Python Functions

Functions in Python are reusable blocks of code defined using the `def` keyword. They can take parameters and return values using the `return` statement. Python also supports anonymous functions using the `lambda` keyword.

```
"1. Write a Python function named add_numbers that takes two arguments, a and b, and returns their sum."
def add_numbers(a, b):
    return a + b
print(f"Addition of 2 and 3 is {add_numbers(2, 3)}")
print(f"Addition of 5 and 7 is {add_numbers(5, 7)}")
```

```
Addition of 2 and 3 is 5
Addition of 5 and 7 is 12
```

```
"2. Write a function calculate_area that calculates the area of a rectangle. It should take two parameters: length and width. Use keyword arguments when calling the function."
def calculate_area(length, width):
    return length * width
print(f"Area of rectangle with length 5 and width 3 is {calculate_area(5, 3)}")
print(f"Area of rectangle with length 7 and width 4 is {calculate_area(7, 4)}")
```

```
Area of rectangle with length 5 and width 3 is 15
Area of rectangle with length 7 and width 4 is 28
```

```
"3. Write a function is_prime that takes an integer as input and returns True if the number is prime, otherwise returns False."
def is_prime(n):
    if n < 2:
        return False
    for i in range(2, n):
        if n % i == 0:
            return False
    return True
print(f"Is 5 prime? {is_prime(5)}")
print(f"Is 10 prime? {is_prime(10)}")
```

```
Is 5 prime? True
Is 10 prime? False
```

```
"4. Write a function square_numbers that takes a list of numbers as input and returns a list containing the square of each number."
def square_numbers(nums):
    return [x*x for x in nums]
print(f"Square of [2, 3, 4] is {square_numbers([2, 3, 4])}")
print(f"Square of [5, 6, 7] is {square_numbers([5, 6, 7])}")
```

```
Square of [2, 3, 4] is [4, 9, 16]
Square of [5, 6, 7] is [25, 36, 49]
```

```
"5. Write a function is_palindrome that takes a string as input and returns True if the string is a palindrome."
def is_palindrome(s):
    return s == s[::-1]
print(f"Is 'hello' palindrome? {is_palindrome('hello')}")
print(f"Is 'madam' palindrome? {is_palindrome('madam')}")
```

```
Is 'hello' palindrome? False
Is 'madam' palindrome? True
```

```python
"6. Write a function factorial that computes the factorial of a number using a for loop."
def factorial(n):
    f = 1
    for i in range(1,n+1): f *= i
    return f
print(f"Factorial of 5 is {factorial(5)}")
print(f"Factorial of 10 is {factorial(10)}")
```

```
Factorial of 5 is 120
Factorial of 10 is 3628800
```

```python
"7. Write a function fibonacci that takes a number n and prints the first n numbers in the Fibonacci sequence."
def fibonacci(n):
    a,b = 0,1
    for _ in range(n): print(a, end=' '); a,b = b,a+b
print(f"First 10 Fibonacci numbers are:")
fibonacci(10); print()
print(f"First 20 Fibonacci numbers are:")
fibonacci(20)
```

```
First 10 Fibonacci numbers are:
0 1 1 2 3 5 8 13 21 34
First 20 Fibonacci numbers are:
0 1 1 2 3 5 8 13 21 34 55 89 144 233 377 610 987 1597 2584 4181
```

```python
"8. Write a function is_armstrong that takes a number and checks whether it is an Armstrong number."
def is_armstrong(n):
    return sum(int(d)**len(str(n)) for d in str(n)) == n
print(f"Is 153 Armstrong number? {is_armstrong(153)}")
print(f"Is 192 Armstrong number? {is_armstrong(192)}")
```

```
Is 153 Armstrong number? True
Is 192 Armstrong number? False
```

```python
"9. Write a function convert_temperature that converts temperature between Celsius and Fahrenheit."
def convert_temperature(temp, scale):
    return temp*9/5+32 if scale=='C' else (temp-32)*5/9
print(f"0°C is {convert_temperature(0, 'C')}°F")
print(f"32°F is {convert_temperature(32, 'F')}°C")
```

```
0°C is 32.0°F
32°F is 0.0°C
```

```python
"10. Write a function prime_numbers_in_range that takes two integers start and end."
def prime_numbers_in_range(start, end):
    return [n for n in range(max(2,start), end+1) if all(n%i for i in range(2,n))]
print(f"Prime numbers between 10 and 20 are: {prime_numbers_in_range(10, 20)}")
print(f"Prime numbers between 100 and 200 are: {prime_numbers_in_range(100, 200)}")
```

```
Prime numbers between 10 and 20 are: [11, 13, 17, 19]
Prime numbers between 100 and 200 are: [101, 103, 107, 109, 113, 127, 131, 137, 139, 149, 151, 157, 163, 167, 173, 179, 181, 191, 193, 197, 199]
```