关于本节中要掌握的内容

1.python开发环境安装

2.标识符规则

3.变量定义

4.交换变量

5.格式化输出

6.作业提交-git与github的介绍与使用

1.安装Anaconda Navigator

- 访问https://www.anaconda.com/download 并下载对应平台的anaconda
- 使用默认选择并安装
- 打开Anaconda Navigator并在右侧中安装Spyder
- 运行Spyder

2.python中的标识符

python标识符的命名规则

- 不能是python的关键字
- 不能包含空格
- 只能是A-Z、a-z、0-9、下划线的组合
- 只能以字母或下划线开头

python中的关键字

使用keyword获取python的关键字

```
import keyword
print(keyword.kwlist)
```

输出

```
['False', 'None', 'True', 'and', 'as', 'assert', 'async',
'await', 'break', 'class', 'continue', 'def', 'del', 'elif',
'else', 'except', 'finally', 'for', 'from', 'global', 'if',
'import', 'in', 'is', 'lambda', 'nonlocal', 'not', 'or',
'pass', 'raise', 'return', 'try', 'while', 'with', 'yield']
```

有效的标识符示例

- var1
- _var1
- _1_var
- var_1

无效的标识符

- !var1
- 1var
- 1_var
- var#1
- var 1

3.使用python定义变量并进行简单计算

pyhton代码

```
a = 1
b = 2 + 3
c = (a + b) * 5
print(30)
```

输出

```
30
```

完成下面的练习，找出结果有什么不同

- 计算 (2 + 3) * 4 / 2

python代码

```
a = 1
b = 2 + 3
c = (a + b) * 5
d = (2 + 3) * 4 / 2
print(c,type(c),d,type(d))
```

输出

```
30 <class 'int'> 10.0 <class 'float'>
```

- 我们发现d输出的是小数类型，python为了确保数值计算的精度和准确度，必定返回浮点类型，即使结果是整数
- python中的Number类型是我们最常使用的一种，它包含了int、float、complex类型
- 如果需要整数类型的除法，使用"//"而不是"/"

一些数学计算的python代码

```
a = 10+20
b = a/5
c = a//5
d = 4+5j
e = 6+10j
```

输出

```
30 <class 'int'> 6.0 <class 'float'> 6 <class 'int'>
(4+5j) <class 'complex'> (6+10j) <class 'complex'> (10+15j) <class 'complex'>
```

注意事项

- python中除0操作是非法的，可以尝试一下除0会发生什么情况
- python中的变量在使用前需要先定义

4.交换两个变量值

试想一下，如果我们有一杯500ml橙汁和一杯600ml西瓜汁，如何交换两个杯子中的饮料呢?

答案很简单，那就是使用第三个杯子

交换两个数

```
cup1 = 500
cup2 = 600
print("Before exchange: ",cup1,cup2)
temp_cpu = cup1
cup1 = cup2
cup2 = temp_cpu
print("After exchange: ",cup1,cup2)
```

输出

```
Before exchange:  500 600
After exchange:  600 500
```

自己试着交换两个数

5.python中的格式化输出

以下是一些信息，尝试使用python将它打印出来：

```
Name      Type        Compiled language     ID
Python    Dynamic            False            1
Java      Static             False           10
C++       Static              True          100
```

使用f-string来进行格式化输出

python格式化输出代码

```
print(f"{'Name' : <10}{'Type' : <10}{'Compiled language' : ^20}{'ID' : >5}")
print(f"{'Python' : <10}{'Dynamic' : <10}{'False': ^20}{'1' : >5}")
print(f"{'Java' : <10}{'Static' : <10}{'False': ^20}{'10' : >5}")
print(f"{'C++' : <10}{'Static' : <10}{'True': ^20}{'100' : >5}")
```

说明

- 使用f作为字符串的开头来使用f-string
- 使用{}来输出变量
- 使用: < 来让前面的变量进行左对齐
- 使用: ^ 来让前面的变量进行居中对齐
- 使用: > 来让前面的变量进行右对齐

再来看一个例子

```
name = 'Tom'
age = 20
print(f'My name is {name},I\'m {age} years old,I like {"python"}')
```

输出

```
My name is Tom,I'm 20 years old,I like python
```

说明

- 字符串既可以使用双引号，也可以使用单引号
- 如果字符串内需要打印的符号和字符串符号相等，可以使用'\'进行转义
- f-string内的字符串可以使用与外部字符串不相等符号

尝试打印下面的例子

```
I will print these prime numbers: 10,11,12
ID   Number   isPrime
0     10           0
1     11           1
2     12           1
```

可能的python实现

```
prime1 = 10
prime2 = 11
prime3 = 12
print(f"I will print these prime numbers: {prime1},{prime2},{prime3}")
print(f"{'ID':<5}{'Number':^5}{'isPrime':>10}")
print(f"{0:<5}{prime1:^5}{False:>10}")
print(f"{1:<5}{prime2:^5}{True:>10}")
print(f"{2:<5}{prime3:^5}{True:>10}")
```

6.作业提交

- 安装git
- 申请github账户
- fork现有源码仓
- clone源码
- 完成作业
- 使用git提交作业
- 作业修改