파이썬프로그래밍및실습

Infectious Disease Infection Probability Notification Program

Final Report

Date : 2023.12.23
Name : joosungmin
ID : 233897

1. Introduction

1) Background

As the COVID pandemic breaks out, people's worries about the epidemic grow. It is necessary to provide information on various infectious diseases. In order to solve this problem, a program is needed to calculate the probability of infection for various infectious diseases and let users know if there is an overlapping movement with the infected person by receiving activity records.

2) Project goal

It aims to compare the input user's activity records with the activity records of infected people of various infectious diseases to find the probability of infection according to the contact time if there is an overlapping movement, and if the probability is high, it is judged as a preliminary infected person and to create a program to warn other users who have come into contact with preliminary infected people other than the infected person.

3) Differences from existing programs

In the case of existing COVID-19 emergency disaster texts, people are simply informed of the time when the confirmed person overlaps with the route, so the judgment on the COVID-19 test is determined according to their condition, and if it is an incubation period, it may not be tested, which may cause the spread to repeat again. We are different from existing programs in that we find the probability of infection by considering the contact time of each disease, and we repeat the process of dividing users with high probability into preliminary infections and comparing them with other users to prevent resurgence.

2. Functional Requirement

1) **Function 1** Register travel routes

- The ability to receive and store the user's travel path by the hour

(1) Detailed function 1 Enter the name of the place that was in hourly increments

- Enter the place where you moved from 8 a.m. to 10 p.m. and the time you stayed.

2) **Function 2** Check for infectious diseases

- The ability to ask for the presence or absence of three infectious diseases (flu, corona, and tuberculosis) and receive input as o,x

2) **Function 3** Notify others

- The function of calculating the probability of transmission according to the overlapping time by comparing the travel route of a user infected with an infectious disease one day before with other users and informing users

(1) Detailed function 1 The time overlapping with the travel route of a user infected with an infectious disease one day ago is calculated, and the probability of infection is set as 30% for 1 hour, 60% for 2 hours, and 90% for 3 hours, respectively. (There is no accurate information on the correlation between contact time and infection probability, so the probability is arbitrarily set) Notify other users.

(2) Detailed function 2 In order to calculate the probability of secondary infection, a third party, not a user, seeks the time overlapping with the infected person,

Calculate the time when the third party and the user overlap. (20% per hour was calculated)

