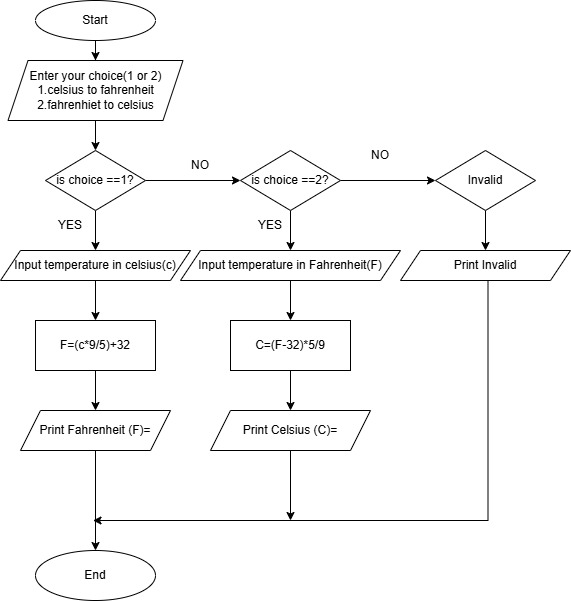
**Step 12:** Convert it to Celsius using the formula: Celsius=(Fahrenheit−32) ×5/9.

**Step 13:**Display the converted Celsius value.

**Step 14:** Display an error message: “Invalid choice! Please run the program again and choose 1 or 2.”

**Step 15:** End the program.

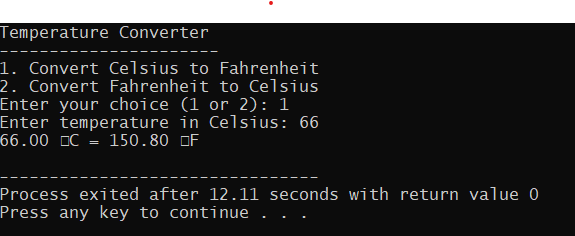
**FLOW CHART:**

****

**SOURCE CODE :**

<https://github.com/murugamuruga123/clab/blob/main/CELSIUS%20TO%20FAHRENHEIT.c>

**OUTPUT:**

****

**RESULT :**

Thus the program is compiled and executed successfully with verified output.

|  |  |  |
| --- | --- | --- |
| **2** | **FIBONACCI SERIES** | **DATE:** 31 Jul 2025 |

**AIM :**

**Write a C program to generate a Fibonacci series.**

**ALGORITHM :**

**Step 1:** Start the program.

**Step 2:** Declare variables:  
  n (number of terms),  
  t1 = 0 (first term),  
  t2 = 1 (second term),  
  next Term (to store next Fibonacci number),  
  i (loop counter).

**Step 3:** read the number of terms n.

**Step 4:** i=3

**Step 5**: Calculate next Term = t1 + t2.

**Step 6**: Print next Term.

**Step 7:** t1 = t2,

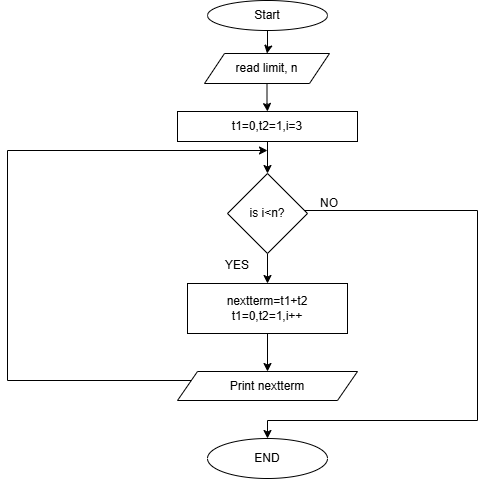
t2 = next Term.

