C / C++ Programming

Lab # 02
Introduction to C & Debugger tools

Name	*Muddassir Ali Siddiqui*
Instructor	*Miss Hira Sohail*
Date	*10th July 2025*

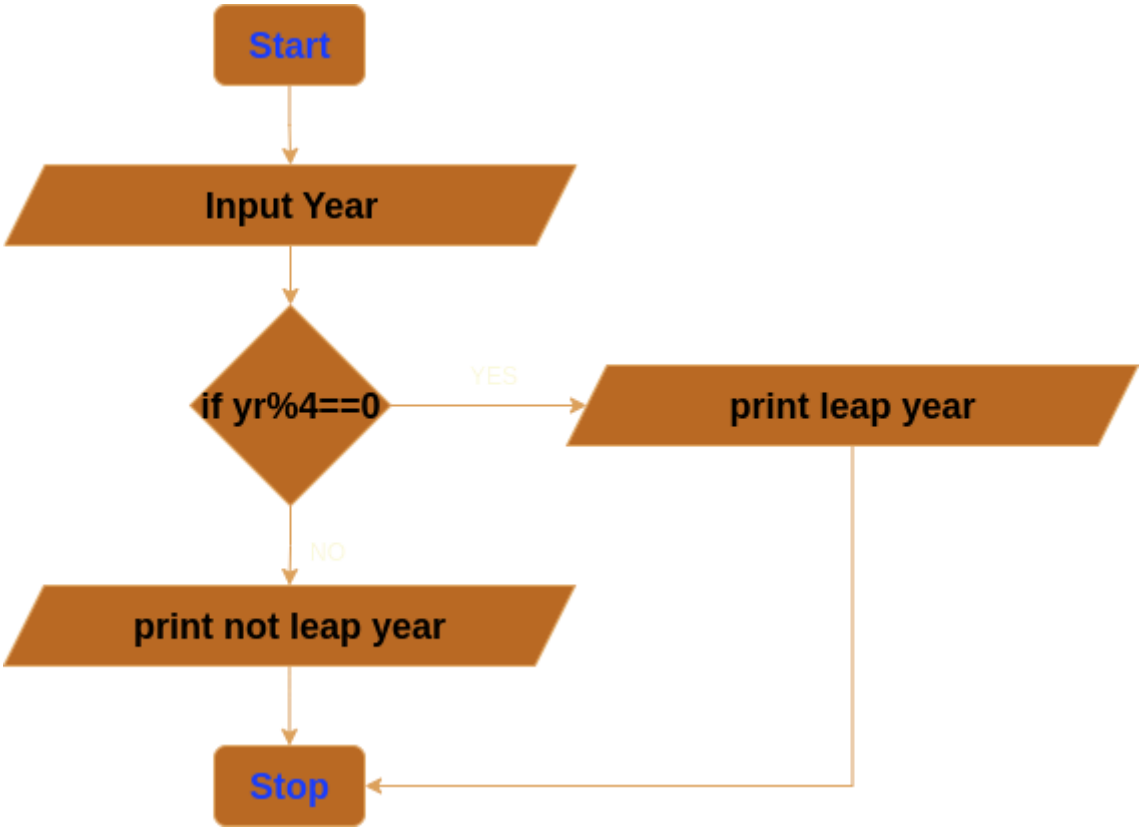
1. In-Lab Tasks: (*Write your lab task & screenshots here*)

i. Task 1:

It takes Earth approximately 365 days and 6 hours to orbit the Sun. It takes Earth approximately 24 hours — 1 day — to rotate on its axis. So, our year is not an exact number of days.

