1. Python介紹
   * Python的歷史和優點
   * Python的應用領域
   * 安裝Python
2. Python基礎語法
   * 變量和賦值
   * 數字和算術運算符
   * 字符串和索引
   * 列表和索引
   * 條件語句和邏輯運算符
   * 循環語句
3. Python函數
   * 定義函數
   * 函數參數
   * 返回值
4. Python模組和庫
   * 模組介紹
   * 常用庫介紹
   * 用pip安裝庫
5. Python項目
   * 應用上述知識進行一些簡單的項目編寫
   * 可以是簡單的文字遊戲或計算機等

**1. 變量和賦值**

在Python中，我們可以通過變量來存儲數據，並且可以將數據賦值給變量。Python中的變量名是區分大小寫的。

變量賦值的一般格式如下：

variable\_name = value

其中，**variable\_name**是變量名，**value**是要賦給變量的值。

以下是一個簡單的變量賦值的例子：

message = "Hello, World!"

print(message)

在上面的代碼中，我們創建了一個變量**message**，並將字符串**"Hello, World!"**賦值給了它。然後，我們使用**print()**函數將**message**的值打印到控制台中。

**2. 數字和算術運算符**

在Python中，我們可以使用數字和算術運算符來執行數學運算。下面是一些常用的算術運算符：

* **+** 加法運算符
* **-** 減法運算符
* **\*** 乘法運算符
* **/** 除法運算符
* **\*\*** 次方運算符
* **%** 取模運算符

以下是一個簡單的例子，演示了如何使用這些算術運算符：

# 數字和算術運算符

x = 3

y = 2

print("x + y =", x + y) # 輸出：x + y = 5

print("x - y =", x - y) # 輸出：x - y = 1

print("x \* y =", x \* y) # 輸出：x \* y = 6

print("x / y =", x / y) # 輸出：x / y = 1.5

print("x \*\* y =", x \*\* y) # 輸出：x \*\* y = 9

print("x % y =", x % y) # 輸出：x % y = 1

**3. 字符串和索引**

在Python中，我們可以使用字符串來存儲和操作文本數據。以下是一些常用的字符串操作：

* 使用引號來定義字符串，可以使用單引號或雙引號。
* 使用**len()**函數來獲取字符串的長度。
* 使用索引來訪問字符串中的單個字符，索引從0開始。

# 3. 字符串和索引（續）

# 定義一個字符串

message = "Hello, World!"

# 獲取字符串的長度

print("The length of the message is", len(message)) # 輸出：The length of the message is 13

# 使用索引來訪問字符串中的單個字符

print("The first character is", message[0]) # 輸出：The first character is H

print("The last character is", message[-1]) # 輸出：The last character is !

# 字符串切片

print("The first five characters are", message[:5]) # 輸出：The first five characters are Hello

print("The last five characters are", message[-5:]) # 輸出：The last five characters are World!

print("Every other character is", message[::2]) # 輸出：Every other character is Hlo ol

# 字符串方法

print(message.upper()) # 將字符串轉換為大寫字母，輸出：HELLO, WORLD!

print(message.lower()) # 將字符串轉換為小寫字母，輸出：hello, world!

print(message.replace("Hello", "Hi")) # 將字符串中的一個子串替換為另一個子串，輸出：Hi, World!

**4. 列表和索引**

在Python中，我們可以使用列表來存儲一系列數據。以下是一些常用的列表操作：

* 使用方括號來定義列表。
* 使用**len()**函數來獲取列表的長度。
* 使用索引來訪問列表中的單個元素，索引從0開始。
* 使用切片來訪問列表的一部分。
* 使用**append()**方法向列表中添加新元素。
* 使用**insert()**方法在列表的指定位置插入新元素。
* 使用**remove()**方法刪除列表中的指定元素。
* 使用**pop()**方法刪除列表中的最後一個元素。

