

# Python开发入门

**NSD PYTHON1**

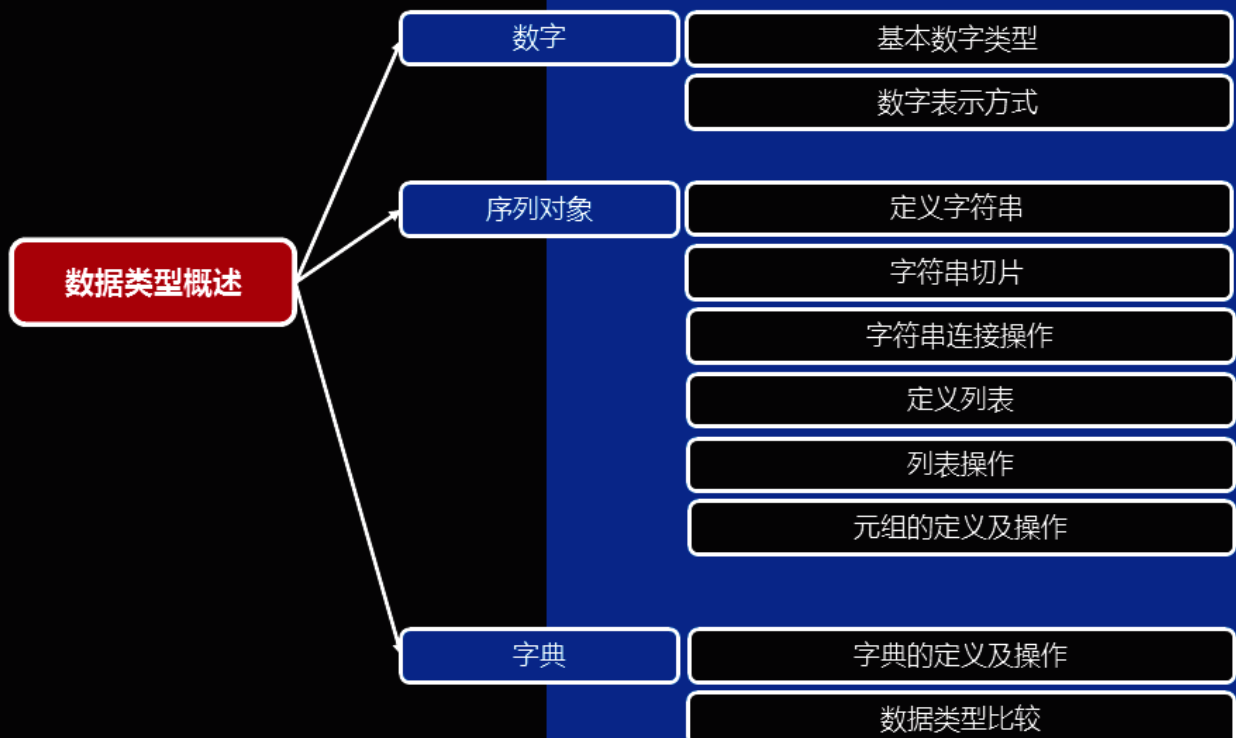
**DAY02**

# 内容

上午	09:00 ~ 09:30	作业讲解和回顾
	09:30 ~ 10:20	数据类型概述
	10:30 ~ 11:20	
	11:30 ~ 12:00	判断语句
下午	14:00 ~ 14:50	
	15:00 ~ 15:50	while循环
	16:10 ~ 17:00	
	17:10 ~ 18:00	总结和答疑



## 数据类型概述



# 数字

---

## 基本数字类型

知识讲解

- int : 有符号整数
- bool : 布尔值
  - True : 1
  - False : 0
- float : 浮点数
- complex : 复数



# 数字表示方式

知识讲解

- python默认以十进制数显示
- 数字以0o或0O开头表示为8进制数
- 数字以0x或0X开头表示16进制数
- 数字以0b或0B开头表示2进制数



# 字符串

---

## 定义字符串

知识讲解

- python中字符串被定义为引号之间的字符集合
- python支持使用成对的单引号或双引号
- 无论单引号，还是双引号，表示的意义相同
- python还支持三引号（三个连续的单引号或者双引号），可以用来包含特殊字符
- python不区分字符和字符串



