Type of a Triangle

Given one of the measures of the interior angle of the triangle. Your task is to determine whether the triangle is "**Acute angle**", "**Obtuse angle**", or "**Right angle**" triangle based on the given angle. **If the angle measure is 180^0 or greater than 180^o**, the output should be "**invalid**".

Note:

- Acute angle - less than 90^o
- Obtuse angle - greater than 90^o and less than 180^0
- Right angle - exactly 90^o

Constraints:

- 1^o <= angle of the triangle <= 360^o
- The measure of the angles should be positive

Sample Test Case:

60
Acute angle

Explanation: In the test case, the given angle is less than 90 degrees, So the output is an Acute angle.

Sample Test Cases

Test Case 1:

Expected Output:

60
Acute angle

Test Case 2:

Expected Output:

typeoftriangle.py

```
1  # write you code here
2  angel = float(input())
3
4  if(angel < 90):
5      print("Acute angle")
6  elif(angel > 90 and angel < 180):
7      print("Obtuse angle")
8  elif(angel == 90):
9      print("Right angle")
10 elif(angel >= 180 ):
11     print("invalid")
```

Last Character of a String

Take a string from the user and verify whether the last character of the string is an **alphabet** or a **digit** or a **character**.

Constraints:

- 1 <= length of the string <= 10^3
- Alphabet can be uppercase or lowercase ('A'--'Z' or 'a'--'z')
- Except numeric data('0'----'9') and alphabets rest all came into the category of character (i.e special characters('@','#',.....)

Sample Test case:

1r#tudn ---> (the first line of the input contains the string)
alphabet ---> (Print the result as required)

Explanation :

In the test case, the last character of the input string is n, hence the output is an alphabet.

Sample Test Cases

Test Case 1:

Expected Output:

```
1r#tudn
alphabet
```

Test Case 2:

Expected Output:

```
$ret%
character
```

firstcharacter.py

Submit

```
# write your code here
st = input()
last_char = st[-1]

if last_char.isalpha():
    print("alphabet")
elif not last_char.isdigit():
    print("character")
else:
    print("digit")
```

Finish | Submit | Prev | Next

Comparison Operations

Take two integers and a comparison operator as input from the user. Your task is to perform the given operation and print the result, the first integer will be operated with another integer using the given operator. If the operator given by the user is inappropriate, then it should print the message **"invalid".**

Note:

- comparison operators are **==, !=, >, >=, <, <=.**
- Return the answer in Boolean data type (True or False).

Constraints:

1 <= integers <= 10^6

Sample Test Case:

12
2
>
True

Explanation:

The first two lines of the input contain the integers, and the third line consists of the operator given by the user.
Recall the concept of the comparison operators. In the test case, 12 is greater than 2. So the output will be True.

Sample Test Cases

Test Case 1:

Expected Output:

12

operation.py

Submit

```
# write your code here
a = int(input())
b = int(input())

operator = input().strip()

if operator == "==":
    print(a == b)
elif operator == "!=":
    print(a != b)
elif operator == ">":
    print(a > b)
elif operator == ">=":
    print(a >= b)
elif operator == "<":
    print(a < b)
elif operator == "<=":
    print(a <= b)
else:
    print("invalid")
```

pe of a Triangle

Given the **three sides** of a triangle as input form the user, Your task is to check whether it is an **"Equilateral"**, **"Isosceles"**, or **"Scalene"** triangle.

Note:

- An **equilateral triangle** is a triangle in which **all three sides are equal**.
- A **scalene triangle** is a triangle that has **three unequal sides**.
- An **isosceles triangle** is a triangle with **(at least) two equal sides**.

Sample Test case :

side1: 11
side2: 13
side3: 10
Scalene

Explanation:

In the test case, All three sides are unequal, so that the output will be a scalene triangle

ample Test Cases

Test Case 1:

Expected Output:

```
side1: 24
side2: 24
side3: 24
Equilateral
```

Test Case 2:

Expected Output:

```
side1: 14
```

typeoftriangle.py

Submit

```python
# write your code here
side1 = int(input("side1: "))
side2 = int(input("side2: "))
side3 = int(input("side3: "))

if(side1 == side2 == side3):
    print("Equilateral")
elif(side1 == side2 or side2 == side3 or side1 == side3):
    print("Isosceles")
else:
    print("Scalene")
```

Finish

Submit Prev Next

Find the Weekday

Take an integer from the user. Your task is to find the corresponding weekday. If the entered value is greater than 7, it should print **"invalid"**.

Constraints :

- 1 <= integer <= 10^2
- Consider **"Sunday"** as the first day of the week.

Sample Test case:

4 ----> (The first line of the input contains the integer)

Wednesday ----> (Print the corresponding weekday to the given integer)

Explanation:

The fourth day in the week is Wednesday, So the output is Wednesday.

Sample Test Cases

Test Case 1:

Expected Output:

```
1
Sunday
```

Test Case 2:

Expected Output:

```
4
Wednesday
```

Test Case 3:

Expected Output:

findmonth.py

Submit

```
# write your code here
week = int(input())

if(week == 1):
    print("Sunday")
elif(week == 2):
    print("Monday")
elif(week == 3):
    print("Tuesday")
elif(week == 4):
    print("Wednesday")
elif(week == 5):
    print("Thrusday")
elif(week == 6):
    print("Friday")
elif(week == 7):
    print("Saturday")
elif(week > 7):
    print("invalid")
```

Marks 5

Jump to

100%

Test time left: 06:56:38

Classify the Result of an Arithmetic Operation

Take two integers **N** and **K** from the user. Raise the power of N to K (**i.e.** N^K). Subtract the sum of **N and K** from N^K

Print "**Good**" if the final result is less than 10

Print "**Better**" if the final result is equal to 10.

Print "**Best**" if the final result is greater than 10.

Constraints:

$1 <= N, K <= 10^3$

Sample Test Case:

Input:

N: 2

K: 4

Output:

Better

Explanation:

In the test case, $2^4 = 16$

sum of N and K is **2 + 4 = 6**

subtract the sum of N and K from N^K is **16 - 6 = 10**

10 is equal to 10, therefore the result is Better

Sample Test Cases

Test Case 1:

Expected Output:

```
N: 4
K: 4
Best
```

raisepower.py

Submit

```
# Type Content here...
n = int(input("N: ")) #2
k = int(input("K: ")) #4

power = n**k #16
add = n+k #6
sub = power - add #10

if(sub < 10):
    print("Good")
elif(sub == 10):
    print("Better")
else:
    print("Best")

```

Finish

Submit

Prev

Next

Section - 1 [Section 1 of 1] | Q1 of 7 | Marks 5 | Jump to | 100% | Test time left: 0

Number of Digits of a Number

Take an integer as input from the user. Your task is to count the **number of digits** present in the number.

Note: The maximum count of the digits of the given number should be 6

Constraints:

- 1 <= number <= 999999
- No loops to be used to design the logic.

Sample Test case :

2341 ---> (The first line of input contains an integer)
4 ---> (Print the count of the digits of the integer)

Sample Test Cases

Test Case 1:

Expected Output:

10

2

Test Case 2:

Expected Output:

560

3

Test Case 3:

Expected Output:

0000

countdigits.py

Submi

```
# Type Content here...
number = int(input())
if(number >= 1 and number <=9):
    print("1")
elif(number >= 10 and number <=99):
    print("2")
elif(number >=100 and number <= 999):
    print("3")
elif(number >= 1000 and number <= 9999):
    print("4")
# elif(number >= 10000 and number <= 99999):
#     print("5")
# elif(number >= 10000 and number <= 999999):
#     print("6")
```

Finish | Submit | Next