3. Implementation

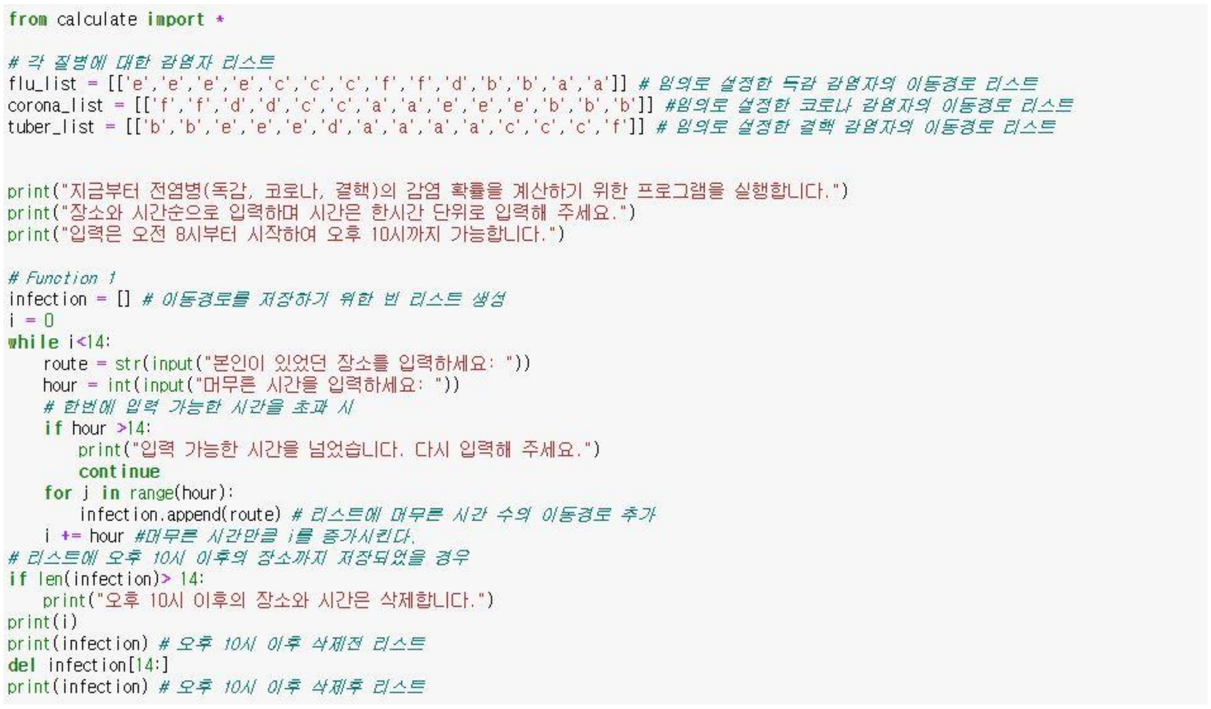
(1) Register travel routes

- input: route, hour / output: infection

- The place and time stayed by the user are inputted as 'input' and stored in the travel path list.

- class, list, condition, loop

- code screen shot



```python
from calculate import *

# 각 질병에 대한 감염자 리스트
flu_list = [['e','e','e','e','c','c','c','f','f','d','b','b','a','a']] # 임의로 설정한 독감 감염자의 이동경로 리스트
corona_list = [['f','f','d','d','c','c','a','a','e','e','e','b','b','b']] #임의로 설정한 코로나 감염자의 이동경로 리스트
tuber_list = [['b','b','e','e','e','d','a','a','a','a','c','c','c','f']] # 임의로 설정한 결핵 감염자의 이동경로 리스트


print("지금부터 전염병(독감, 코로나, 결핵)의 감염 확률을 계산하기 위한 프로그램을 실행합니다.")
print("장소와 시간순으로 입력하며 시간은 한시간 단위로 입력해 주세요.")
print("입력은 오전 8시부터 시작하여 오후 10시까지 가능합니다.")

# Function 1
infection = [] # 이동경로를 저장하기 위한 빈 리스트 생성
i = 0
while i<14:
    route = str(input("본인이 있었던 장소를 입력하세요: "))
    hour = int(input("머무른 시간을 입력하세요: "))
    # 한번에 입력 가능한 시간을 초과 시
    if hour >14:
        print("입력 가능한 시간을 넘었습니다. 다시 입력해 주세요.")
        continue
    for j in range(hour):
        infection.append(route) # 리스트에 머무른 시간 수의 이동경로 추가
    i += hour #머무른 시간만큼 i를 증가시킨다.
# 리스트에 오후 10시 이후의 장소까지 저장되었을 경우
if len(infection)> 14:
    print("오후 10시 이후의 장소와 시간은 삭제합니다.")
print(i)
print(infection) # 오후 10시 이후 삭제전 리스트
del infection[14:]
print(infection) # 오후 10시 이후 삭제후 리스트
```

