

EXPERIMENT 6

6.1.1 Incremented Date

ALGORITHM

Step 1: Start the program.

Step 2: Input day (DD).

Step 3: Input month (MM).

Step 4: Input year (YYYY).

Step 5: Check if $\text{year} \leq 0$.

If yes, print "Invalid Date" and go to Step 20.

Step 6: Check if $\text{month} < 1$ or $\text{month} > 12$.

If yes, print "Invalid Date" and go to Step 20.

Step 7: Determine if the year is a leap year using the condition:

If $(\text{year} \% 400 == 0)$ OR $(\text{year} \% 4 == 0 \text{ AND } \text{year} \% 100 \neq 0)$,

then $\text{leap_year} = \text{True}$,

else $\text{leap_year} = \text{False}$.

Step 8: Set maximum days in the month:

If month is 1, 3, 5, 7, 8, 10, or 12 $\rightarrow \text{max_days} = 31$.

Step 9: Else if month is 4, 6, 9, or 11 $\rightarrow \text{max_days} = 30$.

Step 10: Else if month is 2:

If leap_year is True $\rightarrow \text{max_days} = 29$.

Else $\rightarrow \text{max_days} = 28$.

Step 11: Check if $\text{day} < 1$ or $\text{day} > \text{max_days}$.

If yes, print "Invalid Date" and go to Step 20.

Step 12: If $\text{day} < \text{max_days}$, then

$\text{new_day} = \text{day} + 1$

$\text{new_month} = \text{month}$

$\text{new_year} = \text{year}$

Step 13: Else if $\text{day} == \text{max_days}$ and $\text{month} \neq 12$, then

$\text{new_day} = 1$

$\text{new_month} = \text{month} + 1$

$\text{new_year} = \text{year}$

Step 14: Else if $\text{day} == 31$ and $\text{month} == 12$, then

$\text{new_day} = 1$

$\text{new_month} = 1$

$\text{new_year} = \text{year} + 1$

Step 15: Display the incremented date in the format

DD-MM-YYYY

EXPERIMENT 6

Step 16: End the program.

PYTHON CODE

```
day = int(input())
```

```
month = int(input())
```

```
year = int(input())
```

```
def is_leap_year(year):
```

```
    if (year % 400 == 0) or (year % 4 == 0 and year % 100 != 0):
```

```
        return True
```

```
    return False
```

```
if year <= 0:
```

```
    print("Invalid Date")
```

```
elif month < 1 or month > 12:
```

```
    print("Invalid Date")
```

```
else:
```

```
    if month in [1, 3, 5, 7, 8, 10, 12]:
```

```
        max_days = 31
```

```
    elif month in [4, 6, 9, 11]:
```

```
        max_days = 30
```

```
    elif month == 2:
```

```
        if is_leap_year(year):
```

```
            max_days = 29
```

```
        else:
```

EXPERIMENT 6

```
max_days = 28
```

```
if day < 1 or day > max_days:
```

```
    print("Invalid Date")
```

```
else:
```

```
    day += 1
```

```
if day > max_days:
```

```
    day = 1
```

```
    month += 1
```

```
if month > 12:
```