以下是一個簡單的例子，演示了如何定義列表、獲取列表的長度以及使用索引和切片來訪問列表中的元素：

# 列表和索引

fruits = ["apple", "banana", "cherry", "orange", "kiwi", "melon", "mango"]

print("The length of the list is", len(fruits)) # 輸出：The length of the list is 7

print("The first item is", fruits[0]) # 輸出：The first item is apple

print("The last item is", fruits[-1]) # 輸出：The last item is mango

print("The first three items are", fruits[:3]) # 輸出：The first three items are ['apple', '# 4. 列表和索引（續）

# 定義一個列表

numbers = [1, 2, 3, 4, 5, 6, 7, 8, 9]

# 向列表末尾添加新元素

numbers.append(10)

print(numbers) # 輸出：[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

# 在列表的指定位置插入新元素

numbers.insert(0, 0)

print(numbers) # 輸出：[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

# 刪除列表中的指定元素

numbers.remove(5)

print(numbers) # 輸出：[0, 1, 2, 3, 4, 6, 7, 8, 9, 10]

# 刪除列表中的最後一個元素

numbers.pop()

print(numbers) # 輸出：[0, 1, 2, 3, 4, 6, 7, 8, 9]

# 翻轉列表中的元素

numbers.reverse()

print(numbers) # 輸出：[9, 8, 7, 6, 4, 3, 2, 1, 0]

# 對列表中的元素進行排序

numbers.sort()

print(numbers) # 輸出：[0, 1, 2, 3, 4, 6, 7, 8, 9]

# 列表可以包含任何類型的數據，甚至可以包含其他列表

mixed\_list = ["apple", 1, True, [1, 2, 3]]

print(mixed\_list) # 輸出：['apple', 1, True, [1, 2, 3]]

**5. 控制流程**

在 Python 中，可以使用條件語句和循環語句來控制程序的流程。

* **if**語句用於條件判斷，當條件成立時執行特定的代碼塊。
* **if-else**語句用於在條件成立時執行一個代碼塊，在條件不成立時執行另一個代碼塊。
* **if-elif-else**語句用於在多個條件中進行選擇，當條件成立時執行特定的代碼塊，如果沒有任何條件成立，則執行最後的**else**代碼塊。
* **while**循環用於重複執行代碼塊，直到條件不成立為止。

**for**循環用於對一個序列進行迭代，重複執行代碼塊直到序列中的所有元素都被處理。

以下是幾個常見的控制流程語句的使用方法：

* # 5. 控制流程
* # if語句
* age = 18
* if age >= 18:
* print("你已經成年了")
* # if-else語句
* age = 16
* if age >= 18:
* print("你已經成年了")
* else:
* print("你還未成年")
* # if-elif-else語句
* score = 85
* if score >= 90:
* print("優秀")
* elif score >= 80:
* print("良好")
* elif score >= 60:
* print("及格")
* else:
* print("不及格")
* # while循環
* i = 1
* while i <= 5:
* print(i)
* i += 1
* # for循環
* numbers = [1, 2, 3, 4, 5]
* for number in numbers:
* print(number)

**6. 函數**

在 Python 中，可以使用函數來組織和重複使用代碼。

* 定義函數需要使用**def**關鍵字。
* 函數可以有多個參數，也可以沒有參數。
* 函數可以返回一個值或多個值，使用**return**語句返回。

以下是函數的使用方法：

# 6. 函數

# 定義一個求和函數

def add(a, b):

return a + b

# 使用函數

result = add(3, 5)

print(result) # 輸出：8

# 定義一個無參數函數

def say\_hello():

print("Hello!")

# 使用函數

say\_hello() # 輸出：Hello!

# 定義一個有多個返回值的函數

def calculate(a, b):

add\_result = a + b

subtract\_result = a - b

multiply\_result = a \* b

divide\_result = a / b

return add\_result, subtract\_result, multiply\_result, divide\_result

# 使用函數

add\_result, subtract\_result, multiply\_result, divide\_result = calculate(5, 2)

print(add\_result) # 輸出：7

print(subtract\_result) # 輸出：3

print(multiply\_result) # 輸出：10

print(divide\_result) # 輸出：2.5