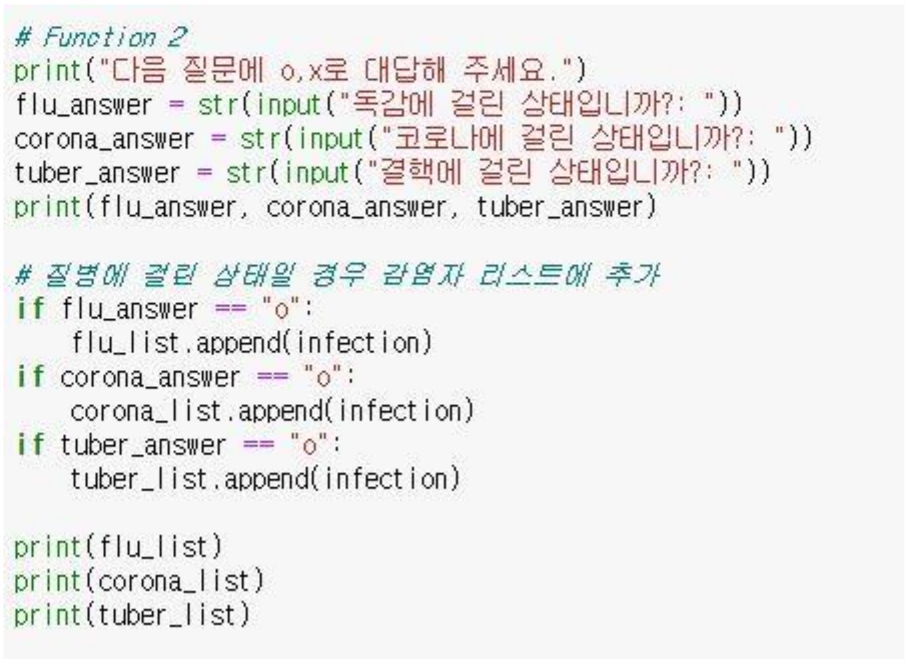
(2) Check for infectious diseases

- input: str(input) / output: flu_answer, corona_answer, tuber_answer

- Ask for the presence or absence of three infectious diseases (flu, corona, and tuberculosis) and receive input as o,x

- class

- code screen shot



```python
# Function 2
print("다음 질문에 o,x로 대답해 주세요.")
flu_answer = str(input("독감에 걸린 상태입니까?: "))
corona_answer = str(input("코로나에 걸린 상태입니까?: "))
tuber_answer = str(input("결핵에 걸린 상태입니까?: "))
print(flu_answer, corona_answer, tuber_answer)

# 질병에 걸린 상태일 경우 감염자 리스트에 추가
if flu_answer == "o":
    flu_list.append(infection)
if corona_answer == "o":
    corona_list.append(infection)
if tuber_answer == "o":
    tuber_list.append(infection)

print(flu_list)
print(corona_list)
print(tuber_list)
```

(3) Notify others 1st infection percentage

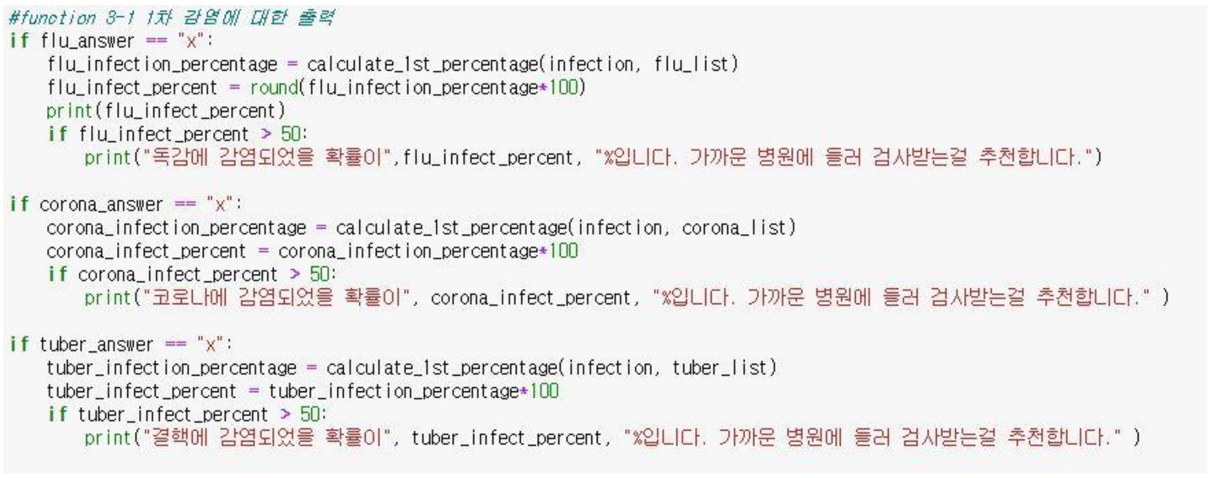
- input: a list of travel routes for an infected person with any disease, a list of travel routes for a user

