问答题

1. 请给出以下解析式的结果，并解释该解析式的含义

```
{key: value for key in 'ABC'for value in range(3)}
{x: x*x for x in range(10) if x % 2 == 0}
{x ** 2 for x in [1, 1, 2]}
```

运行结果：

```
{'A': 2, 'B': 2, 'C': 2}
{0: 0, 2: 4, 4: 16, 6: 36, 8: 64}
{1, 4}
```

解释：

1. 对于第一条解析式，生成的字典的键值分别为A, B, C. 而当生成键值A时，按照0，1，2的顺序给键A赋值，因而最后留下的值为2。其他两个键值生成同理。从而，生成的字典为

```
{'A': 2, 'B': 2, 'C': 2}
```

2. 生成的字典的键值是10以内能够被2整除的数，其对应的值为x^2。
3. 生成集合。由于集合中的元素不可重复，即使列表中有两个元素1，最终得到的集合只有一个1和一个4

上机题

1. 基于字典的通讯录 保存已有好友通讯录信息，并通过字符串提示用户对好友通讯录信息进行增删改查操作，输入数字 1 进行好友添加，输入数字 2 删除好友，输入数字 3 和 4 分别进行好友信息修改和查询，接着根据用户选择的处理方式来进行针对性的好友信息管理。

源代码：

```
# 基于字典的通讯录
address_book = {"小明": {'tel': '001', 'add': "广州"}, "小红": {'tel': '002', 'add': "上海"},
                "小王": {'tel': '003', 'add': "北京"}}
while True:
    print('\n1：好友添加', '2：好友删除', '3：好友信息修改', '4：好友信息查询', '0：退出', sep='\n')
    op = input('请输入操作代码：')

    if not op.isdigit():
        print('输入错误！程序退出！')
        break

    op = eval(op)
    if op == 1:
        name = input('输入新好友姓名：')
        tel = input('输入新好友电话：')
        add = input('输入新好友地址：')
        cache = {name: {'tel': tel, 'add': add}}
```

```
        address_book.update(cache)
        print('操作成功！')

    elif op == 2:
        delete = input('输入要删除的好友姓名：')
        del address_book[delete]

    elif op == 3:
        name = input('输入要修改的好友姓名：')
        tel = input('输入修改的电话：')
        add = input('输入修改的地址：')
        cache = {name: {'tel': tel, 'add': add}}
        address_book.update(cache)

    elif op == 4:
        search = input('输入要查找的好友姓名：')
        if search in address_book:
            print('姓名：', search)
            for key, value in address_book[search].items():
                print(key, value)
        else:
            print('查无此人！')

    elif op == 0:
        break

print("再见")
```

运行结果：

```
1：好友添加
2：好友删除
3：好友信息修改
4：好友信息查询
0：退出
请输入操作代码：1
输入新好友姓名：肥陀
输入新好友电话：666
输入新好友地址：伏龙树底
操作成功！

1：好友添加
2：好友删除
3：好友信息修改
4：好友信息查询
0：退出
请输入操作代码：4
输入要查找的好友姓名：肥陀
姓名： 肥陀
tel 666
add 伏龙树底

1：好友添加
2：好友删除
3：好友信息修改
4：好友信息查询
0：退出
```

```
请输入操作代码：2
输入要删除的好友姓名：小红

1：好友添加
2：好友删除
3：好友信息修改
4：好友信息查询
0：退出
请输入操作代码：4
输入要查找的好友姓名：小红
查无此人！
```

2. 删除字符串中出现次数最少的字符，若多个字符出现次数一样，则都删除。输出删除这些字符以后的字符串，字符串中其他字符保持原来的顺序，请用字典字符串实现。

   输入：字符串只包含小写英文字母，不考虑非法输入，输入的字符串长度小于或 等于 20 字节（如：abcdd）

   输出：删除字符串中出现次数最少的字符后的字符串（如：dd）

源代码：

```python
str_in = input('输入小写的英文字母（不多于20个字母）：')

# 统计字母出现次数
lstr = list(str_in)
dic = {}
for e in lstr:
    if e not in dic:
        dic[e] = 1
    else:
        dic[e] += 1
dic_copy = dic.copy()
k0 = lstr[0]
mini = dic_copy.pop(k0)

# 找出现次数最少的字符
for value in dic.values():
    if value < mini:
        mini = value

lstr_select = lstr.copy()
for key, value in dic.items():
    if value == mini:
        lstr_select.remove(key)

# 输出
for i in range(len(lstr_select)):
    print(lstr_select[i], end='')

```

运行结果：

```
输入小写的英文字母（不多于20个字母）：abcdd
dd
```

探索上机题

1. 学习 enumerate 函数，然后使用 enumerate 函数来创建一个原始柱状图， 其中每个条由星号（*）组成，条的长度与列表对应元素值成正比例。结果如图所示：

```
Creating a bar chart from numbers:
Index   value       Bar
    0      19       ********************
    1       3       ***
    2      15       ***************
    3       7       *******
    4      11       ***********
```

源代码：

```
l = [19, 3, 15, 7, 11]
print('Creating a bar chart from numbers:')
print('Index\tValue\tBar')
for i, e in enumerate(l):
    print(i, '\t\t', e, '\t\t', '*'*e)
```

运行结果：

```
Creating a bar chart from numbers:
Index   Value   Bar
0       19          ********************
1       3       ***
2       15          ***************
3       7       *******
4       11          ***********
```