I=i+1

**Step 8**: if i<n repeat step 5, else continue

**step 9:** End the program.

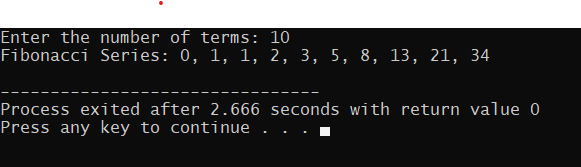
**FLOW CHART:**



**SOURCE CODE:**

[**https://github.com/murugamuruga123/clab/blob/main/fibonnaci%20series.c**](https://github.com/murugamuruga123/clab/blob/main/fibonnaci%20series.c)

**OUTPUT:**

****

**RESULT :**

Thus the program is compiled and executed successfully with verified output.

|  |  |  |
| --- | --- | --- |
| **3** | **SQUARE AND CUBE OF NUMBERS** | **DATE:07 Aug 2025** |

**AIM :**

Write a C program to calculate the square and cube of 1 to n numbers

**ALGORITHM :**

**Step 1:** Start

**Step 2:** Declare integer variables limit(n) and counter (i)

**Step 3:** Read the value n

**Step 4:** Use a **for loop** that runs from i = 1

**Step 5:** Calculate square = i \* i

**Step 6:** Calculate cube = i \* i \* i

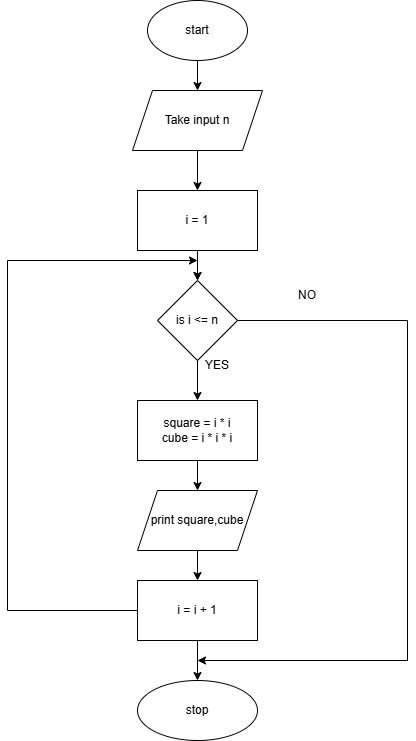
**Step 7:** Display i, square, and cube

**Step 8:** i = i + 1

**Step 9:** If i < n go to step 5 else go to step 10

**Step 10:** stop

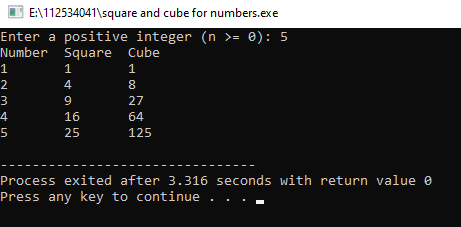
**FLOWCHART:**



**SOURCE CODE :**

<https://github.com/Rohith-Tech-king/c-lab/blob/main/square%20and%20cube%20for%20numbers.c>

**OUTPUT:**



**RESULT :**

Thus the program is compiled and executed successfully with verified output.

