day04 数据类型（二）

今日内容

- 列表
- 元组

内容回顾和补充

1. 计算机基础
    - 硬件：CPU/内存/硬盘/主板/网卡
    - 操作系统：
        - linux（免费/开源）
            - centos
            - ubuntu
            - redhat
        - windows
        - mac
    - 解释器/编译器
        - 补充：编译型语言和解释型语言？

        ```
        # 编译型：代码写完后，编译器将其变成成另外一个文件，然后交给计算机执行。
        # 解释型：写完代码交给解释器，解释器会从上到下一行行代码执行：边解释边执行。 【实时翻译】
        ```

    - 软件（应用程序）
2. 环境的安装
    - python解释器
        - py2
        - py3
    - 开发工具：pycharm（推荐） / 文本编辑器
3. Python语法
    1. 解释器路径: hello.py

        ```
        #!/usr/bin/env python

        print('你好')
        ```

        linux系统应用：
        - 赋予文件可执行权限
        - ./hello.py

2. 编码

```
#!/usr/bin/env python
# -*- coding:utf-8 -*-

print('你好')
```

    1. 编码种类
        - acsii
        - unicode
        - utf-8 / utf-16
        - gbk/gb2312
    2. 中文表示
        - utf-8: 3字节
        - gbk: 2字节
    3. python默认解释器编码
        - py3： utf-8
        - py2： ascii
3. 输入输出
    - py2：
        - 输入：raw_input
        - 输出：print ""
    - py3:
        - 输入：input
        - 输出：print()
4. 数据类型
    1. int
        - py2中有：int/long ; py3中有 int 。
        - 强制转换：int("76"")
        - 除法：py2（多加一行代码） 和 py3（正常）
    2. bool
        - True/False （其他语言：true/false）
        - 特殊为False的其他类型： 0 和 ""
    3. str
        - 独有功能
            - upper/lower
            - replace
            - strip/lstrip/rstrip
            - isdigit
            - split / rsplit
            - 补充：
                - startswith / endswith

```
name = 'alex'

# 判断是否已al开头
"""
# 方式一:
flag = name.startswith('al')
print(flag)
"""
"""
# 方式二:
val = name[0:2]
if val == 'al':
    print('是以al开头')
else:
    print('不是')
"""
```

- format

```
name = "我叫{0},年龄:{1}".format('老男孩',73)
print(name)
```

- encode

```
name = '李杰' # 解释器读取到内存后，按照unicode编码存储：8个字节。
v1 = name.encode('utf-8')
print(v1)
v2 = name.encode('gbk')
print(v2)
```

- join

```
name = 'alex' # a_l_e_x
result = "**".join(name) # 循环每个元素，并在元素和元素之间加入连接符。
print(result)
```

- 公共功能
  - 索引，获取一个字符。
  - 切片，获取一段字符串（子序列）。
  - 步长

```
name = 'alex'

# val = name[0:-1:2]
# val = name[1:-1:2]
# val = name[1::2]
# val = name[::2]
# val = name[-1:0:-2]
# print(val)
# 笔试题：请将字符串反转。
val = name[::-1]
print(val)
```

- 长度，获取字符长度。
- for循环

```
name = 'alex'
for item in name:
    print(item)
```

```
name = 'alex'
for item in name:
    print(item)
    break
    print('123')
```

```
name = 'alex'
for item in name:
    print(item)
    continue
    print('123')
```

```
# 练习题

# 1. for循环打印 “alex” 的每个元素:  for > while

# 2. 请打印:  1 - 10
"""
 for i in range(1,11): # [1,2,3,4,5,6,7,8,9,10,11,12,14]
"12345678910"
     print(i)
"""

# 3. 请打印:  1 2 3 4 5 6 8 9 10
"""
for i in range(1,11):
    if i == 7:
        pass
    else:
```

```
        print(i)
"""
```

