字典

- 目标
  - 字典的应用场景
  - 创建字典的语法
  - 字典常见操作
  - 字典的循环遍历

1、字典的应用场景

思考1： 如果有多个数据，例如：`正心`，`男`，`20`，如何快速存储?

答：列表

```
list1 = ['正心', '男', 20]
```

思考2：如何查找到数据`正心`？

答：查找到下标为0的数据即可。

```
list1[0]
```

思考3：如果将来数据顺序发生变化，如下所示，还能用`list1[0]`访问到数据`正心`吗？。

```
list1 = ['男', 20, '正心']
```

答：不能，数据`正心`此时下标为2。

思考4：数据顺序发生变化，每个数据的下标也会随之变化，如何保证数据顺序变化前后能使用同一的标准查找数据呢?

答：字典，字典里面的数据是以`键值对`形式出现，字典数据和数据顺序没有关系，即字典不支持下标，后期无论数据如何变化，只需要按照对应的键的名字查找数据即可。

> 定义: 字典是一种可变的、无序的、键值对的、复杂的数据容器

2、创建字典的语法

字典特点：

- 符号为`{}`
- 数据为`键值对`形式出现
- 各个键值对之间用`逗号`隔开

```
"""字典的创建"""

"""
{}     大括号是字典创建的符号
<class 'dict'>  字典类型
```

```
字典中数据形式是键值对的形式，即 key: value 的形式
    key: value 键值对之间用英文形式的冒号隔开
    key   -->
        冒号左边是字典的键，
        在一个字典中键必须唯一：键可以是字符串/数值/元组/布尔类型，不能是列表
        取值是根据键取对应的值
    value  -->
        冒号右边是键的值，可以是Python任意类型
"""
dict1 = {'name': '正心', 520: True, (1,): 18}
print(dict1['name'])
print(type(dict1))

# 空字典
dict2 = {}
dict3 = dict()
```

> 注意：一般称冒号前面的为键(key)，简称k；冒号后面的为值(value)，简称v。

3、字典常见操作

3、1 增

- 写法：`字典序列[key] = 值`

  > 注意：如果key存在则修改这个key对应的值；如果key不存在则新增此键值对。

  ```
  dict1 = {
      'name': '正心',
      'age': 18,
      'gender': '男'
  }

  # 如果key不存在则新增此键值对
  dict1['爱好'] = '吃喝玩乐'
  print(dict1)
  # 结果: {'name': '正心', 'age': 18, 'gender': '男', '爱好': '吃喝玩乐'}

  # 如果key存在则修改这个key对应的值
  dict1['gender'] = '女'
  print(dict1)
  # 结果: {'name': '正心', 'age': 18, 'gender': '女', '爱好': '吃喝玩乐'}
  ```

  > 注意：字典为可变类型

- `update`：更新/合并字典

```
"""合并字典"""
dict2 = {'name': '正心', 'age': 18, 'gender': '男', '爱好': '吃喝玩乐'}
dict3 = {'学校': '青灯教育', '爱好': ['吃', '喝', '玩', '乐']}

dict2.update(dict3)
print(dict2)
# 结果: {'name': '正心', 'age': 18, 'gender': '男', '爱好': ['吃', '喝', '玩',
'乐'], '学校': '青灯教育'}
```

3、2 删

- del() / del：指定键, 删除字典中键值对

```
dict1 = {
    'name': '正心',
    'age': 18,
    'gender': '男'
}

# 指定键, 删除字典中键值对
del(dict1['gender'])
print(dict1)
# 结果: {'name': '正心', 'age': 18}
```

3、3 改

写法：`字典序列[key] = 值`

> 注意：如果key存在则修改这个key对应的值； 如果key不存在则新增此键值对。

3、4 查

3、4、1 key值查找

```
dict1 = {
    'name': '正心',
    'age': 18,
    'gender': '男'
}
print(dict1['name'])  # 正心
print(dict1['id'])  # 报错
```

> 如果当前查找的key存在，则返回对应的值；否则则报错。

3、4、2 get()

- 语法

```
字典序列.get(key, 默认值)
```

> 注意：如果当前查找的key不存在则返回第二个参数(默认值)，如果省略第二个参数，则返回None。

- 快速体验

```
dict1 = {
    'name': '正心',
    'age': 18,
    'gender': '男'
}
print(dict1.get('name'))  # 正心
print(dict1.get('id', 110))  # 110
print(dict1.get('id'))  # None
```

3、4、3 keys()

```
dict1 = {
    'name': '正心',
    'age': 18,
    'gender': '男'
}
print(dict1.keys())  # dict_keys(['name', 'age', 'gender'])
```

3、4、4 values()

```
dict1 = {
    'name': '正心',
    'age': 18,
    'gender': '男'
}
print(dict1.values())  # dict_values(['正心', 18, '男'])
```

3、4、5 items()

```
dict1 = {
    'name': '正心',
    'age': 18,
    'gender': '男'
}
print(dict1.items())  # dict_items([('name', '正心'), ('age', 18), ('gender',
'男')])
```

4、字典的循环遍历

4、1 遍历字典的key

```
dict1 = {
    'name': '正心',
    'age': 18,
    'gender': '男'
}

for key in dict1.keys():
    print(key)
```

```
D:\Anaconda3\python.exe "Z:/01 青灯教育课程/01 青灯教育-vip课程/
name
age
gender

Process finished with exit code 0
```

4、2 遍历字典的value

```
dict1 = {
    'name': '正心',
    'age': 18,
    'gender': '男'
}
for value in dict1.values():
    print(value)
```

```
D:\Anaconda3\python.exe "Z:/01 青灯教育课程/01 青灯教育-vip课程/
正心
18
男

Process finished with exit code 0
```

4、3 遍历字典的元素

```
dict1 = {
    'name': '正心',
    'age': 18,
    'gender': '男'
}
for item in dict1.items():
    print(item)
```

```
D:\Anaconda3\python.exe "Z:/01 青灯教育课程/01 青灯教育-vip课程/
('name', '正心')
('age', 18)
('gender', '男')

Process finished with exit code 0
```

4、4 解包键值对

```
dict1 = {
    'name': '正心',
    'age': 18,
    'gender': '男'
}
for key, value in dict1.items():
    print(key, value)
```

```
D:\Anaconda3\python.exe "Z:/01 青灯教育课程/01 青灯教育-vip课程
name 正心
age 18
gender 男

Process finished with exit code 0
```

5、总结

- 定义字典

```
dict1 = {'name': 'Python', 'age': 30}

dict2 = {}

dict3 = dict()
```

- 常见操作

    - 增/改

```
字典序列[key] = 值
```

- 查找
    - 字典序列[key]
    - keys()
    - values()
    - items()