|  |  |  |
| --- | --- | --- |
| **4** | **GENERATE ODD NUMBERS** | **DATE:14 Aug 2025** |

**AIM :**

Write a C program to display odd numbers from 1 to n

**ALGORITHM :**

**Step 1:** Start

**Step 2:** Declare integer variables limit (n) and counter (i)

**Step 3:** Read the value of n

**Step 4:** Use a **for loop** that runs from i = 1

**Step 5:** if i % 2! = 0 (i.e., i is odd) go to step 6, else go to step 7

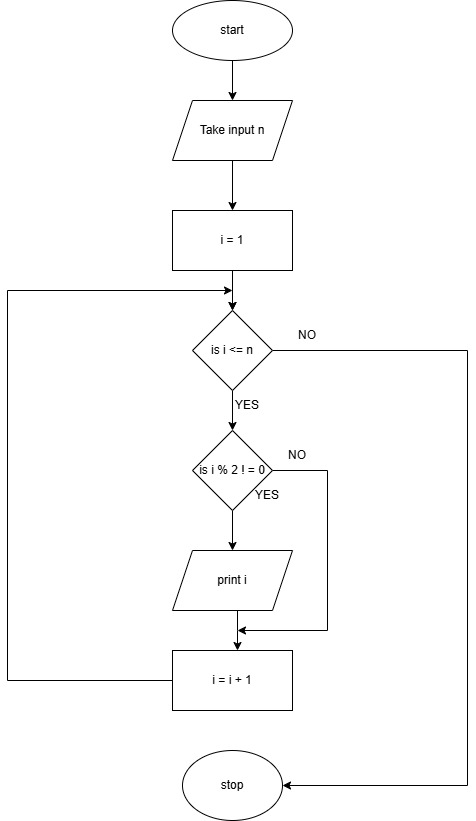
**Step 6:** Print i

**Step 7:** i=i+1

**Step 8:** Go to Step 5

**Step 9:** Stop

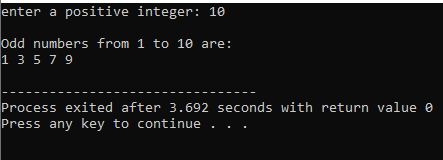
**FLOWCHART:**



**SOURCE CODE:**

[**https://github.com/Rohith-Tech-king/c-lab/blob/main/odd%20num.c**](https://github.com/Rohith-Tech-king/c-lab/blob/main/odd%20num.c)

**OUTPUT:**

****

**RESULT :**

Thus the program is compiled and executed successfully with verified output**.**

|  |  |  |
| --- | --- | --- |
| **5** | **GENERATE GRADE** | **DATE:21 Aug 2025** |

**AIM:**

To write a C program that reads marks (from 0 to 100) and prints the corresponding grade using a switch-case statement based on this grading:

**ALGORITHM:**

**Step 1:** start

**Step 2**: Declare a variable(n) to store marks

**Step 3:** enter marks

**Step 4:** Check if 0 <= marks <= 100 continue else go to step 8

**Step 5:** calculate range=marks/10

**Step 6:** Use switch case to decide the grade:

If range (80-100) grade= A

If range (60-79) grade =B

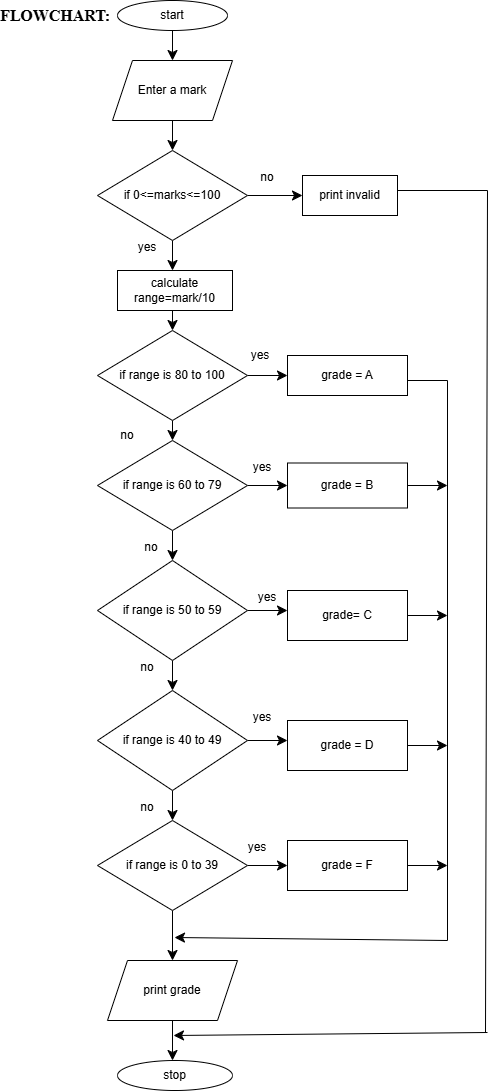
If range (50-59) grade =C

If range (40-49) grade =D

If range (0-39) grade=F

**Step 7:** Print the grade.

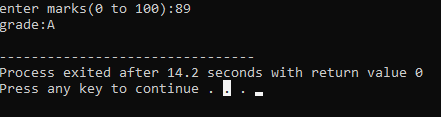
**Step 8:** End



**SOURCE CODE:**

[**https://github.com/sruthisruthi18112007/c-lab/blob/main/grade.c**](https://github.com/sruthisruthi18112007/c-lab/blob/main/grade.c)

**OUTPUT:**

****

**RESULT:**

Thus the program is compiled and executed successfully with verified output.

|  |  |  |
| --- | --- | --- |
| **6** | **GENERATE TOWER** | **DATE:28 Aug 2025** |

**AIM:**

To write a C program that prints a right angled triangle pattern using a given character and number of lines.

**ALGORITHM:**

Step 1:start.

Step 2:declare variables: Limit (n), counters (i, j).

Step 3**:** enter a character to print**,** ch.

