Lecture05

July 6, 2025

[4]:
```python
def greet():
    print("Hello, World!")

greet()
```

```
Hello, World!
```

[6]:
```python
def message():
    print('I am Arthur')
    print('King of the Britons')

print('I have a message for you.')
message()
print('Goodbye!')
```

```
I have a message for you.
I am Arthur
King of the Britons
Goodbye!
```

[9]:
```python
def greet(name):
    print(f'Hello, {name}!')

greet("Alice")
```

```
Hello, Alice!
```

[10]:
```python
def add(a, b):
    return a + b

result = add(3, 5)
print(result)
```

```
8
```

[11]:
```python
def greet(name="World"):
    print(f'Hello, {name}!')

greet("Alice")
```

```
Hello, 2!
```

[24]:
```python
def sum_all(*args):
    return sum(args)

print(sum_all(1,2,3,5,4))
```

```
15
```

[31]:
```python
def sum_all(*args):
    count = 0
    for arg in args:
        count += 1
        if len(args) == count:
            print(arg, end="=")
        else:
            print(arg, end="+")
    return sum(args)

print(sum_all(1,2,3,5,4))
```

```
1+2+3+5+4=15
```

[33]:
```python
def find_max(*args):
    if not args:
        return None
    max_value = args[0]
    for number in args:
        if number > max_value:
            max_value = number
    return max_value

result = find_max(3,5,7,2,8)
print(f"The maximum value is: {result}")
```

```
The maximum value is: 8
```

[42]:
```python
def print_all(*args):
    for index, arg in enumerate(args):
        print(f"Argument {index + 1}: {arg}")
    print(type(args))

print_all("Python", 3.8, True, [1, 2, 3], {"key": "value"})
```

```
Argument 1: Python
Argument 2: 3.8
Argument 3: True
Argument 4: [1, 2, 3]
Argument 5: {'key': 'value'}
<class 'tuple'>
```

[43]:
```
def display_info(**kwargs):
    for key, value in kwargs.items():
        print(f"{key}: {value}")
    print(type(kwargs))

display_info(name="Alice", age=30, city="New York")
```

```
name: Alice
age: 30
city: New York
<class 'dict'>
```

[41]:
```
def calculate_stats(numbers):
    total_sum = sum(numbers)
    average = total_sum / len(numbers)
    maximum = max(numbers)
    minimum = min(numbers)
    return total_sum, average, maximum, minimum

numbers = [5, 10, 15, 20, 25]
total, avg, max_num, min_num = calculate_stats(numbers)

print(f"Total Sum: {total}")
print(f"Average: {avg}")
print(f"Maximum: {max_num}")
print(f"Minimum: {min_num}")
```

```
Total Sum: 75
Average: 15.0
Maximum: 25
Minimum: 5
```

[87]:
```
def is_armstrong(input_num):
    str_nums = str(input_num)
    total_num = 0
    for str_num in str_nums:
        total_num += int(str_num) ** (len(str_nums))
    if total_num == input_num:
        return True
    else:
        return False

print(is_armstrong(153))
```

```
True
```

[ ]:
```
def is_armstrong(input_num):
    str_nums = str(input_num)
    list_num = []
    for str_num in str_nums:
        list_num.append(int(str_num) ** (len(str_nums)))
    if sum(list_num) == input_num:
        result = True
    else:
        result = False
    if result:
        str_result = f"{result} ({sum(list_num)} = "
        for num in range(len(str_nums)):
            if num != len(str_nums) - 1:
                str_result += f"{str_nums[num]}^{len(str_nums)} + "
            else:
                str_result += f"{str_nums[num]}^{len(str_nums)})"
        return str_result, result
    else:
        str_result = f"False ({input_num} is not an Armstrong number)"
        return str_result, result

for i in range(1, 100000):
    x, y = is_armstrong(i)
    if y:
        print(x)
```

[79]:
```
def my_function():
    local_variable = "I'm inside the function"
    print(local_variable)

my_function()
# print(local_variable)
```

```
I'm inside the function
```

[83]:
```
global_variable = "I'm outside the function"

def my_function():
    print(global_variable)

my_function()
print(global_variable)
```

```
I'm outside the function
I'm outside the function
```

[84]:
```
import random

HEADS = 1
TAILS = 2
TOSSES = 10
```

```
def tosses_coin():
    for toss in range(TOSSES):
        if random.randint(HEADS, TAILS) == HEADS:
            print('Heads')
        else:
            print('Tails')

tosses_coin()
```

```
Tails
Heads
Heads
Heads
Tails
Heads
Tails
Tails
Heads
Heads
```

[85]:
```
counter = 0

def increment():
    global counter
    counter += 1

increment()
increment()
print(counter)
```

```
2
```

[90]:
```
def factorial(n):
    print(n)
    if n == 0:
        return 1
    else:
        return n * factorial(n - 1)

print(factorial(5))
```

```
5
4
3
2
1
0
120
```

[95]:
```python
def fibonacci(n):
    if n == 0:
        return 0
    elif n == 1:
        return 1
    else:
        return fibonacci(n - 1) + fibonacci(n - 2)

print(fibonacci(9))
```

```
34
```