output: infection_percentage

- Comparing the two lists, increasing the 'conflict_hour' if they are in the same place at the same time. Finally, the probability of infection is calculated. After calculating the probability of primary infection, notify the user if it exceeds 50%.

- class, list, loop, function, module

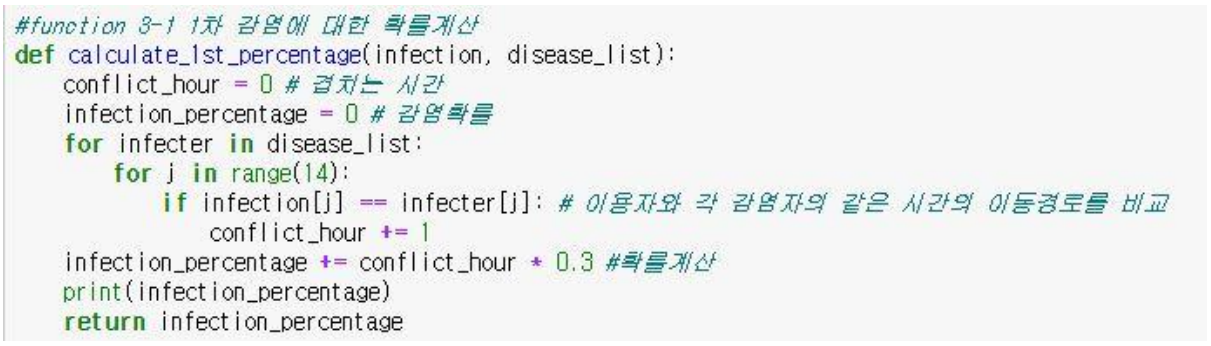
- code screen shot



```python
#function 3-1 1차 감염에 대한 출력
if flu_answer == "x":
    flu_infection_percentage = calculate_1st_percentage(infection, flu_list)
    flu_infect_percent = round(flu_infection_percentage*100)
    print(flu_infect_percent)
    if flu_infect_percent > 50:
        print("독감에 감염되었을 확률이",flu_infect_percent, "%입니다. 가까운 병원에 들러 검사받는걸 추천합니다.")

if corona_answer == "x":
    corona_infection_percentage = calculate_1st_percentage(infection, corona_list)
    corona_infect_percent = corona_infection_percentage*100
    if corona_infect_percent > 50:
        print("코로나에 감염되었을 확률이", corona_infect_percent, "%입니다. 가까운 병원에 들러 검사받는걸 추천합니다." )

if tuber_answer == "x":
    tuber_infection_percentage = calculate_1st_percentage(infection, tuber_list)
    tuber_infect_percent = tuber_infection_percentage*100
    if tuber_infect_percent > 50:
        print("결핵에 감염되었을 확률이", tuber_infect_percent, "%입니다. 가까운 병원에 들러 검사받는걸 추천합니다." )
```



```python
#function 3-1 1차 감염에 대한 확률계산
def calculate_1st_percentage(infection, disease_list):
    conflict_hour = 0 # 겹치는 시간
    infection_percentage = 0 # 감염확률
    for infecter in disease_list:
        for j in range(14):
            if infection[j] == infecter[j]: # 이용자와 각 감염자의 같은 시간의 이동경로를 비교
                conflict_hour += 1
    infection_percentage += conflict_hour * 0.3 #확률계산
    print(infection_percentage)
    return infection_percentage
```

