一 课程回顾

```
# 2.将列表["h","e","l","l","o",",","w","o","r","l","d"] 转为字符串 hello,world
# 4.
'''
 输入一个字符串统计每个元音字母（aeiou）在字符串中出现的次数。
 对于结果输出5行，格式如下：
 a:num1（a的个数）
 e:num2（b的个数）
 i:num3（i的个数）
 o:num4（o的个数）
 u:num5（u的个数）
 例如输入aeioubbbccc，输出：
 a:1
 e:1
 i:1
 o:1
 u:1
'''

s = "Hello, World!"

1.给定一个字符串s，如何获取字符串中第一个字符？
2.给定一个字符串s，如何获取字符串中倒数第二个字符？
3.给定一个字符串s，如何获取字符串中前三个字符的子串？
4.给定一个字符串s，如何获取字符串中除了最后一个字符以外的所有字符？
5.给定一个字符串s，如何获取字符串中从第二个字符开始到倒数第四个字符为止的子串？
6.给定一个字符串s，如何获取字符串中每隔两个字符取一个字符的子串？
7.给定一个字符串s，如何获取字符串中所有奇数索引的字符的子串？
8.给定一个字符串s，如何将字符串倒序输出？
9.给定一个字符串s，如何获取字符串中最后五个字符的子串？
10.给定一个字符串s，如何获取字符串中倒数第四个字符到末尾的子串？



# list1 = ["h","e","l","l","o",",","w","o","r","l","d"]
# s = ''.join(list1)
# print(s)

# s = input('请输入一个字符串:')
# print('a:',s.count('a'))
# print('e:',s.count('e'))
# print('i:',s.count('i'))
# print('o:',s.count('o'))
# print('u:',s.count('u'))
# for i in 'aeiou':
#     print(f'{i}:{s.count(i)}次')




# 1.给定一个字符串s，如何获取字符串中第一个字符？
```

```
# s = "Hello, World!"
# s1 = s[0]
# print(s1)
# print(s[0])
# 2.给定一个字符串s，如何获取字符串中倒数第二个字符？
# s = "Hello, World!"
# print(s[-2])
# 3.给定一个字符串s，如何获取字符串中前三个字符的子串？
# s = "Hello, World!" "asdasilodjfials!@#$%^&*()_+"
# print(s[:3])
# 4.给定一个字符串s，如何获取字符串中除了最后一个字符以外的所有字符？
# s = "Hello, World!"
# print(s[:-1])
# 5.给定一个字符串s，如何获取字符串中从第二个字符开始到倒数第四个字符为止的子串？
# s = "Hello, World!"
# print(s[1:-3])
# 6.给定一个字符串s，如何获取字符串中每隔两个字符取一个字符的子串？
# s = "Hello, World!"
# print(s[::3])
# 7.给定一个字符串s，如何获取字符串中所有奇数索引的字符的子串？
# s = "Hello, World!"
# print(s[1::2])
# 8.给定一个字符串s，如何将字符串倒序输出？
# s = "Hello, World!"
# print(s[::-1])
# 9.给定一个字符串s，如何获取字符串中最后五个字符的子串？
# s = "Hello, World!"
# print(s[-5:])
# 10.给定一个字符串s，如何获取字符串中倒数第四个字符到末尾的子串？
# s = "Hello, World!"
# print(s[-4:])
```

二 上节课没有讲完的知识点

2.1 删

```
pop()：删除一个值，默认从最后一个开始删，也可以指定位置
语法：    列表名.pop()
          列表名.pop(下标)

list1 = [1,2,3,4,5,6,7,8,9]
list1.pop(-1)
print(list1)

[1, 2, 3, 4, 5, 6, 7, 8]

remove():删除一个指定的值，如果有多个，从第一个开始删
语法：    列表名.remove(删除对象)
list1.remove(3)
print(list1)

[1, 2, 4, 5, 6, 7, 8]

clear():清空列表里面的所有数据
```

```
语法:    列表名.clear()
list1.clear()
print(list1)

[]

del: 全局删除，可以删除一个变量
语法:    del  列表名[下标]
del list1[3]
print(list1)

[1, 2, 4, 6, 7, 8]
```

2.2 改

```
单个修改: 直接通过下标进行修改
语法:    列表名[下标] = 内容
list1[1] = '久违'
print(list1)

[1, '久违', 4, 5, 6, 7, 8]

多个修改: 通过切片的方式进行修改
语法:    列表名[起点:终点] = 数据1，数据2，数据n
list1[1:3] = 70,20
print(list1)

[1, 70, 20, 5, 6, 7, 8]
```

2.3 查

```
index():根据内容获取指定数据的下标
语法:    列表名.index(要找的内容)
print(list1.index(8))

6
列表名.index(要找的内容，起点值)
print(list1.index(5,5))  # 报错 ，显示5不在列表中

count:统计数据出现的次数
语法:    列表名.count(要找的内容)
print(list1.count(5))

1
```

