函数

函数

- 参数传递
- 函数返回值
- 变量作用域
- 匿名函数
- 综合案例

参数传递

◆ 形参与实参

```python
# 形参是函数定义时的参数，实参是函数调用时的参数

def create_model(layers, units):  # layers和units是形参
    print(f"Creating a model with {layers} layers and {units} units in each layer.")

# 调用函数
create_model(3, 128)  # 3和128是实参
```

```
Creating a model with 3 layers and 128 units in each layer.
```

参数传递

◆ 位置参数

```python
# 位置参数的顺序很重要

def create_model(layers, units):
    print(f"Creating a model with {layers} layers and {units} units in each layer.")

# 调用函数
create_model(3, 128)
create_model(128, 3)
```

```
Creating a model with 3 layers and 128 units in each layer.
Creating a model with 128 layers and 3 units in each layer.
```

参数传递

◆ 关键字参数

```python
# 使用关键字参数调用函数

def create_model(layers, units):
    print(f"Creating a model with {layers} layers and {units} units in each layer.")

# 调用函数
create_model(units=128, layers=3)  # 使用关键字参数，顺序不重要
create_model(layers=3, units=128)
```

```
Creating a model with 3 layers and 128 units in each layer.
Creating a model with 3 layers and 128 units in each layer.
```

参数传递

◆ 默认参数

```python
# 使用默认参数值

def create_model(layers=3, units=128):
    print(f"Creating a model with {layers} layers and {units} units in each layer.")

# 调用函数
create_model()  # 使用默认值
```

```
Creating a model with 3 layers and 128 units in each layer.
```

参数传递

◆ 可变参数

```python
# 使用可变参数接收多个参数值

def add_layers(model, *layers):
    for layer in layers:
        print(f"Adding layer {layer} to model {model}.")

# 调用函数
add_layers("Model1", "conv", "relu", "softmax")
```

```
Adding layer conv to model Model1.
Adding layer relu to model Model1.
Adding layer softmax to model Model1.
```

函数返回值

◆ 通过 return 返回

```python
# 函数返回模型的信息

def create_model(layers, units):
    info = f"Creating a model with {layers} layers and {units} units in each layer."
    return info

# 调用函数
model_info = create_model(3, 128)
print(model_info)
```

```
Creating a model with 3 layers and 128 units in each layer.
```

变量作用域

◆ 全局变量

```python
# 全局变量

MODEL_NAME = "CNN"

def print_model_name():
    print(f"The model name is {MODEL_NAME}.")

# 调用函数
print_model_name()
```

```
The model name is CNN.
```

变量作用域

◆ 局部变量

```python
# 局部变量

def create_model():
    model_name = "RNN"  # 局部变量
    print(f"Creating a model named {model_name}.")

# 调用函数
create_model()

print(model_name) # 此行代码会报错
```

```
Creating a model named RNN.
```

匿名函数

- 冒号前面是输入，冒号后面是输出

```python
# 使用lambda创建匿名函数

calculate_units = lambda layers: layers * 128

# 调用函数
units = calculate_units(3)
print(f"Total units: {units}")
```

```
Total units: 384
```

综合案例

- 创建一个函数，输入包括包括：
  - 输入图像尺寸（channels, height, width）
  - 卷积核大小（默认为3）
  - 边缘补零, padding（默认为0）
  - 卷积步长（默认为1）

  最终需要返回卷积后图像的尺寸

  计算方法为：

  new_height = ((height + 2 * padding - kernel) // stride) + 1

  new_width = ((width + 2 * padding - kernel) // stride) + 1

  最后将 channels, new_height, new_width 组成新的元组output_size进行返回