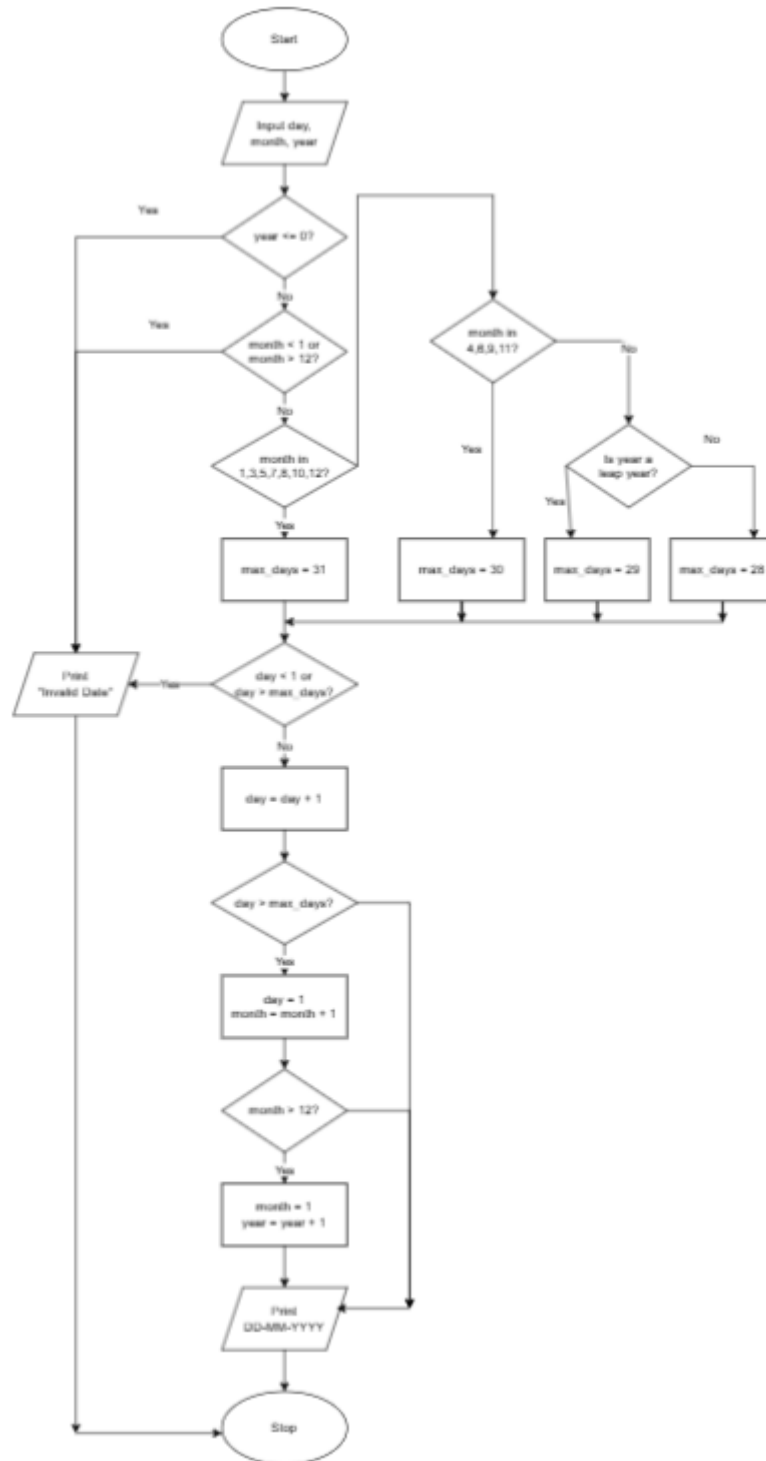
```
    month = 1
```

```
    year += 1
```

```
print(f"{day:02d}-{month:02d}-{year}")
```

FLOWCHART

EXPERIMENT 6



EXPERIMENT 6

EXCECUTION

CODETANTRA

Home

harsh.ghaoghare.batch2025@sitnagpur.siu.edu.inSupportLogout

6.1.1. Incremented Date

Write a Python program to check if a given date is valid. If the date is valid, print the next day's date (incremented date). If the date is invalid, print "Invalid Date".

Date Validation Rules:

1. Valid Month: 1 to 12

2. Valid Day: Depends on the month and year

- January (1), March (3), May (5), July (7), August (8), October (10), December (12): 1 to 31 days
- April (4), June (6), September (9), November (11): 1 to 30 days
- February (2): 1 to 29 days in a leap year and 1 to 28 days in a non-leap year

3. Valid Year: Any positive Integer greater than zero.

Date Increment Rules:

If the date is valid, increment it by one day:

- If it's the last day of the month, move to the 1st day of the next month
- If it's December 31st, move to January 1st of the next year

Input Format:

- First line contains an integer representing the day
- Second line contains an integer representing the month
- Third line contains an integer representing the year

Output Format:

- If the date is valid, print the incremented date in the format:
`<DD>-<MM>-<YYYY>`
- If the date is invalid, print: "Invalid Date".

Sample Test Cases

nextDate.py

```
1 # Input
2 day = int(input())
3 month = int(input())
4 year = int(input())
5
6 # Function to check leap year
7 def is_leap_year(year):
8     if (year % 400 == 0) or (year % 4 == 0 and year % 100 != 0):
9         return True
10    return False
11
12 # Function to get days in a month
13 def get_days_in_month(month, year):
14     if month in [1, 3, 5, 7, 8, 10, 12]:
15         return 31
16     elif month in [4, 6, 9, 11]:
17         return 30
```

Average time: 0.014 s

Maximum time: 0.022 s

13.90 ms

22.00 ms

5 out of 5 shown test case(s) passed

5 out of 5 hidden test case(s) passed

Test case 1

Expected output

Actual output

15

15

3

3

2024

2024

16-03-2024

16-03-2024

Test cases

Terminal

Test cases