排序

```
排序(全是int的列表才可以排序)
sort：让列表的内容按照降序/升序的方式来排序
    列表名.sore() ==》 升序
li1 = [1,564,5,1541,6568,464,115,31,53468,4865,4135]
li1.sort()
```

```
print(li1)

[1, 5, 31, 115, 464, 564, 1541, 4135, 4865, 6568, 53468]

    列表名.sort(reverse=True)  ==> 降序
li1 = [1,564,5,1541,6568,464,115,31,53468,4865,4135]
li1.sort(reverse=True)
print(li1)

[53468, 6568, 4865, 4135, 1541, 564, 464, 115, 31, 5, 1]
```

2.4 列表生成式

```
list1 = [1,2,3,4,5,6,7,8,9]
print(list1)

print(list(range(1,10)))
[1, 2, 3, 4, 5, 6, 7, 8, 9]
```

2.4.1 普通的列表生成式

```
# 普通写法
li = []
for i in range(1,10):
    li.append(i)
print(li)

# 列表生成式写法
print([i for i in range(1,10)])
```

```
[1*1,2*2,3*3,....9*9]
# 普通写法
li = []
for i in range(1,10):
    li.append(i*i)
print(li)

# 列表生成式
print([i*i for i in range(1,10)])
```

```
[1*2,2*3,3*4,...9*10]
# 普通写法
li = []
for i in range(1,10):
    li.append(i*(i+1))
print(li)

print([i*(i+1) for i in range(1,10)])
```

2.4.2 带if的列表生成式

```
# 求偶数，普通写法
li = []
for i in range(1,11):
    if i % 2 == 0:
        li.append(i)
print(li)

# 带if的列表生成式写法
print([i for i in range(1,11) if i % 2 == 0])
```

使用两层循环，生成一切组合可能(ABC和123)

```
# 普通写法
li = []
for i in 'ABC':
    for j in '123':
        li.append(i+j)
print(li)

# 列表生成式
print([i+j for i in 'ABC' for j in '123'])
```

2.5 元组

元组通常使用一对小括号将所有元素包围起来，但是小括号不是必须的

```
t = '左手',
print(t)
print(type(t))
```

2.5.1 查

```
count():统计某个元素在元祖中出现的次数
t = (123,'python','JD','TaoBao',123)
print(t.count(123))
print(t.count('JD'))

index:检索某个元素的下标值
t = (123,'python','JD','TaoBao',123)
print(t.index(123))
print(t.index(123,1))
```

2.5.2 功能

```
相加，两个元祖相加生成一个新的元祖
data = ('飞最帅','小白')+('左手','火影')
print(data)

想乘
```

```
data = ('飞最帅','小白')*2
print(data)

获取长度  len()==> 获取长度方法
data = ('飞最帅','小白')
print(len(data))

索引
data = ('飞最帅','小白','久违','子轩','张三')
print(data[0])
print(data[1])
print(data[2])
print(data[3])

切片
data = ('飞最帅','小白','久违','子轩','张三')
print(data[0:3])

步长
data = ('飞最帅','小白','久违','子轩','张三')
print(data[0:4:2])

for循环
data = ('飞最帅','小白','久违','子轩','张三')
for i in data:
    if i == '久违':
        continue
    print(i)
```

2.6 字符串格式化输出

```
你好某某某  我叫某某某  再见某某某

name = '左手'
age = 18
print('大家好，我叫'+name,'我今年'+str(age)+'岁')
```

2.6.1 %方法(占位符)

```
%s = 字符串 ==》 只能放字符串
%d = 整数 ==》 只能放整数
%f = 小数 ==》 默认保留6为小数点
    %.1f == 》保留一位小数
    %.2F == 》 保留两位小数
```

```
语法：("xx%dxx%s"%(变量1，变量2))
name = '左手'
age = 18
height = 175.0
print('你好，我叫%s,今年%d,身高%f'%(name,age,height))
print('你好，我叫%s,今年%d,身高%.1f'%(name,age,height))
print('你好，我叫%s,今年%d,身高%.2f'%(name,age,height))
# 也可以直接放数据
print('你好，我叫%s,今年%d,身高%f'%('左手',18,175.0))

s = '我叫%s,今年%d岁了,身高%.1f'
print(s%('左手',18,175.0))
```

2.6.2 format()方法

```
name = '左手'
age = 18
height = 175.0
print('你好，我叫{},今年{}岁了,身高{}'.format(name,age,height))

传入的数据类型不限，字符串，元祖，列表都行
数据跟{}顺序从左往右一一对应
直接传入数据
    "{}{}{}".format(数据1，数据2，数据3)
print('你好，我叫{},今年{}岁了,身高{}'.format('左手',18,175.0))
自定义数据：
    "{下标}{下标}".format(数据1，数据2)
print('你好，我叫{0},今年{2}岁了,身高{1}'.format('左手',175.0,18))
```