## 字符串切片

知识讲解

- 使用索引运算符[ ]和切片运算符[ :]可得到子字符串
- 第一个字符的索引是0，最后一个字符的索引是-1
- 子字符串包含切片中的起始下标，但不包含结束下标

```
>>> py_str = 'python'
>>> py_str[0]
'p'
>>> py_str[-2]
'o'
>>> py_str[2:4]
'th'
>>> py_str[2:]
'thon'
>>> py_str[:4]
'Pyth'
```



## 字符串连接操作

- 使用+号可以将多个字符串拼接在一起
- 使用\*号可以将一个字符串重复多次

知识讲解

```
>>> py_str = 'python'
>>> is_cool = 'is Cool'
>>> print py_str + ' ' + is_cool
python is Cool
>>> py_str * 2
'pythonpython'
```



## 定义列表

- 可以将列表当成普通的“数组”，它能保存任意数量任意类型的python对象
- 像字符串一样，列表也支持下标和切片操作
- 列表中的项目可以改变

知识讲解

```
>>> alist = [1, "tom", 2, "alice"]
>>> alist[1] = 'bob'
>>> alist[2:]
```



# 列表操作

- 使用in或not in判断成员关系
- 使用append方法向列表中追加元素

知识讲解

```
>>> alist = [1, "tom", 2, "alice"]
>>> 'tom' in alist
True
>>> 'alice' not in alist
False
>>> alist.append(3)
>>> alist[5] = 'bob'
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
IndexError: list assignment index out of range
```



# 元组的定义及操作

- 可以认为元组是“静态”的列表
- 元组一旦定义，不能改变

知识讲解

```
>>> atuple = (1, "tom", 2, "alice")
>>> 'tom' in atuple
True
>>> atuple[0] = 3
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: 'tuple' object does not support item assignment
```



# 字典

## 字典的定义及操作

- 字典是由键-值(key-value)对构成的映射数据类型
- 通过键取值，不支持下标操作

知识讲解

```
>>> user_dict = {'name':'bob', 'age':23}
>>> use_dict['gender'] = 'male'
>>> 'bob' in user_dict
False
>>> 'name' in user_dict
True
>>> user_dict[0]
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
KeyError: 0
```





# 数据类型比较

知识讲解

- 按存储模型分类
  - 标量类型：数值、字符串
  - 容器类型：列表、元组、字典
- 按更新模型分类：
  - 可变类型：列表、字典
  - 不可变类型：数字、字符串、元组
- 按访问模型分类
  - 直接访问：数字
  - 顺序访问：字符串、列表、元组
  - 映射访问：字典



## 判断语句



# if语句

## if语句语法结构

- 标准if条件语句的语法

```
if expression:  
    if_suite  
else:  
    else_suite
```

- 如果表达式的值非0或者为布尔值True, 则代码组if\_suite被执行; 否则就去执行else\_suite
- 代码组是一个python术语, 它由一条或多条语句组成, 表示一个子代码块

# if语句示例解析

## 知识讲解

- 只要表达式数字为非零值即为True

```
>>> if 10:  
...     print('Yes')  
Yes
```

- 空字符串、空列表、空元组，空字典的值均为False

```
>>> if "":  
...     print('Yes')  
... else:  
...     print('No')  
No
```



## 案例1：判断合法用户

## 课堂练习

1. 创建login2.py文件
2. 提示用户输入用户名和密码
3. 获得到相关信息后，将其保存在变量中
4. 如果用户输的用户名为bob，密码为123456，则输出Login successful，否则输出Login inorrect



# 扩展if语句

## 扩展if语句结构

- 扩展if语句结构

```
if expression1:  
    if_suite  
elif expression2:  
    elif_suite  
else:  
    else_suite
```



## 条件表达式

### 知识讲解

- Python 在很长的一段时间里没有条件表达式( $C ? X : Y$ )，或称三元运算符，因为范·罗萨姆一直拒绝加入这样的功能
- 从Python 2.5集成的语法确定为： $X \text{ if } C \text{ else } Y$

```
>>> x, y = 3, 4
>>> smaller = x if x < y else y
>>> print smaller
3
```



## 案例2：编写判断成绩的程序

### 课堂练习

- 创建grade.py脚本，根据用户输入的成绩分档，要求如下：
  1. 如果成绩大于60分，输出“及格”
  2. 如果成绩大于70分，输出“良”
  3. 如果成绩大于80分，输出“好”
  4. 如果成绩大于90分，输出“优秀”
  5. 否则输出“你要努力了”