(4) Notify others 2nd infection percentage

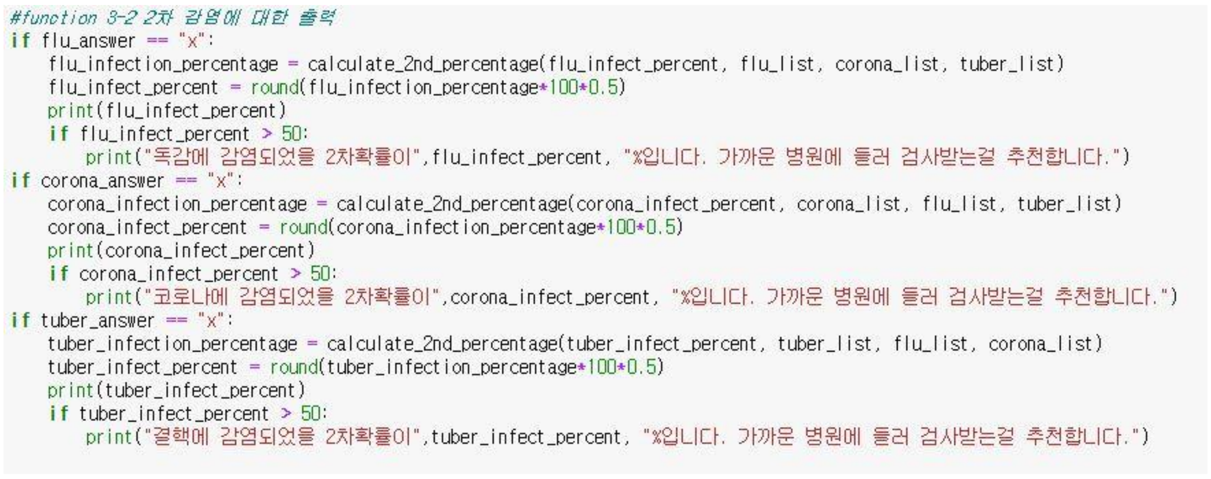
- input: a list of travel routes for an infected person with main disease, compare disease1, compare disease2

output: The probability of infection of the main disease through the compare disease

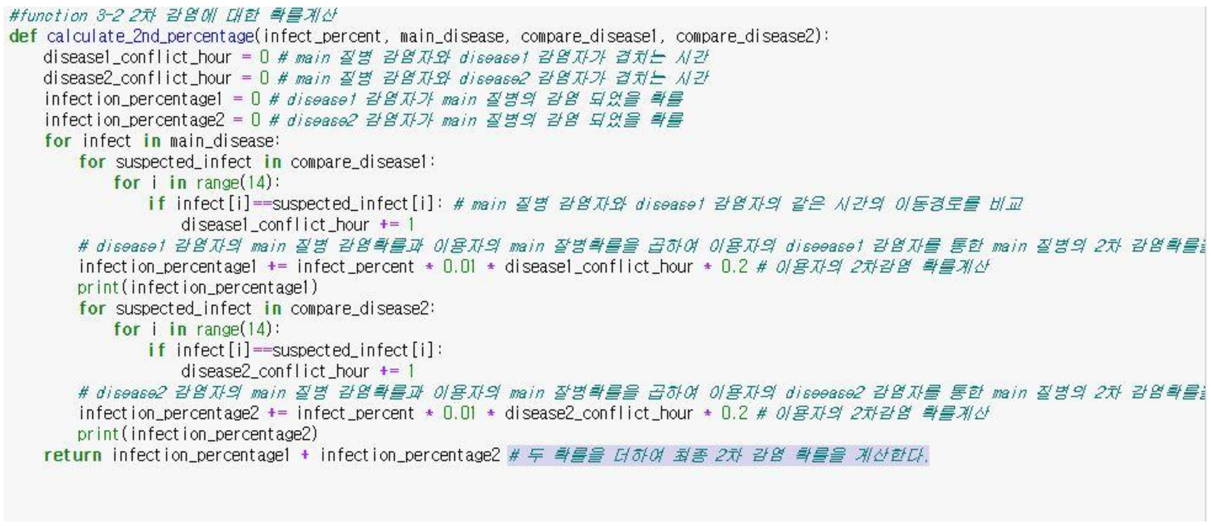
- Through comparison of routes of movement between different diseases and the probability of each primary infection, the probability of secondary infection of the main disease through a person infected with another disease is obtained.