Step 4: enter a limit n.

Step 5:i=1.

Step 6: j=1.

Step 7: Print character.

Step 8**:** j++.

Step 9: If j<i, goto step 7, else continue

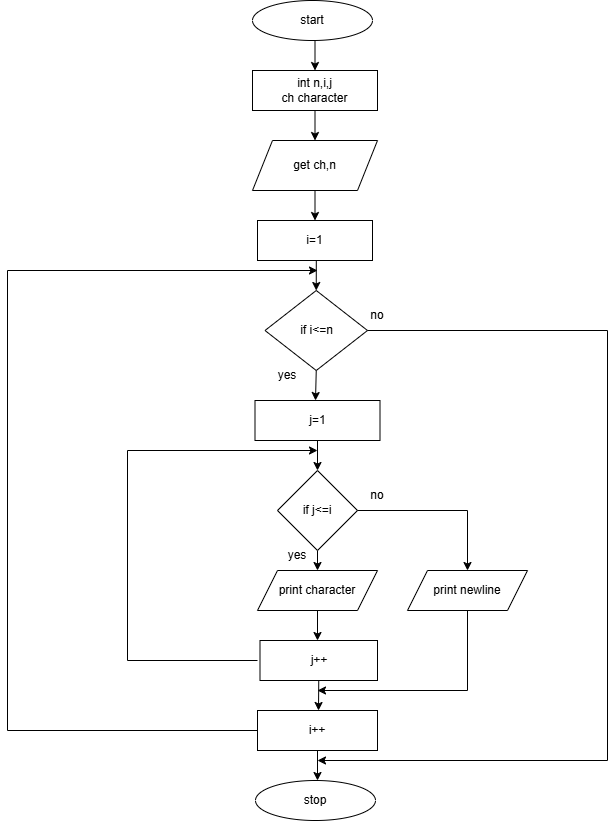
Step 10: i++.

Step 11: print ‘\n’

Step 12: if i<n, goto step 6, else continue

Step 13: stop.

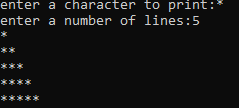
**FLOWCHART:**

****

**SOURCE CODE:**

[**https://github.com/sruthisruthi18112007/c-lab/blob/main/tower.c**](https://github.com/sruthisruthi18112007/c-lab/blob/main/tower.c)

**OUTPUT:**

****

**RESULT:**

Thus the program is compiled and executed successfully with verified output.

|  |  |  |
| --- | --- | --- |
| **10** | **FACTORIAL** | **DATE:09 Oct 2025** |

**AIM:**

Write a C program to find the factorial of an integer using recursive function

**ALGORITHM:**

**Step 1:** Start

**Step 2:** Read limit, n

**Step 3:** If n< 0 print “Give positive numbers” and go to step 5 else continue

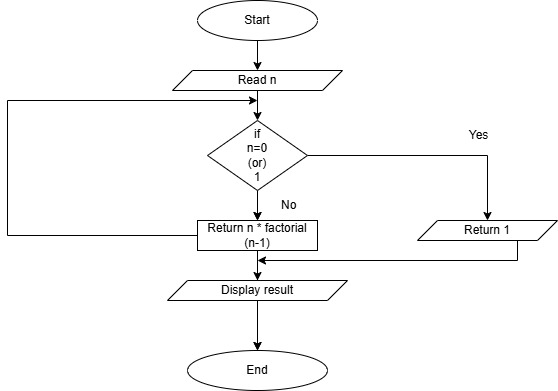
**Step 4:** Print factorial (n)

**Step 5:** Stop

Algorithm for factorial (int n)

1. If n=0 or n=1, return 1else, return n\*factorial (n-1)

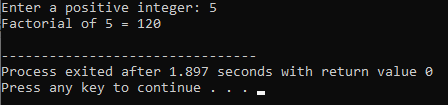
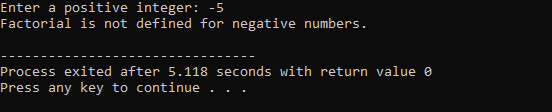
**FLOWCHART:**



**SOURCE CODE:**

<https://github.com/vaibav77/c-lab-vaibav/blob/main/factorial.c>

**OUTPUT:**

** **

**RESULT:**

Thus the program is compiled and executed successfully with verified output.