例 6.1 Python 程序代码

本代码以一个经典的马氏链模型为例，论述如何实现 3 状态下的离散时间马氏链。

读者可根据本代码提示实现书中例题代码。

如果某人心情不好，她可能会跑步，也有可能冰淇淋，或者打个盹儿来调整。

根据以往数据，如果她睡了一觉调整心情，第二天她有 60%的可能选择去跑步，20%的可能选择继续待在床上，还有 20%的可能选择吃冰淇淋。

如果她选择跑步散心，第二天她有 60%的可能选择接着跑步，30%的可能选择吃冰淇淋，只有 10%的可能选择去睡觉。

最后，如果她难过时选择吃冰淇淋，第二天只有 10%的可能性选择继续吃冰淇淋，有 70%的可能性选择去跑步，还有 20%的可能性选择睡觉

请问，从睡觉状态开始，2 天后她最后选择跑步（跑步状态）的概率是多少？

"""

```
import numpy as np
import random as rm
# 状态空间
states = ["Sleep","Icecream","Run"]
# 可能的事件序列
transitionName = [["SS","SR","SI"],["RS","RR","RI"],["IS","IR","II"]] # 概率矩阵（转移矩阵）
transitionMatrix = [[0.2,0.6,0.2],[0.1,0.6,0.3],[0.2,0.7,0.1]]
if sum(transitionMatrix[0])+sum(transitionMatrix[1])+sum(transitionMatrix[1]) != 3:
    print("Somewhere, something went wrong. Transition matrix, perhaps?")
else: print("All is gonna be okay, you should move on!! ;)")

def activity_forecast(days):
    # 选择初始状态
    activityToday = "Sleep"
    print("Start state: " + activityToday)
    # 应该记录选择的状态序列。这里现在只有初始状态。
    activityList = [activityToday]
    i = 0
    # 计算 activityList 的概率
    prob = 1
    while i != days:
        if activityToday == "Sleep":
            change = np.random.choice(transitionName[0],replace=True,p=transitionMatrix[0])
            if change == "SS":
                prob = prob * 0.2
                activityList.append("Sleep")
                pass
            elif change == "SR":
```

```
                prob = prob * 0.6
                activityToday = "Run"
                activityList.append("Run")
            else:
                prob = prob * 0.2
                activityToday = "Icecream"
                activityList.append("Icecream")
        elif activityToday == "Run":
            change                                                              =
np.random.choice(transitionName[1],replace=True,p=transitionMatrix[1])
            if change == "RR":
                prob = prob * 0.5
                activityList.append("Run")
                pass
            elif change == "RS":
                prob = prob * 0.2
                activityToday = "Sleep"
                activityList.append("Sleep")
            else:
                prob = prob * 0.3
                activityToday = "Icecream"
                activityList.append("Icecream")
        elif activityToday == "Icecream":
            change                                                              =
np.random.choice(transitionName[2],replace=True,p=transitionMatrix[2])
            if change == "II":
                prob = prob * 0.1
                activityList.append("Icecream")
                pass
            elif change == "IS":
                prob = prob * 0.2
                activityToday = "Sleep"
                activityList.append("Sleep")
            else:
                prob = prob * 0.7
                activityToday = "Run"
                activityList.append("Run")
        i += 1
    print("Possible states: " + str(activityList))
    print("End state after "+ str(days) + " days: " + activityToday)
    print("Probability of the possible sequence of states: " + str(prob))

# 预测 2 天后的可能状态
activity_forecast(2)
```