实验1

年级	20 级	专业（方向）	软件工程
学号	20337270	姓名	钟海财
电话	13397996670	Email	2940599563@qq.com

Task 1 ★

给定一个字符串并判断这个字符串是否是回文字符串。回文是一个向前和向后读取相同的字符串，本题中只考虑字母和数字字符。如果是，输出True；否则输出False。

测试用例：

```
用例1: A man, a plan, a canal: Panama
用例2: race a car
用例3: 111111111111111111112111111111111111111
```

思路：

回文字符串的判断：不论是从左往右，还是从右往左，字符的顺序都是一样的（如aba, abcba, abba等），而本题中的回文字符串只考虑数字和字母（字母不考虑大小写），不考虑其他字符，例如 “ab, cBa” 按本题要求则是回文字符串。

所以我先定义一个根据ASCII码判断是否为字符或数字的函数，用于把读取的字符串中的数字和字母按顺序提取出来得到一个新字符串。

```
def num_or_char(ss):     # 判断 s0 是否为数字或字母
    key = ord(ss)
    if 48 <= key <= 57 \
            or 65 <= key <= 90 \
            or 97 <= key <= 122:
        return True
    else:
        return False
```

再定义一个判断是否为回文字符串的函数，根据ASCII码的差值来判断首尾对应位置的字符是否相同（差值为0）或为大小写关系（差值为±32）.

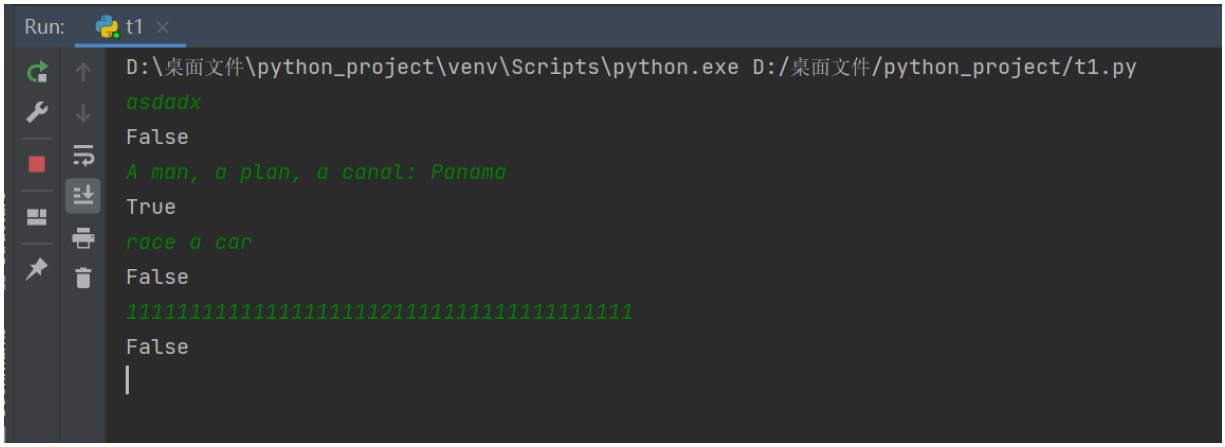
```
def is_curcle(ss):   # 判断 ss 是否为回文字符串
    k = 0
    j = len(ss)-1
    while k <= j:
        key = ord(ss[k]) - ord(ss[j])
        if key == 0 or key == 32 or key == -32:
            k = k + 1
            j = j - 1
        else:
            return False
    return True
```

最后将前面得到的新字符串用is_curcle函数判断是否为回文字符串，是则输出True,否则输出False。

代码：

```python
# 回文字符串：不论是从左往右，还是从右往左，字符的顺序都是一样的（如 aba,abcba,abba 等）
# 本题中的回文字符串只考虑数字和字母（不考虑大小写），不考虑其他字符
def is_curcle(ss):  # 判断 ss 是否为回文字符串
    k = 0
    j = len(ss)-1
    while k <= j:
        key = ord(ss[k]) - ord(ss[j])
        if key == 0 or key == 32 or key == -32:
            k = k + 1
            j = j - 1
        else:
            return False
    return True


def num_or_char(ss):    # 判断 s0 是否为数字或字符
    key = ord(ss)
    if 48 <= key <= 57 \
            or 65 <= key <= 90 \
            or 97 <= key <= 122:
        return True
    else:
        return False


s0 = "s0"
while s0 != "":     # 当输入字符串为空时停止
    s0 = input()
    if s0 == "":
        break
    s0_len = len(s0)
    s1 = ""
    # 将 s0 转化为只含数字和字母的字符串并保存在 s1 中
    for i in range(s0_len):
        if num_or_char(s0[i]):
            s1 = s1 + s0[i]
    # print(s1)
    if is_curcle(s1):
        print("True")
    else:
        print("False")

# A man, a plan, a canal: Panama - True
# race a car - False
# 111111111111111111112111111111111111111 - False
```