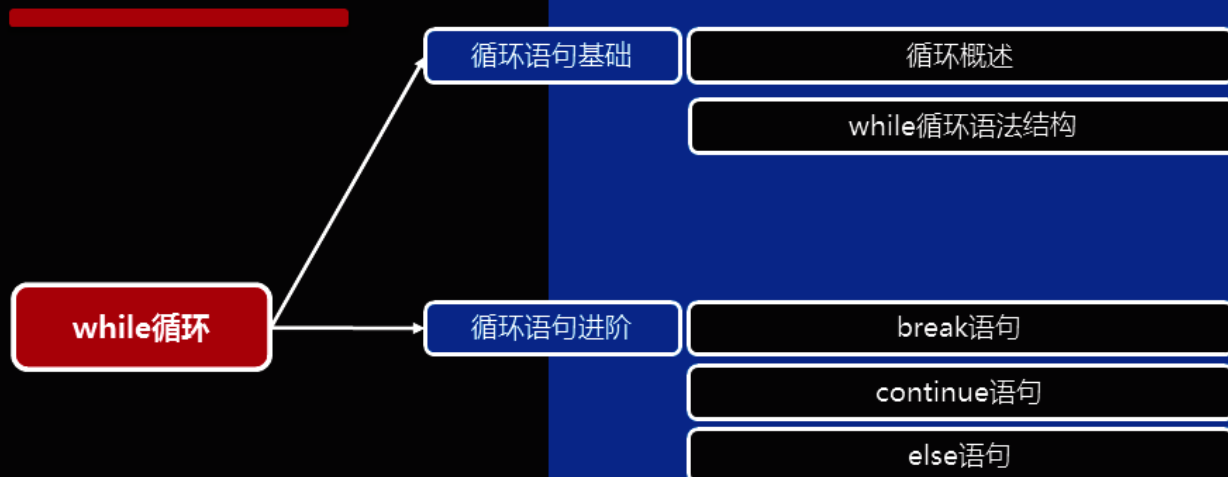
## 案例3：编写石头剪刀布小游戏

课堂练习

- 编写game.py，要求如下：
  1. 计算机随机出拳
  2. 玩家自己决定如何出拳
  3. 代码尽量简化



### while循环



# 循环语句基础

## 循环概述

- 一组被重复执行的语句称之为循环体，能否继续重复，决定循环的终止条件
- Python中的循环有while循环和for循环
- 循环次数未知的情况下，建议采用while循环
- 循环次数可以预知的情况下，建议采用for循环



## while循环语法结构

- 当需要语句不断的重复执行时，可以使用while循环

```
while expression:  
    while_suite
```

- 语句while\_suite会被连续不断的循环执行，直到表达式的值变成0或False

```
sum100 = 0  
counter = 1
```

```
while counter <= 100:  
    sum100 += counter  
    counter += 1  
print ("result is %d" % sum100)
```

知识讲解



## 循环语句进阶



# break语句

知识讲解

- break语句可以结束当前循环然后跳转到下条语句
- 写程序的时候，应尽量避免重复的代码，在这种情况下可以使用while-break结构

```
name = input('username: ')
while name != 'tom':
    name = input('username: ')
#可以替换为
while True:
    name = input('username: ')
    if name == 'tom':
        break
```



# continue语句

知识讲解

- 当遇到continue语句时，程序会终止当前循环，并忽略剩余的语句，然后回到循环的顶端
- 如果仍然满足循环条件，循环体内语句继续执行，否则退出循环

```
sum100 = 0
counter = 0
while counter <= 100:
    counter += 1
    if counter % 2:
        continue
    sum100 += counter
print ("result is %d" % sum100)
```



## else语句

知识讲解

- python中的while语句也支持else子句
- else子句只在循环完成后执行
- break语句也会跳过else块

```
sum10 = 0
```

```
i = 1
```

```
while i <= 10:
```

```
    sum10 += i
```

```
    i += 1
```

```
else:
```

```
    print (sum10)
```



## 案例4：完善石头剪刀布小游戏

- 编写game2.py，要求如下：
  1. 基于上节game.py程序
  2. 实现循环结构，要求游戏三局两胜

课堂练习



## 案例5：猜数程序

课堂练习

- 编写guess.py，要求如下：
  1. 系统随机生成100以内的数字
  2. 要求用户猜生成的数字是多少
  3. 最多猜5次，猜对结束程序
  4. 如果5次全部猜错，则输出正确结果



## 总结和答疑

