1. BMI Calculator

Objective: Create a function to calculate the Body Mass Index (BMI).
Instructions:

1. Define a function `calculate_bmi` that takes two arguments: `weight` (in kilograms) and `height` (in meters).
2. Use the formula `BMI = weight / (height ** 2)`.
3. Return the BMI value rounded to 2 decimal places.
   Example:

```
calculate_bmi(70, 1.75)  # Output: 22.86
calculate_bmi(95, 1.9)   # Output: 26.32
```

2. Area Calculator

Objective: Write a function to calculate the area of a rectangle or a circle.
Instructions:

1. Define a function `calculate_area` that takes two arguments: `shape` (a string: "rectangle" or "circle") and `value` (a tuple containing dimensions).
2. If the shape is "rectangle", calculate the area as `length * width`.
3. If the shape is "circle", calculate the area as `π * radius ** 2` (use `math.pi`).
4. Return the area rounded to 2 decimal places.
   Example:

```
calculate_area("rectangle", (5, 10))  # Output: 50
calculate_area("circle", (7,))       # Output: 153.94
```

3. Prime Number Checker

Objective: Create a function to check if a number is prime.
Instructions:

1. Define a function `is_prime` that takes an integer `number`.
2. Check if the number is divisible by any value other than 1 and itself.
3. Return `True` if it is prime, otherwise `False`.
   Example:

```
is_prime(7)   # Output: True
is_prime(10)  # Output: False
```

4. Palindrome Checker

Objective: Write a function to check if a string is a palindrome.
Instructions:

1. Define a function `is_palindrome` that takes a string `text` .
2. Remove spaces and convert the string to lowercase.
3. Check if the string reads the same forwards and backwards.
4. Return `True` if it is a palindrome, otherwise `False` .
   Example:

```
is_palindrome("racecar")       # Output: True
is_palindrome("hello world")   # Output: False
```

5. Factorial Calculator

Objective: Create a function to calculate the factorial of a number.
Instructions:

1. Define a function `factorial` that takes an integer `n` .
2. Use a loop or recursion to calculate the factorial (`n! = n * (n-1) * ... * 1`).
3. Return the factorial value.
   Example:

```
factorial(5)   # Output: 120
factorial(0)   # Output: 1
```

6. Fibonacci Sequence Generator

Objective: Write a function to generate the first `n` numbers in the Fibonacci sequence.
Instructions:

1. Define a function `fibonacci` that takes an integer `n` .
2. Generate the sequence where each number is the sum of the two preceding ones, starting with 0 and 1.
3. Return the sequence as a list.
   Example:

```
fibonacci(5)   # Output: [0, 1, 1, 2, 3]
fibonacci(8)   # Output: [0, 1, 1, 2, 3, 5, 8, 13]
```

7. Word Counter

Objective: Create a function to count the number of words in a string.
Instructions:

1. Define a function `word_count` that takes a string `text`.
2. Split the string into words using spaces as the delimiter.
3. Return the count of words.
   Example:

```
word_count("Hello world!")                # Output: 2
word_count("Python is an amazing language.")  # Output: 5
```

8. Reverse a String

Objective: Write a function to reverse a given string.
Instructions:

1. Define a function `reverse_string` that takes a string `text`.
2. Reverse the string using slicing or a loop.
3. Return the reversed string.
   Example:

```
reverse_string("hello")       # Output: "olleh"
reverse_string("Python")      # Output: "nohtyP"
```

9. Sum of Digits

Objective: Write a function to calculate the sum of digits in a number.
Instructions:

1. Define a function `sum_of_digits` that takes an integer `number`.
2. Convert the number to a string to iterate through its digits.
3. Return the sum of all digits.
   Example:

```
sum_of_digits(123)  # Output: 6
sum_of_digits(4567) # Output: 22
```

10. Vowel Counter

Objective: Create a function to count the number of vowels in a string.
Instructions:

1. Define a function `count_vowels` that takes a string `text`.
2. Count the occurrences of vowels (`a, e, i, o, u`) in the string, ignoring case.
3. Return the count.
   Example:

```
count_vowels("hello")          # Output: 2
count_vowels("Python rocks!")  # Output: 3
```