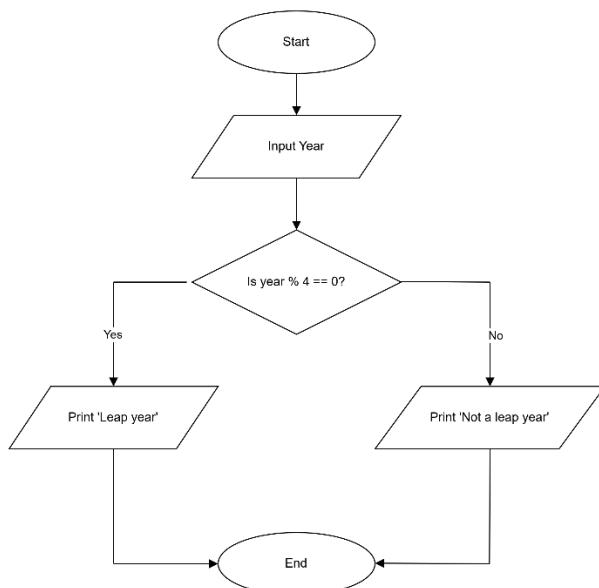


### **5.1.1. Leap Year Checker**

#### **Algorithm:**

1. START
2. INPUT year
3. IF year % 4 == 0 THEN  
    PRINT "Leap year"  
ELSE  
    PRINT "Not a leap year"
4. END

#### **Flowchart:**



#### **Code:**

```
year = int(input())
if year % 4 == 0:
    print("Leap year")
else:
    print("Not a leap year")
```

## Code Tantra Execution:

The screenshot shows the Code Tantra platform interface for executing a Python script named `leapYear.py`. The code checks if a given year is a leap year or not.

```
1 year = int(input())
2 if (year % 4 == 0):
3     print("Leap year")
4 else:
5     print("Not a leap year")
```

The execution results indicate that 2 out of 2 shown test cases passed, and 2 out of 2 hidden test cases passed. The average time for execution is 0.009 s, and the maximum time is 0.012 s. Test case 1 (2024) shows the expected output "Leap year" and the actual output "Leap year". Test case 2 (10 ms) also shows the expected and actual outputs as "Leap year".

Sample Test Cases

Average time: 0.009 s Maximum time: 0.012 s  
9.50 ms 12.00 ms

2 out of 2 shown test case(s) passed  
2 out of 2 hidden test case(s) passed

Test case 1 7 ms  
Expected output: 2024 Actual output: Leap year

Test case 2 10 ms  
Expected output: Leap year Actual output: Leap year

Terminal Test cases

< Prev Reset Submit Next >

## **5.1.2. Student Grade Based on Aggregate**

### **Algorithm:**

1. **START**
2. **INPUT** four marks (a, b, c, d) from user
3. **CALCULATE** total\_marks = a + b + c + d
4. **PRINT** total\_marks
5. **CALCULATE** aggregate = total\_marks / 4
6. **PRINT** aggregate with 2 decimal places
7. **CHECK CONDITIONS:**
8. IF aggregate > 75: PRINT "Distinction"
9. ELSE IF aggregate >= 60: PRINT "First Division"
10. ELSE IF aggregate >= 50: PRINT "Second Division"
11. ELSE IF aggregate >= 40: PRINT "Third Division"
12. ELSE: PRINT "Fail"
13. **END**

### **Code:**

```
a, b, c, d = map(int, input().split())
```

```
total_marks = a + b + c + d
```

```
print(total_marks)
```

```
aggregate = total_marks / 4
```

```
print(f"{aggregate:.2f}")
```

```
if aggregate > 75:
```

```
    print("Distinction")
```

```
elif aggregate >= 60:
```

```
    print("First Division")
```

```
elif aggregate >= 50:
```

```
    print("Second Division")
```