- class, list, loop, function, module

- code screen shot



```python
#function 3-2 2차 감염에 대한 출력
if flu_answer == "x":
    flu_infection_percentage = calculate_2nd_percentage(flu_infect_percent, flu_list, corona_list, tuber_list)
    flu_infect_percent = round(flu_infection_percentage*100*0.5)
    print(flu_infect_percent)
    if flu_infect_percent > 50:
        print("독감에 감염되었을 2차확률이",flu_infect_percent, "%입니다. 가까운 병원에 들러 검사받는걸 추천합니다.")
if corona_answer == "x":
    corona_infection_percentage = calculate_2nd_percentage(corona_infect_percent, corona_list, flu_list, tuber_list)
    corona_infect_percent = round(corona_infection_percentage*100*0.5)
    print(corona_infect_percent)
    if corona_infect_percent > 50:
        print("코로나에 감염되었을 2차확률이",corona_infect_percent, "%입니다. 가까운 병원에 들러 검사받는걸 추천합니다.")
if tuber_answer == "x":
    tuber_infection_percentage = calculate_2nd_percentage(tuber_infect_percent, tuber_list, flu_list, corona_list)
    tuber_infect_percent = round(tuber_infection_percentage*100*0.5)
    print(tuber_infect_percent)
    if tuber_infect_percent > 50:
        print("결핵에 감염되었을 2차확률이",tuber_infect_percent, "%입니다. 가까운 병원에 들러 검사받는걸 추천합니다.")
```