2.6.3 f-format() 方法

```
语法:在字符串前面加上一个F/f,把要输出的变量用大括号进行包裹
name = '左手'
age = 18
height = 175.0
print(f'大家好，我叫{name},我今年{age}岁，今年身高是{height}')
print(f'大家好，我叫{name[0]},我今年{age+1}岁，今年身高是{height+10}')
```

三 数据类型进阶(二)

1.数值类型 : int,float,bool ==> 存储一个数值

2.序列类型 : str,list,tuple ==> 存储多个数据

3.散列类型

无序，内部元素不重复，没有下标

3.1 字典 dict

用来保存一些有典型的对应关系的数据类型，特点是使用键值对的方式来存储数据

key ==> 键 (拼音，偏旁)， value ==> 值（查到的那个字）

键值对的语法: key:value

表现形式:

{}，大括号里面如果存储的是键值对，他才是字典类型

```
list1 = ['左手','1857102433541','久违','14534154135']
dict1 = {'左手':'153153','久违':'1531356141'}
print(dict1['左手'])
print(dict1['久违'])
```

通过键取值

字典名[键]

字典的key是不可以重复的，如果重复则取最后一个重复键的值

```
dict1 = {'a':1,'b':2,'a':3,'a':4}
print(dict1)
```

可变性

```
dict1 = {'a':1,'b':2,'a':3,'a':4}
dict1['b'] = 100
print(dict1)
```

```
字典包含多个键值对，key是字典的关键数据，程序对字典的操作都是基于key
- 通过key访问value
- 通过key添加key-value对
- 通过key删除key-value对
- 通过key修改key-value对
- 通过key判断指定key-value对是否存在
使用的也是中括号语法，在中括号内放的是key，而不是索引也不是value
```

如果要为字典添加key-value对，就只需要为不存在key赋值

```
a = {'语文':89}
a['数学'] = 93
a[92] = 85
print(a)
```

如果要删除字典中的键值对，就可以使用del语句

```
a = {'语文':89}
a['数学'] = 93
a[92] = 85
del a['语文']   # del  字典名[要删除的内容]
del a['数学']
print(a)
```

对存在的键值对赋值，新的值会覆盖原来的值

```
a = {'BMW':88,'benchi':83,'aodi':95}
a['benchi'] = 4.3
a['BMW'] = 3.8
print(a)
```

判断字典是否包含指定的key,使用in或者not in

```
a = {'BMW':88,'benchi':83,'aodi':95}
print('tesila' in a)
print('baoshijie' in a)
print('BYD' not in a)
```

作业

```
# 1.给定两个字符串，使用拼接符将它们连接起来。
str_1 = "Hello"
str_2 = "World"

# 2.使用占位符将一个姓名和年龄拼接成一句话。
name = "小明"
age = 18

# 3.使用 join 方法将一组字符串按照指定的字符进行连接。(I-love-Python)
words = ["I", "love", "Python"]

# 4.使用 format 方法将三个字符串按照指定的顺序进行连接。(他真帅是吗？)
str_1 = "他"
str_2 = "真帅"
str_3 = "是吗？"


# 5.使用 format 方法根据下标位置将三个字符串进行连接。(他真帅是吗？)
str_1 = "他"
str_2 = "真帅"
str_3 = "是吗？"

# 6.使用 format 方法通过给变量重新命名进行字符串连接。(他真帅是吗？)
str_1 = "他"
str_2 = "真帅"
str_3 = "是吗？"


# 7.使用 f-string 直接将三个字符串进行连接。
str_1 = "他"
str_2 = "真帅"
str_3 = "是吗？"



# 11.将一个字符串中所有的字符转换为大写。
data_string = "hello world"
```

```
# 13
"""
假设有一个字符串 "Hello, my name is John. I am 25 years old."，请完成以下操作(years old--岁的意思)：  (不能直接使用下标  要用字符串的方法来写)

    1.使用索引和切片获取并输出该字符串中的姓名（John）。
    data_string = "Hello, my name is John. I am 25 years old."

    2.使用整数计算获取并输出该字符串中年龄的两倍。
    data_string = "Hello, my name is John. I am 25 years old."

    3.将该字符串中的名字（John）替换为另一个名字，并输出替换后的字符串。
    data_string = "Hello, my name is John. I am 25 years old."

    4.使用字符串的格式化方法，将该字符串中的年龄替换为当前年份（2023）减去出生年份（出生年份为1980），并输出替换后的字符串。
    birth_year = 1980  # 当前年份
    current_year = 2023 # 出生年份
    data_string = "Hello, my name is John. I am 1980"

"""
```