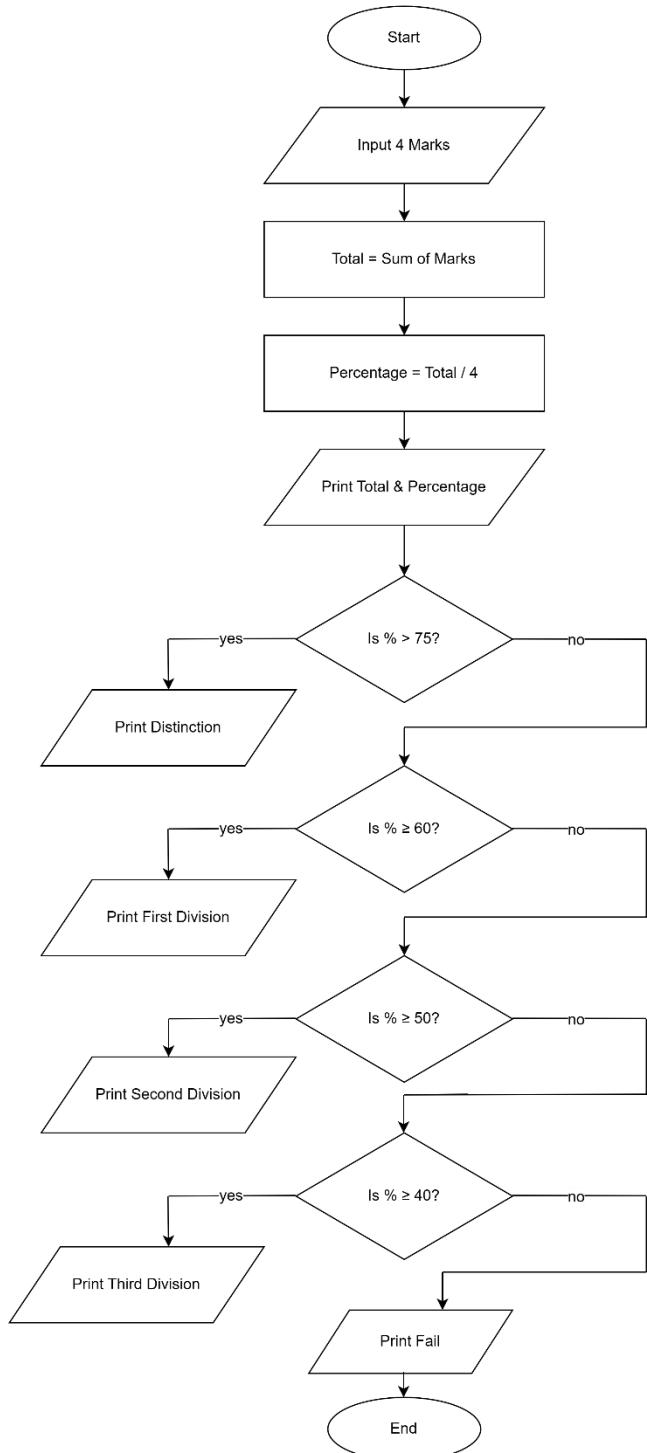
```
elif aggregate >= 40:
```

```

print("Third Division")
else:
    print("Fail")

```

**Flowchart:**



## **Code Tantra Execution:**

The screenshot shows the Code Tantra Execution interface. At the top, there's a navigation bar with the logo, a 'Home' link, user information (ira.handa.batch2025@sitnagpur.siu.edu.in), a 'Support' link, and a 'Logout' button.

The main area has a title '5.1.2. Student Grade Based on Aggregate'. Below it, a text box provides instructions: "Write a program to calculate the total marks, aggregate percentage, and grade of a student based on marks in four subjects. The grade is determined as follows:" followed by a bulleted list of grade criteria. To the right of this text is a code editor window titled 'studentG...'. The code is a Python script that calculates total marks, aggregate percentage, and student grade based on marks in four subjects (a, b, c, d). It uses if-elif-else statements to determine the grade (Distinction, First Division, Second Division, Third Division, or Fail) and prints the results. Below the code editor is a terminal window showing the command 'Terminal' and a 'Test cases' section. At the bottom, there are buttons for 'Sample Test Cases' (with a plus sign), 'Terminal', 'Test cases', and navigation links ('< Prev', 'Reset', 'Submit', 'Next >').

```
a, b, c, d = map(int, input().split())
total_marks = a + b + c + d
print(total_marks)
Aggregate = total_marks / 4
print(f"Aggregate:{.2f}")
if Aggregate > 75 :
    print("Distinction")
elif Aggregate >= 60 :
    print("First Division")
elif Aggregate >=50 :
    print("Second Division")
elif Aggregate >= 40 :
    print("Third Division")
else :
    print("Fail")
```