注意：for和while的应用场景：有穷尽优先使用for，无穷尽用while

5. 变量
6. 注释
7. 条件语句
8. 循环语句：while + for + break + continue
9. 运算符
10. 字符串格式化
    - %s
    - %d
    - %%
11. 其他
    - markdown笔记
    - git
        - 本地：git软件【常用命令】
            - git status
            - git add .
            - git commit -m ''提交记录'
            - git push origin master
        - 远程：码云 / github（程序员交友平台）
        - 面试相关：
            1. 写出你常用的git命令。
            2. 你们公司是怎么用git做开发的?
                1. 在码云或GitHub等代码托管的网站创建自己仓库，创建完之后码云会给我一个仓库地址，如：https://gitee.com/old_boy_python_stack_21/190326032.git
                2. 自己写代码.....
                3. 将代码提交到远程 仓库。
                    - 初始化
                        - 进入一个任意文件夹，如：D:\homework\
                        - git init
                        - git config 邮箱
                        - git config 姓名
                        - git remote add origin https://gitee.com/old_boy_python_stack_21/190326032.git

                          注意：至此git已经将 D:\homework\目录管理起来，以后此文件夹有任何变化，git都会检测到（使用 git status 命令可以查看状态）
                    - 代码收集并提交
                        - git status

        - git add .
        - git commit -m "记录"
        - git push origin master 将本地D:\homework\目录下的内容同步到 码云仓库。
    - 修改代码或删除文件等对本地 D:\homework\ 下任何文件做操作。
        - git status
        - git add .
        - git commit -m "记录"
        - git push origin master 将本地D:\homework\目录下的内容同步到 码云仓库。
    - 【避免】如果远程有本地没有的代码，必须先执行：【可能引发合并问题】
        - git pull origin master
        - git status
        - git add .
        - git commit -m "记录"
        - git push origin master 将本地D:\homework\目录下的内容同步到 码云仓库。

12. 总结
    1. 语法：变量/if/while/运算符（辅助）
        1. 必备：变量/if/while/
        2. 重点：数据类型中的字符串
            1. 独有功能
            2. 公共功能
            3. for
    2. 解决实际问题：
        - 逻辑 + 代码

内容详细

1. 列表

> 如果想要表示两个同学 users = "李邵，李奇航...".
>
> 以后想要表示多个"事物"，可以使用列表。

```
users = ["李邵奇","奇航",99]
```

- 公共功能
    - len

```
users = ["李邵奇","奇航",99]
val = len(users)
print(val) # 3
```

- 索引

```
users = ["李邵奇","奇航",99]
val = users[0]
print(val)
```

- 切片

```
users = ["李邵奇","奇航",99]

val = users[0:2]
```

- 删除（数字/布尔/字符串除外）

```
users = ["李邵奇","奇航",99]
# 方式一
users.pop(1)
print(users)

# 方式二:
del users[1]
print(users)
```

注意:

  - 字符串本身不能修改或删除【不可变类型】 v1 = "alex".upper()
  - 列表是可变类型。

- 修改（字符串/数字/布尔除外）

```
users = ["李邵奇","奇航",99]
users[2] = 66

users[0] = '李杰'
users[0][1]
```

- 步长

```
users = ["李邵奇","奇航",99]

val = users[0:2:2]
```

- 练习题

```
"""
实现一个整数加法计算器(两个数相加):

如: content = input("请输入内容:") 用户输入: 5+9或5+ 9或5 + 9 (含空白) , 然后进行分割转换最终进行整数的计算得到结果。
"""
```

```
# 思路一:
"""
content = input('请输入: ') # [5+9]  或 [5 +9]  或者 [ 5 + 9 ]
content = content.strip() # [5+9]  或 [5 +9]  或者 [5 + 9]
v1 = int(content[0])
v2 = int(content[-1])
v3 = v1 + v2
"""

# 思路二:
"""
content = input('请输入: ') # [5+9]  或 [5 +9]  或者 [ 5 + 9 ]
content_len = len(content)
index = 0
total = 0
while True:
    char = content[index]
    if char.isdigit():
        total += int(char)
    index += 1
    if index == content_len:
        break
print(total)
"""

# 思路三:
"""
content = input('请输入: ') # [5+9]  或 [5 +9]  或者 [ 5 + 9 ]
result = content.split('+')
# print(result) # ['55 ', ' 99 ']
v1 = int(result[0]) # "55"
v2 = int(result[1]) # " 99 "
v3 = v1 + v2
print(v3)
"""
```