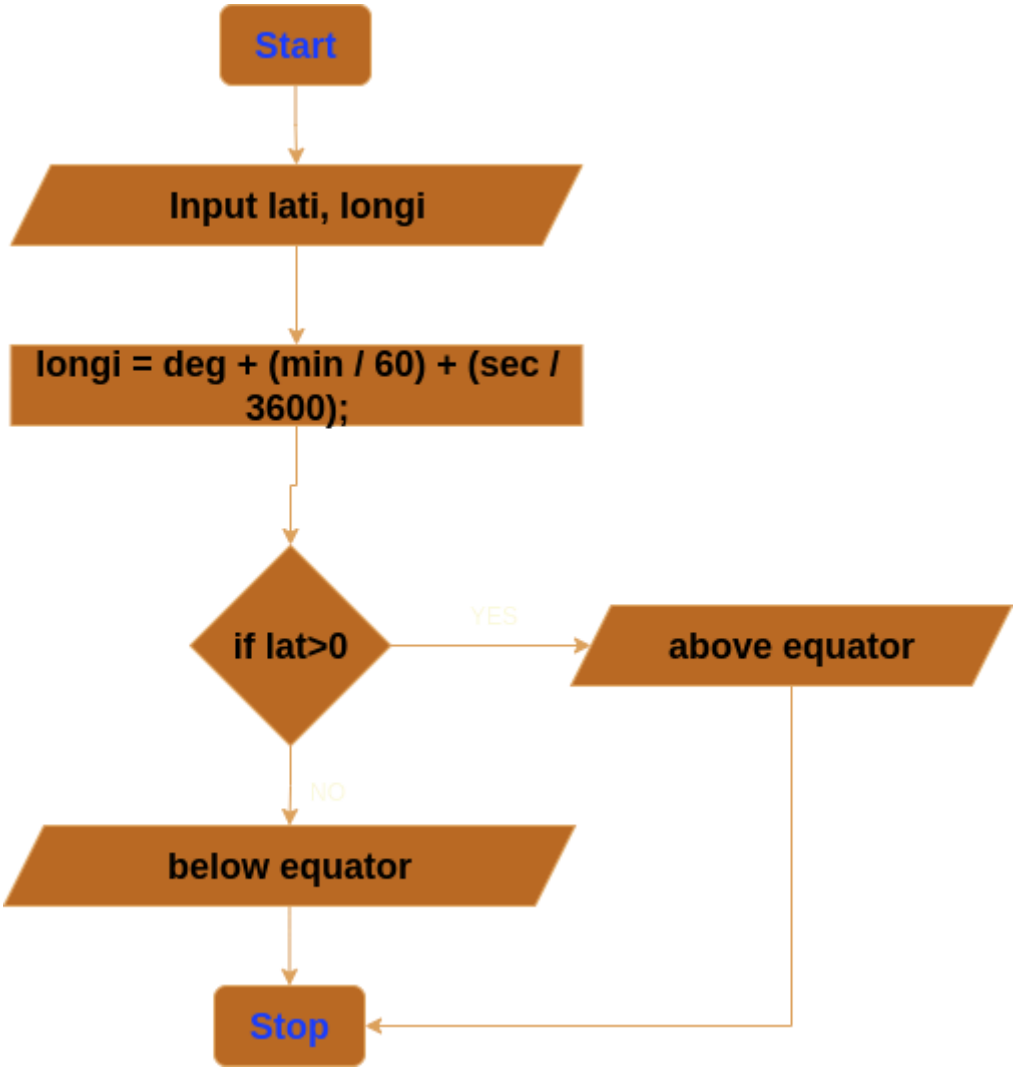
Because of that, most years, we round the days in a year down to 365. However, that leftover piece of a day doesn't disappear. To make sure we count that extra part of a day, we add one day to the calendar approximately every four years. Here's a table to show how it works:

Year	Days in Year	Leap Year?
2017	365	No
2018	365	No
2019	365	No
2020	366	Yes



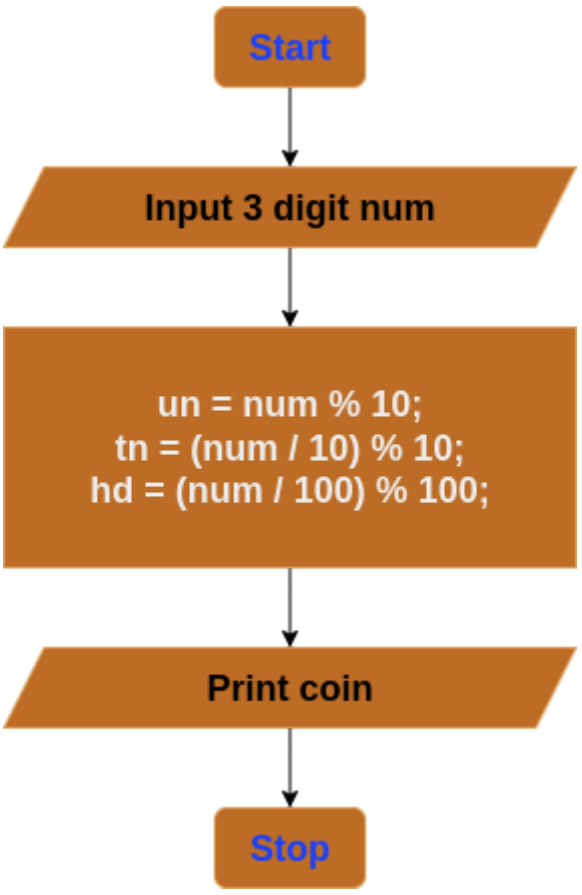
```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS  TEROSHDL: LOG REP

[cc@ncdc-0053 code]$ gcc lab2_task1.c -o task1
[cc@ncdc-0053 code]$ ./task1
Enter the year you want to find whether a leapyear or not:
2024
Yes! this is a leap year.
[cc@ncdc-0053 code]$ ./task1
Enter the year you want to find whether a leapyear or not:
2022
No! this is not a leap year.
[cc@ncdc-0053 code]$ ./task1
Enter the year you want to find whether a leapyear or not:
2026
No! this is not a leap year.
[cc@ncdc-0053 code]$
```

ii. Task 2:



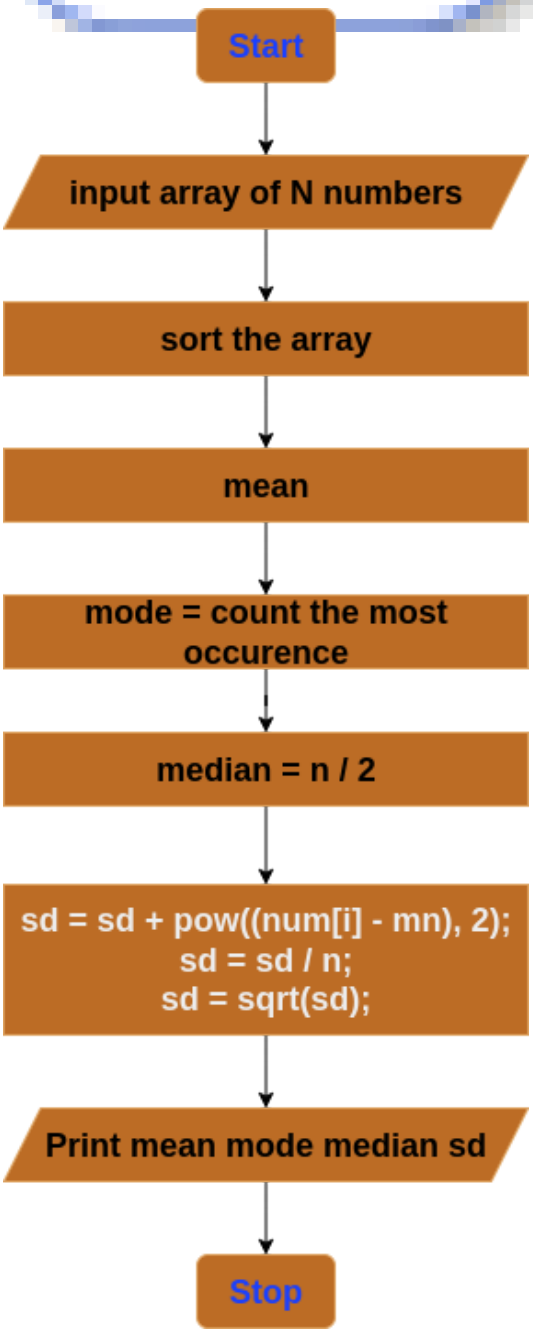
```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

[cc@ncdc-0053 code]$ gcc lab2_task2.c -o task2
[cc@ncdc-0053 code]$ ./task2
Enter the Latitude:
Enter the Latitude's degree:
32
Enter the Latitude's minutes:
45
Enter the Latitude's seconds:
60
Enter the Longitude:
Enter the Longitude's degree:
56
Enter the Longitude's minutes:
74
Enter the Longitude's seconds:
23
Your co-ordinates are: 32.000000,57.000000
The location is above the equator
[cc@ncdc-0053 code]$
```

