1.1.1. Calculate Momentum

12:05

Write a program that accepts the mass of an object (in kilograms) and its velocity (in meters per second), then calculates and displays the momentum of the object. The momentum p is calculated using the formula:

$$p = m \times v$$

where:
m is the mass of the object (in kilograms).
v is the velocity of the object (in meters per second).

Input Format:
A single floating-point number representing the mass of the object in kilograms.
A single floating-point number representing the velocity of the object in meters per second.

Output Format:
The output will display calculated momentum with appropriate units (kgm/s) (rounded up to 2 decimal places).

Sample Test Cases +

Explorer

calculate...

Submit

```
1  m=float(input())
2  v=float(input())
3  p=m*v
4  print('%0.2f'%p,end="")
5  print("kgm/s")
```

Debugger

Terminal Test cases

< Prev Reset Submit Next >

1.1.2. Conditional Calculation Based on the Number of Digits

16:12

Write a Python program that accepts an integer n as input. Depending on the number of digits in n.

Constraints:
$1 \le n \le 999$

Input Format:
The input consists of a single integer n.

Output Format:
If n is a single-digit number, print its square.
If n is a two-digit number, print its square root (rounded to two decimal places).
If n is a three-digit number, print its cube root (rounded to two decimal places).
Else print "Invalid".

Sample Test Cases +

Explorer

condition...

Submit

```
1  n=int(input())
2  if(0<n<10):
3      print(n*n)
4  elif(10<=n<100):
5      p=n**0.5
6      print( "%0.2f"%p)
7  elif(100<=n<=999):
8      r=n**(1/3)
9      print("%.2f"%r)
10 else:
11     print("Invalid")
```

Debugger

Terminal Test cases

1.1.3. Age and Salary Calculation

Write a Python program that reads the birth date and salary of employees.

Input Format:
The input consists of:
A string representing the birth date of the employee in the format $DD-MM-YYYY$.
A floating-point number representing the salary of the employee in rupees.

Output Format:
The output should include:
The age of the employee.
The salary of the employee in dollars.

Note:
1INR=0.012USD

Sample Test Cases +

Explorer

birthDate...

Submit

Debugger

```
def calculate_age(birthdate):
    date_object = datetime.strptime(birthdate, "%d-%m-%Y")
    today = datetime.today()
    if ((today.month, today.day ) < (date_object.month,
date_object.day)):
        age = today.year-date_object.year-((today.month, today.day ) <
(date_object.month, date_object.day))
        return age
    elif((today.month, today.day ) >
(date_object.month,date_object.day)):
        age = today.year-date_object.year- ((today.month, today.day) >
(date_object.month, date_object.day))
        return age


def convert_salary_to_dollars(salary_in_rupees):
    salary=salary_in_rupees*0.012
    return salary

birthdate = input()
salary_in_rupees = float(input())
age = calculate_age(birthdate)
salary_in_dollars = convert_salary_to_dollars(salary_in_rupees)
print(f"Age: {age}")
print(f"Salary in dollars: {salary_in_dollars: 2f}")
```

Terminal Test cases

< Prev Reset Submit Next >

1.1.4. Reverse a Number

15:54

You are given an integer number. Your task is to reverse the digits of the number and print the reversed number.

Input Format
The input is an integer.

Output Format
Print a single integer which is the reversed number.

Sample Test Cases +

Explorer

reverseN...

Submit

Debugger

```
1  num=int(input())
2  reverse=0
3  while num != 0:
4      digit = num % 10
5      reverse = reverse*10 + digit
6      num = num//10
7
8  print(reverse)
```

Terminal Test cases

< Prev Reset Submit Next >

1.1.5. Multiplication Table

10:48

Write a Python program that takes an integer as input and prints the multiplication table for that integer from 1 to 10.

Input Format:
The first line of input contains an integer that represents the number for which the multiplication table is to be printed.

Output Format:
Print the multiplication table for the given number .

Sample Test Cases +

Explorer

multiplica...