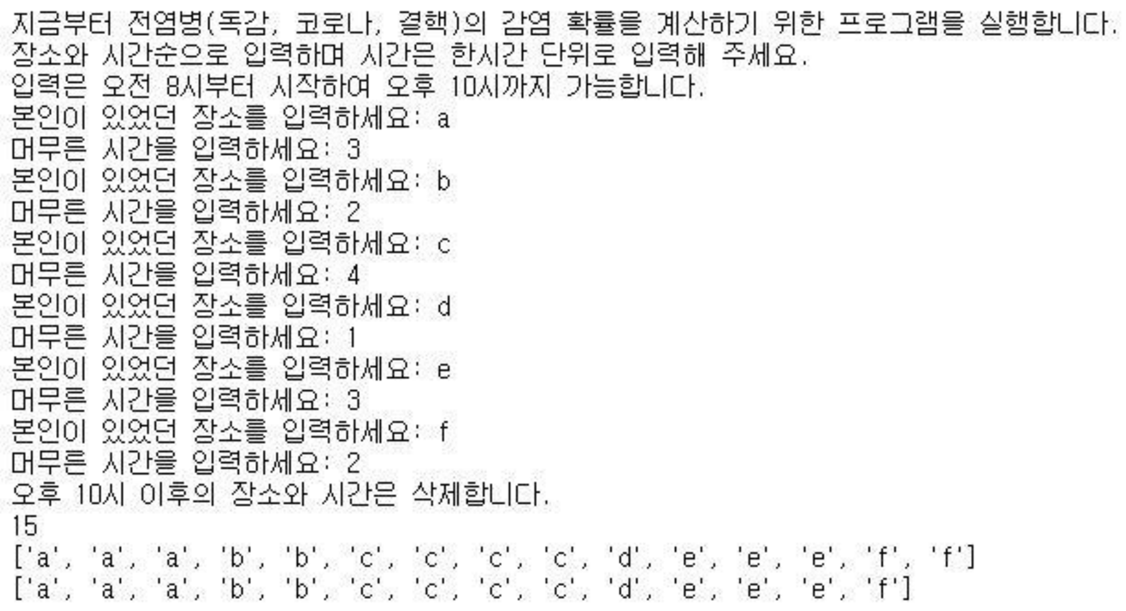
```python
#function 3-2 2차 감염에 대한 확률계산
def calculate_2nd_percentage(infect_percent, main_disease, compare_disease1, compare_disease2):
    disease1_conflict_hour = 0 # main 질병 감염자와 disease1 감염자가 겹치는 시간
    disease2_conflict_hour = 0 # main 질병 감염자와 disease2 감염자가 겹치는 시간
    infection_percentage1 = 0 # disease1 감염자가 main 질병의 감염 되었을 확률
    infection_percentage2 = 0 # disease2 감염자가 main 질병의 감염 되었을 확률
    for infect in main_disease:
        for suspected_infect in compare_disease1:
            for i in range(14):
                if infect[i]==suspected_infect[i]: # main 질병 감염자와 disease1 감염자의 같은 시간의 이동경로를 비교
                    disease1_conflict_hour += 1
        # disease1 감염자의 main 질병 감염확률과 이용자의 main 잘병확률을 곱하여 이용자의 diseease1 감염자를 통한 main 질병의 2차 감염확률
        infection_percentage1 += infect_percent * 0.01 * disease1_conflict_hour * 0.2 # 이용자의 2차감염 확률계산
        print(infection_percentage1)
        for suspected_infect in compare_disease2:
            for i in range(14):
                if infect[i]==suspected_infect[i]:
                    disease2_conflict_hour += 1
        # disease2 감염자의 main 질병 감염확률과 이용자의 main 잘병확률을 곱하여 이용자의 diseease2 감염자를 통한 main 질병의 2차 감염확률
        infection_percentage2 += infect_percent * 0.01 * disease2_conflict_hour * 0.2 # 이용자의 2차감염 확률계산
        print(infection_percentage2)
    return infection_percentage1 + infection_percentage2 # 두 확률을 더하여 최종 2차 감염 확률을 계산한다.
```

4. Test Result

(1) Register travel routes

- After receiving the input from the user, the total input time is output, the list is output, and the information after 10 p.m. is deleted, and a final list to be used for a subsequent function is output.

- test result screen shot

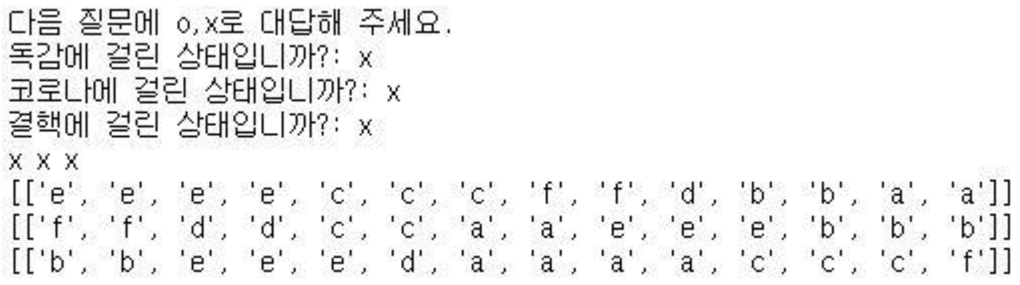


```
지금부터 전염병(독감, 코로나, 결핵)의 감염 확률을 계산하기 위한 프로그램을 실행합니다.
장소와 시간순으로 입력하며 시간은 한시간 단위로 입력해 주세요.
입력은 오전 8시부터 시작하여 오후 10시까지 가능합니다.
본인이 있었던 장소를 입력하세요: a
머무른 시간을 입력하세요: 3
본인이 있었던 장소를 입력하세요: b
머무른 시간을 입력하세요: 2
본인이 있었던 장소를 입력하세요: c
머무른 시간을 입력하세요: 4
본인이 있었던 장소를 입력하세요: d
머무른 시간을 입력하세요: 1
본인이 있었던 장소를 입력하세요: e
머무른 시간을 입력하세요: 3
본인이 있었던 장소를 입력하세요: f
머무른 시간을 입력하세요: 2
오후 10시 이후의 장소와 시간은 삭제합니다.
15
['a', 'a', 'a', 'b', 'b', 'c', 'c', 'c', 'c', 'd', 'e', 'e', 'e', 'f', 'f']
['a', 'a', 'a', 'b', 'b', 'c', 'c', 'c', 'c', 'd', 'e', 'e', 'e', 'f']
```