测试结果：



Task 2 ★

制作一个两人石头剪刀布游戏。 （提示：使用input 输入两个玩家的选择，比较它们，打印出祝贺获胜者的消息，并询问玩家是否想开始新游戏，输入y 则重新开始游戏，输入n 退出游戏，石头：Rock，剪刀：Scissors，布：Paper）

示例输入：

```
Player1 input: Rock
Player2 input: Paper
```

示例输出：

```
Congratulate Player2!
Try new game?
```

思路：

在这里我默认两人的输入都是合法的（输入只有：Rock，Scissors，Paper三者）

判断谁获胜只需根据两人的输入进行判断，由于结果只有三种：Player1胜，平，Player2胜。所以使用if...elif...else语句进行判断：

当(p1 == "Rock" and p2 == "Scissors")or (p1 == "Scissors" and p2 == "Paper") or (p1 == "Paper" and p2 == "Rock")时Player1胜；p1==p2时平局；否则Player2胜。（再次强调默认两人输入合法）

然后根据满足的条件输出对应的结果即可。

由于玩家输入y/n来判断游戏是否继续，所以我将以上模块放在一个while循环里面，在while循环之前设置一个bool类型变量keep初始值为真，当keep为真时while循环进行，最后根据玩家输入的y/n选择改变keep的值，为y时keep仍为True，为n时keep为False。

代码：

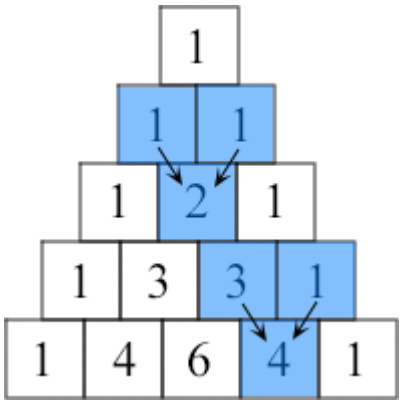
```
keep = True
while keep:
    print("Player1 input:")
    p1 = input()
    print("Player2 input:")
    p2 = input()
    if (p1 == "Rock" and p2 == "Scissors")\
            or (p1 == "Scissors" and p2 == "Paper") \
            or (p1 == "Paper" and p2 == "Rock"):
        print("Congratulate Player1!")
    elif p1 == p2:
        print("No Winner!")      # 两人平局
    else:
        print("Congratulate Player2!")
    print("Try new game?")
    new_game = input()
    if new_game == "y":
        keep = True
    elif new_game == "n":
        keep = False
```

测试结果：

```
t1 × t2 ×
D:\桌面文件\python_project\venv\Scripts\python.exe D:/桌面文件/python_project/t2.py
Player1 input:
Rock
Player2 input:
Paper
Congratulate Player2!
Try new game?
y
Player1 input:
Scissors
Player2 input:
Paper
Congratulate Player1!
Try new game?
y
Player1 input:
Rock
Player2 input:
Rock
No Winner!
Try new game?
n

Process finished with exit code 0
```

Task 3 ★★★

实现一个可以打印出帕斯卡三角形前 n 行的函数。（帕斯卡三角形，又称杨辉三角，每个数等于它上方两数之和）



思路：

帕斯卡三角形，每个数等于它上方两数之和： data[n][i] = data[n-1][i-1]+data[n-1][i] ， i-1或i越过[0，n-2]的界时取0。