iii. **Task 3:**



```
PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL   ···
[cc@ncdc-0053 code]$ gcc lab2_task3.c -o task3
[cc@ncdc-0053 code]$ ./task3
Enter a 3 digit number:
345
The unit is. 5
The ten is. 4
The hundred is. 3
[cc@ncdc-0053 code]$
```

iv. **Task 4:**

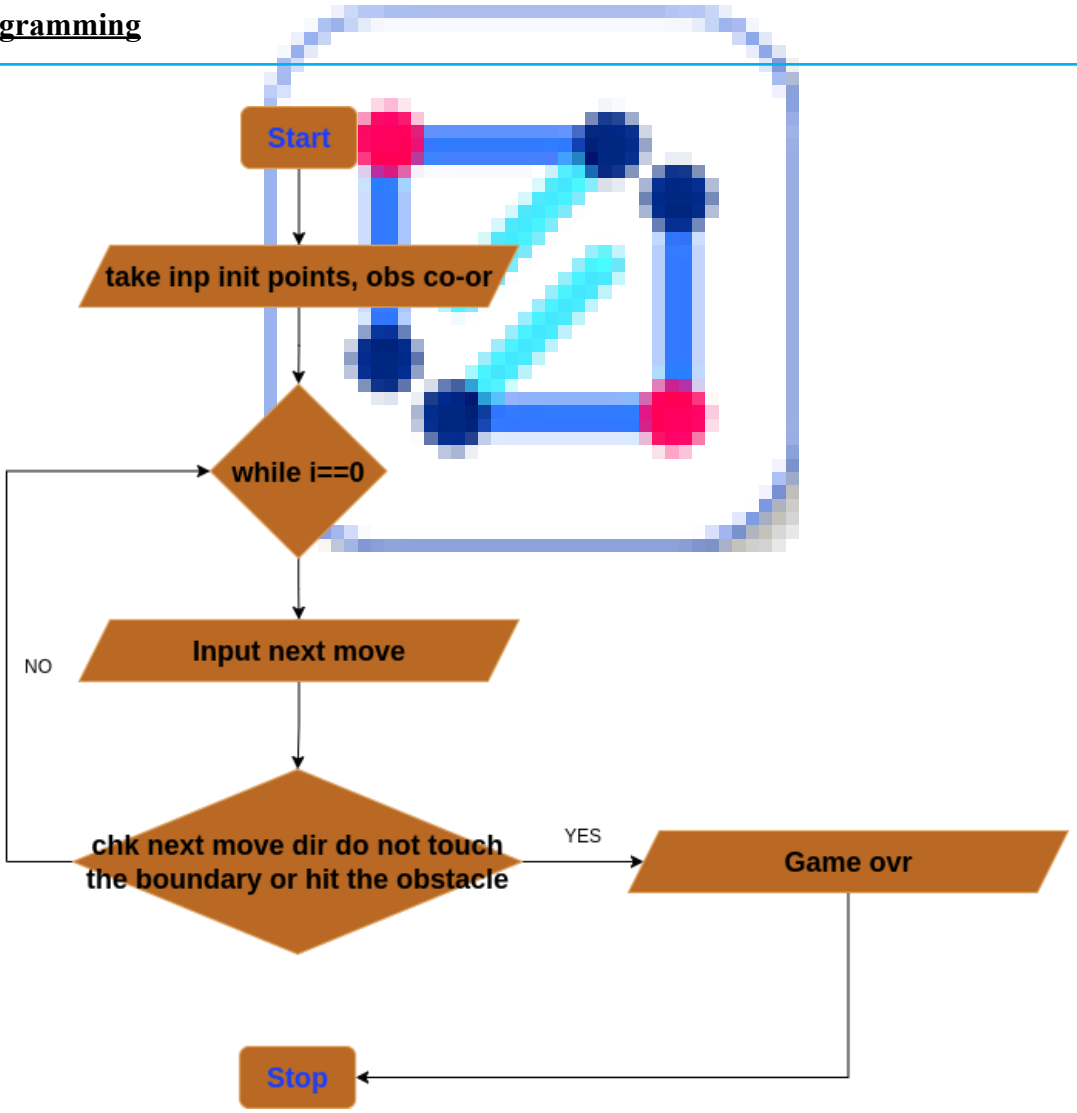


```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS  TERO

[cc@ncdc-0053 code]$ gcc lab2_task4.c -o task4 -lm
[cc@ncdc-0053 code]$ ./task4
Enter numbers one by one:
3
Enter numbers one by one:
2
Enter numbers one by one:
3
Enter numbers one by one:
2
Enter numbers one by one:
2

Sorted array is
2, 2, 2, 3, 3,

The mean is. 2.40
The mode is. 2
The median is. 2
The standard Deviation is. 0.49
[cc@ncdc-0053 code]$
```