- for循环

```
"""
users = ['李邵奇','利奇航','张三丰','李子森']
for i in users:
    print(i)
"""
"""
users = ['李邵奇','利奇航','张三丰','李子森']
for i in users:
    # 第一次循环: i="李邵奇"
    print(i)
    for ele in i:
        print(ele)
"""
```

```
# 练习题：请通过for循环和数字计数器实现：users = ['李邵奇','利奇航','张三丰','李子森']
"""
    0 李邵奇
    1 利奇航
    2 张三丰
    3 李子森
"""
"""
# 方式一
users = ['李邵奇','利奇航','张三丰','李子森']
count = 0
for i in users:
    print(count,i)
    count += 1
"""
"""
# 方式二
users = ['李邵奇','利奇航','张三丰','李子森']
users_len = len(users) # 4
for index in range(0,users_len): # [0,1,2,3]
    print(index,users[index])
"""
```

- 独有功能
  - append，在列表的最后追加一个元素

```
users = []
users.append('alex')
print(users)
```

```
"""
示例一：
users = []
while True:
    name = input('请输入姓名:')
    users.append(name)
    print(users)
"""
"""
示例二：
# 录入用户和密码
users = []
for i in range(0,3):
    name = input('请输入用户名和密码:')
    users.append(name)
print(users) # ['alex,123', 'oldboy,888', 'lishaoqi,123']

# 用户和密码校验
username = input('请输入要登陆用户名：')
password = input('请输入要登陆密码：')
for item in users:
```

```
    result = item.split(',') # ['alex','123']
    user = result[0]
    pwd = result[1]
    if user == username and pwd == password:
        print('登陆成功')
        break

"""
```

  - insert
  - remove
  - pop
  - clear
- 总结:
  - 增:
    - append / insert
  - 删:
    - remove / pop / clear / del users[2]
  - 改:
    - users[3] = "新值"
  - 查:
    - 索引/切片
  - 列表嵌套

```
users = ["alex",0,True,[11,22,33,"老男孩"],[1,['alex','oldboy'],2,3]]

users[0]
users[2]
users[0][2]
users[3] # [11,22,33,"老男孩"]
users[3][-1] # "老男孩"
users[3][-1][1] # '男'
users[3] = 666
```

2. 元组

1. 元组书写规范

```
users = [11,22,33,"老男孩"] # 列表（可变）

users = (11,22,33,"老男孩") # 元组（不可变）
```

2. 公共功能
    1. 索引（排除：int/bool）

```
users = (11,22,33,"老男孩")

print(users[0])
print(users[-1])
```

2. 切片（排除：int/bool）

```
users = (11,22,33,"老男孩")
print(users[0:2])
```

3. 步长（排除：int/bool）

```
users = (11,22,33,"老男孩")
print(users[0:2:2])
```

4. 删除（排除：tuple/str/int/bool）
5. 修改（排除：tuple/str/int/bool）
6. for循环（排除：int/bool）

```
users = (11,22,33,"老男孩")
for item in users:
    print(item)
```

7. len（排除：int/bool）

```
users = (11,22,33,"老男孩")
print(len(users))
```

3. 独有功能（无）
4. 特殊：元组中的元素(儿子)不可被修改/删除。

```
# 示例一：
v1 = (11,22,33)
v1[1] = 999 # 错误
v1 = 999  # 正确

# 示例二：可以嵌套
v1 = (11,22,33,(44,55,66),(11,2,(99,88,),3))

# 示例三：嵌套
v2 = [11,22,33,(11,22,33)]
v2[-1][1] = 99 # 错误
v2[-1] = 123 # 正确

# 示例四：嵌套
v3 = (11,[1,2,3],22,33)
v3[1] = 666 # 错误
v3[1][2] = 123
```

总结

1. 解释型语言和编译型区别以及列举你了解的语言?
2. 字符串补充功能
    - 独有
        - startswith/endswith
        - format
        - encode
        - join
    - 公共
        - 切片
        - 索引
        - len
        - 步长（面试题）
        - for循环
        - range(0,10) # 帮助你生成一个数字列表 [0,1,2,3,4,5,6,7,8,9]
    - 特性:
        - 不可变，所以字符串元素不能删除和修改。
3. git本地和远程要同步，以后只能操作本地然后提交。
4. 列表（可变）
    - 公共
        - 索引
        - 切片
        - 步长
        - 修改
        - 删除 del
        - for
        - len
    - 独有
        - append
        - insert
        - pop
        - remove
        - clear
    - 列表嵌套
5. 元组（不可变）
    - 公共
        - 索引
        - 切片
        - 步长
        - for
        - len

- 独有功能（无）
- 元组嵌套

```
v3 = (11,[1,2,3],22,33)
v3[1] = 666 # 错误
v3[1][2] = 123
```