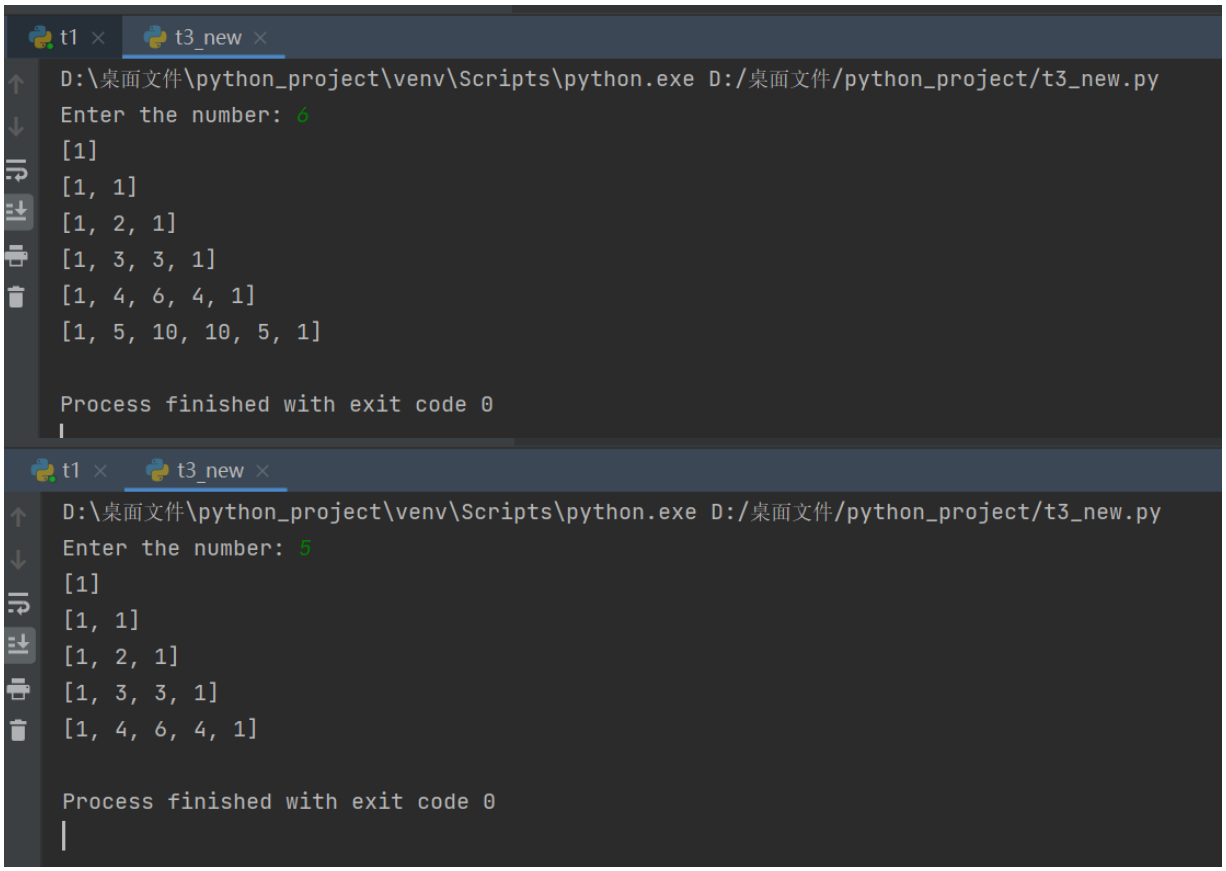
而Python中没有数组，所以使用列表组成线性表，为了防止越界的情况，从第三行开始使用data[n][i] = data[n-1][i-1]+data[n-1][i]给每行中间的元素赋值，每行的第1个元素都为1，除第一行外每行的末尾再插入1作为末尾。

最后，顺序输出列表的每个元素（每个元素都是一个列表，代表第几行）。

代码：

```python
# 帕斯卡三角形： data_n[i] = data_n-1[i-1]+data_n-1[i] , i-1 或 i 越过[0，n-2]的界时取 0
num = int(input("Enter the number: "))
list1 = []  # 空列表
for i in range(num):
    list1.append([])          # 使用线性表，第 i 行有 i 个数
    list1[i].append(1)        # 每行第一个数为 1
    for j in range(1, i):
        # 给每行中间非 1 的数赋值，i 的范围为 2~num-1，从第 3 行到最后一行，避免越界
        list1[i].append(list1[i - 1][j - 1] + list1[i - 1][j])
    if i != 0:                # 第一行只有 1 个数
        list1[i].append(1)    # 每行最后一个为 1
for i in range(num):          # 输出每行
    print(list1[i])
```

测试结果：



Task 4 ★★

实现一个关于列表的类 MyList，该类包括以下方法：

1. 添加元素：push(args)，参数元素类型为字符串或整型，将 args 添加到列表末尾；
2. 获取元素：get(num)，参数为整型，从列表中随机获取 num 个元素并打印；
3. 获取列表长度：len()，打印列表长度；
4. 删除元素：del()，删除列表第一个元素并打印；
5. 清空列表：clear()，清空列表元素。