v. **Task 5:**

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS  TEROSHD

[cc@ncdc-0053 code]$ gcc lab2_task5.c -o task5 -lm
[cc@ncdc-0053 code]$ ./task5
Enter the co-ordinates of point a:
x1 =  0
y1 =  2
Enter the co-ordinates of point b:
x2 =  7
y2 =  1
Enter the co-ordinates of point c:
x3 =  1
y3 =  -1

 50.00 40.00 10.00

The Triangle of co-ordinates make a rt angle Triangle
[cc@ncdc-0053 code]$
```

vi. **Task 6:**




```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

[cc@ncdc-0053 code]$ gcc lab2_task6.c -o task6
[cc@ncdc-0053 code]$ ./task6
Enter the co-ordinates of initial point.
x1 =  1
y1 =  2
Enter the co-ordinates of obstacle.
x2 =  1
y2 =  3
Enter the direction you want to go...
u
Next move
Enter the direction you want to go...
u
Sorry You exceed the boundary.
Game Over
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PO

[cc@ncdc-0053 code]$ ./task6
Enter the co-ordinates of initial point.
x1 =  1
y1 =  2
Enter the co-ordinates of obstacle.
x2 =  1
y2 =  3
Enter the direction you want to go...
u
Next move
Enter the direction you want to go...
r
Next move
Enter the direction you want to go...
r
Next move
Enter the direction you want to go...
r
Next move
Enter the direction you want to go...
d
Next move
Enter the direction you want to go...
d
```

vii. **Task 7:**

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS  TEROSHDL: LOG REPORT  ...

[cc@ncdc-0053 lab2_files_bitops]$ gcc bit_ops.c test_bit_ops.c -o bit_ops
[cc@ncdc-0053 lab2_files_bitops]$ ./bit_ops

Testing get_bit()

get_bit(0x0000004e,0)returned 0x00000000, correct
get_bit(0x0000004e,1)returned 0x00000001, correct
get_bit(0x0000004e,5)returned 0x00000000, correct
get_bit(0x0000001b,3)returned 0x00000001, correct
get_bit(0x0000001b,2)returned 0x00000000, correct
get_bit(0x0000001b,9)returned 0x00000000, correct

Testing set_bit()

set_bit(0x0000004e,2,0) returned 0x0000004e but we expected 0x0000004a
set_bit(0x0000006d,0,0) returned 0x0000006d but we expected 0x0000006c
set_bit(0x0000004e,2,1) returned 0x0000004e, correct
set_bit(0x0000006d,0,1) returned 0x0000006d, correct
set_bit(0x0000004e,9,0) returned 0x0000004e, correct
set_bit(0x0000006d,4,0) returned 0x0000006d, correct
set_bit(0x0000004e,9,1) returned 0x0000024e, correct
set_bit(0x0000006d,7,1) returned 0x000000ed, correct

Testing flip_bit()

flip_bit(0x0000004e,0) returned 0x0000004f, correct
flip_bit(0x0000004e,1) returned 0x0000004c, correct
flip_bit(0x0000004e,2) returned 0x0000004a, correct
flip_bit(0x0000004e,5) returned 0x0000006e, correct
flip_bit(0x0000004e,9) returned 0x0000024e, correct

[cc@ncdc-0053 lab2_files_bitops]$
```

2. Critical Analysis: (*Write you critical analysis / conclusion here*)

In this lab we learned the conditional way of coding like switch case and if-else. Also we implement loops in this lab. A bit manipulation operation in which we do bit set, get, and flip by logical operations.