






 Done Description [Submission view](#) **Available from:** Tuesday, 2 September 2025, 9:15 AM **Due date:** Tuesday, 2 September 2025, 10:45 AM **Requested files:** p1.py, p2.py, p3.py, p4.py, p5.py ( [Download](#))**Type of work:**  Individual work

## Problem 1 - Circle Perimeter Calculator

### Description:

Write a program that calculates the perimeter (circumference) of a circle based on its radius.

### Concept:

This problem assesses fundamental skills: reading user input, converting the input to a numeric type (**typecasting**), applying a given mathematical formula, and formatting the output to a specific number of decimal places.

### Task:

Your program must produce output that exactly matches the format specified. Pay close attention to all text, spaces, and punctuation in your input prompts and your final output.

1. Prompt the user for the circle's radius with the message: `Enter the radius of the circle: .`
2. Convert the input to a floating-point number.
3. Calculate the perimeter using the formula: `Perimeter = 2 * π * r`. Use `3.14159` as the value for  $\pi$ .
4. Print the final perimeter in the format `Perimeter: [calculated_perimeter]`, rounded to two decimal places.

*Note:* To round your final answer to two decimal places, you can use the built-in `round()` function. It takes two arguments: the number to round and the number of decimal places.

For example, `round(123.4567, 2)` will result in `123.46`.

### Example:

(Text in **bold** is what the user types.)

```
Enter the radius of the circle: 5
Perimeter: 31.42
```

(Another example)

```
Enter the radius of the circle: 12.5
Perimeter: 78.54
```

### Restrictions:

The input will be a positive number.

You must use `3.14159` for the value of  $\pi$ .

No external libraries are necessary.

---

## Problem 2 - Count of Numbers Divisible by 3

### Description:

Write a program that takes two positive integers, `a` and `b`, as input and prints the count of numbers divisible by 3 in the range from `a` to `b-1`.

Range of input: `1 <= a < b <= 1000`

### Concept:

This problem tests the ability to use loops and conditional statements to count numbers that meet a divisibility condition.

### Task:

Your program must produce output that exactly matches the format specified. Pay close attention to all text, spaces, and punctuation in your input prompts and your final output.

1. Prompt the user with: `Enter two numbers:`

?

2. Loop through numbers starting from the first input (a) up to, but not including, the second input (b).
3. Count and print how many numbers in this range are divisible by 3.

**Example:**

Enter two numbers:3 10

Result: 3

**Explanation:**

The numbers divisible by 3 between 3 and 9 are: 3, 6, 9 → Count = 3

**Restrictions:**

The input will be two positive integers where  $1 \leq a < b \leq 1000$ .

No external libraries are necessary.

## Problem 3 - Reverse Number Pattern with Nested Loops

**Description:**

Write a program that takes a positive integer  $n$  as input and prints a number pattern where the first line contains the number 1 repeated  $n$  times, the second line contains the number 2 repeated  $n-1$  times, and so on, until the last line contains the number  $n$  repeated once.

**Concept:**

This problem tests the ability to use nested loops with decreasing repetitions to generate structured patterns.

**Task:**

Your program must produce output that exactly matches the format specified. Pay close attention to all text, spaces, and punctuation in your input prompts and your final output.

1. Prompt the user with: Enter a number:
2. Use nested loops to print the required number pattern.
3. Print the pattern row by row. In each row, there is a space after every number.

**Example:**

Enter a number:3

1 1 1

2 2

3

**Restrictions:**

The input will be a positive integer.

No external libraries are necessary.

## Problem 4: Valid Date Checker

**Description**

Write a program that takes three integers representing day, month, and year, and checks if the date is valid in the AD calendar. Assume February has 28 days.

**Concept**

This problem tests **nested if-else logic** and handling of different ranges for day and month.

**Task**

1. Prompt the user with:

Enter day, month and year:

2. Read three integers.

3. Check if the year is non-negative, the month is between 1 and 12, and the day is valid for that month.

4. Print **Valid date** if correct, otherwise print **Invalid date**.

**Input format**

Prompt before input:

Enter day, month and year:

**Output format**

Output must be either:

**Valid date**

or

Invalid date

### Examples

#### Example 1

Enter day, month and year: 30 2 2023

Invalid date

#### Example 2

Enter day, month and year: 31 12 2025

Valid date

### Restrictions

- Assume February has only 28 days.
- Year must be  $\geq 0$ .
- Do not use in built-in functions or libraries.

---

## Problem 5: Power Series ( $i^i$ )

### Description

Generate the series where the  $i$ th term is  $i^i$  ( $i$  raised to the power  $i$ ).

### Task

Read an integer  $n$  and print the series of  $i^i$  up to  $n$  terms. You can assume input will always be greater than 0.

### Input format

Prompt before input:

Enter  $n$ :

### Output format

Output must be the required series.

### Examples

#### Example 1

Enter  $n$ : 4

1 4 27 256

#### Example 2

Enter  $n$ : 5

1 4 27 256 3125

### Restrictions

- Use loops and exponentiation ( $**$ ).Do not use any in-built functions or libraries.
- Assume  $n > 0$ .

## Requested files

### p1.py

```
1 # write your code here below
```

### p2.py

```
1 # write your code here below
```

### p3.py

```
1 # write your code here below
```

### p4.py

```
1 # write your code here below
```

**p5.py**

[VPL](#)