在代码中实例化类并测试每个方法，并打印每次操作后的列表。

思路：

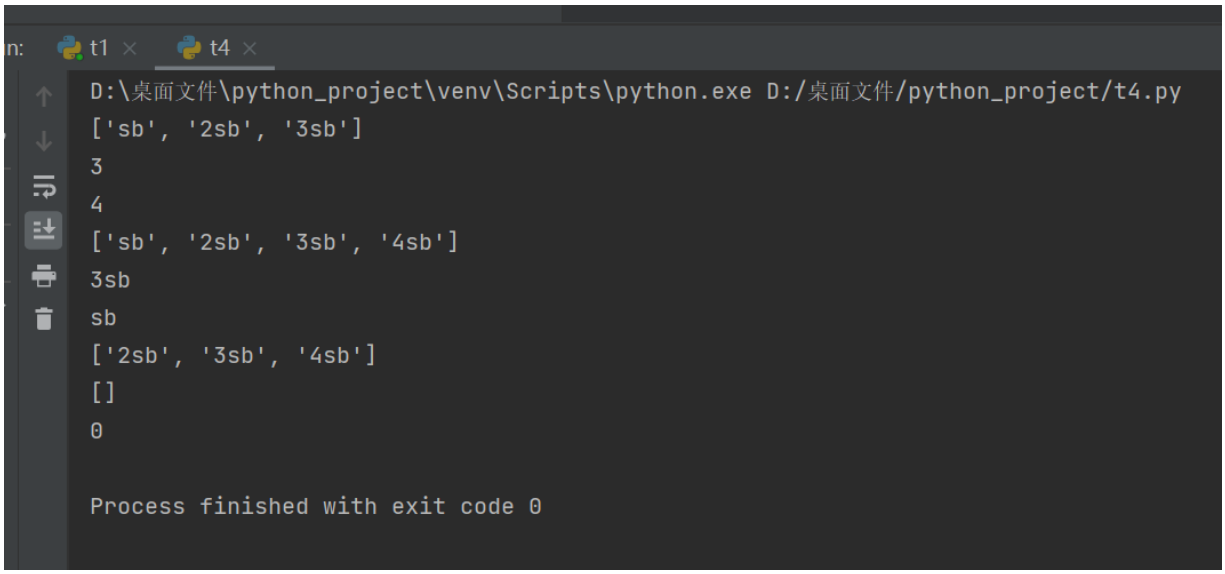
直接利用列表的相关函数构造需要的类里的函数。

同时在构造函数中使用深复制，利用copy.deepcopy（）函数。

代码：

```
import copy  # 用于深拷贝


class MyList:
    mylist = []

    def __init__(self):
        self.mylist = []

    def __init__(self, list00):
        self.mylist = copy.deepcopy(list00)

    def push(self, args):
        self.mylist.append(args)

    def get(self, num):
        print(self.mylist[num])

    def len0(self):
        print(len(self.mylist))

    def del0(self):
        print(self.mylist.pop(0))

    def clear(self):
        while len(self.mylist) > 0:
            self.mylist.pop()

    def print0(self):    # 用于输出列表
       print(self.mylist)


list0 = ['sb', '2sb', '3sb']     # 以下为实例化测试
list1 = MyList(list0)    # 构造函数
list1.print0()           # 这里输出应为 ['sb', '2sb', '3sb']
list1.len0()             # len()函数，此处输出应为 3
list1.push('4sb')        # push(args)函数
list1.len0()             # len()函数，应输出 4
list1.print0()           # 这里输出应为 ['sb', '2sb', '3sb', '4sb']
list1.get(2)             # get(num)函数 ，应输出第 2 个元素(从 0 开始)为 3sb
list1.del0()             # del()函数 ，删除首个元素并输出，应输出 sb
list1.print0()           # 应输出 ['2sb', '3sb', '4sb']
list1.clear()            # clear()函数
list1.print0()           # 应输出[]
list1.len0()             # 应输出 0
```

测试结果：


n:
t1
t4
D:\桌面文件\python_project\venv\Scripts\python.exe D:/桌面文件/python_project/t4.py
['sb', '2sb', '3sb']
3
4
['sb', '2sb', '3sb', '4sb']
3sb
sb
['2sb', '3sb', '4sb']
[]
0
Process finished with exit code 0