Submit

```
1  n=int(input())
2  i=1
3  while i<=10:
4      print(n,"x",i,"=",n*i)
5      i=i+1
```

Debugger

Terminal Test cases

1.2.1. Pass or Fail

15:36

Write a Python program that accepts the number of courses and the marks of a student in those courses.
The grade is determined based on the aggregate percentage:

- If the aggregate percentage is greater than 75, the grade is Distinction.
- If the aggregate percentage is greater than or equal to 60 but less than 75, the grade is First Division.
- If the aggregate percentage is greater than or equal to 50 but less than 60, the grade is Second Division.
- If the aggregate percentage is greater than or equal to 40 but less than 50, the grade is Third Division.

Input Format:
The first input will be an integer n, the number of courses.
The second input will be n integers representing the marks of the student in each of the n courses, separated by a space.

Output Format:
If the student passes all courses:

- Print the aggregate percentage (rounded to two decimal places).
- Print the grade based on the aggregate percentage.

If the student fails any course (marks < 40 in any course), print:

Sample Test Cases +

Explorer

passorFa...

Submit

Debugger

```python
n=int(input())
marks= list(map(int, input().split()))
if any(mark<40 for mark in marks):
    print("Fail")
else:
    totalmarks = sum(marks)
    Aggregate_percentage=(totalmarks/(n*100))*100
    print(f"Aggregate Percentage: {Aggregate_percentage:.2f}")
    if(Aggregate_percentage > 75):
        print("Grade: Distinction")
    elif(60<=Aggregate_percentage < 75):
        print("Grade: First Division")
    elif(50<=Aggregate_percentage < 60):
        print("Grade: Second Division")
    elif(40<=Aggregate_percentage > 50):
        print("Grade: Third Division")
    else:
        print("Fail")


```

Terminal Test cases

1.2.3. Pattern - 1

19:59

Write a Python program to print a pattern of asterisks in the form of a right-angled triangle.

Input Format:
The input is an integer, representing the number of rows in the pattern.

Output Format
The output should display the pattern of asterisks (*), with each row containing an increasing number of asterisks.

Note:
Refer to the displayed test cases for the sample pattern.

Sample Test Cases +

Explorer

rightangl...

Submit

```
1  n=int(input())
2  for i in range(1,n+1):
3      for j in range(0,i):
4          print("*",end=" ")
5      print()
```

Debugger

Terminal Test cases

‹ Prev Reset Submit Next ›

1.2.4. Pattern - 2

01:20

Write a Python program to print a right-angled triangle pattern of numbers.

Input Format:
The input is an integer, representing the number of rows in the pattern.

Output Format:
The output should display the pattern of numbers, with each row containing increasing numbers starting from 1 up to the row number.

Note:
Refer to the displayed test cases for the sample pattern.

Sample Test Cases +

Explorer

numberP...

Submit

```
1  n=int(input())
2  for i in range(1,n+1):
3      for j in range(1,i+1):
4          print(j,end=" ")
5      print()
```

Debugger

Terminal Test cases

1.2.2. Fibonacci series using Recursive Function

05:28

Write a Python program to find the Fibonacci series of a given number of terms using recursive function calls.

Expected Output-1:
Enter terms for Fibonacci series: 5
0 1 1 2 3

Expected Output-2:
Enter terms for Fibonacci series: 9
0 1 1 2 3 5 8 13 21

Instructions:

- Your input and output must follow the input and output layout mentioned in the visible sample test case.
- Hidden test cases will only pass when users' input and output match the expected input and output.

Sample Test Cases +

Explorer

fib.py

Submit

Debugger

```
1  def fib(n):
2      if n <= 0:
3          return 0
4      elif n == 1:
5          return 1
6      else:
7          return fib(n-1) + fib(n-2)
8
9
10
11
12
13
14
15 n=int(input("Enter terms for Fibonacci series: "))
16 for i in range (n):
17     print(fib(i),end=" ")
```

Terminal Test cases