(2) Check for infectious diseases

- Through 'input', the infection of flu, corona, and tuberculosis is inputted as o,x and then outputted.

- test result screen shot

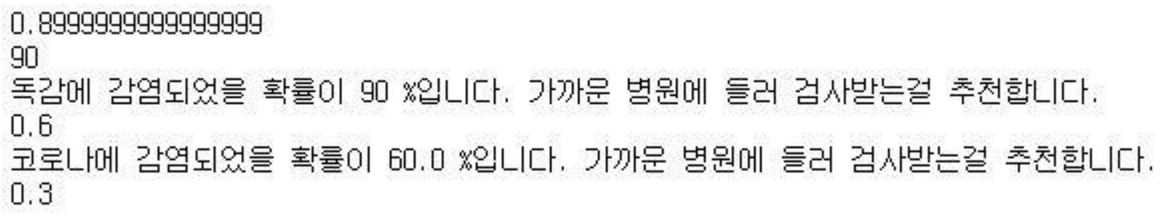


```
다음 질문에 o,x로 대답해 주세요.
독감에 걸린 상태입니까?: x
코로나에 걸린 상태입니까?: x
결핵에 걸린 상태입니까?: x
x x x
[['e', 'e', 'e', 'e', 'c', 'c', 'c', 'f', 'f', 'd', 'b', 'b', 'a', 'a']]
[['f', 'f', 'd', 'd', 'c', 'c', 'a', 'a', 'e', 'e', 'e', 'b', 'b', 'b']]
[['b', 'b', 'e', 'e', 'e', 'd', 'a', 'a', 'a', 'a', 'c', 'c', 'c', 'f']]
```

(3) Notify others 1st infection percentage

- If user answer 'o' for each disease, add it to the list of infected people, and find the probability for other diseases other than the infected disease. Establish an arbitrary path of infection for each disease, The probability of primary infection is calculated by comparing it with the movement path input by the user. After calculating the probability of primary infection, notify the user if it exceeds 50%.

- test result screen shot

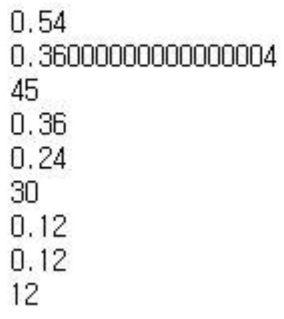


```
0.8999999999999999
90
독감에 감염되었을 확률이 90 %입니다. 가까운 병원에 들러 검사받는걸 추천합니다.
0.6
코로나에 감염되었을 확률이 60.0 %입니다. 가까운 병원에 들러 검사받는걸 추천합니다.
0.3
```

(4) Notify others 2nd infection percentage

- After determining the disease in which the user answered ‘x’ as the main disease, the movement path of the infected person of the main disease is compared with the movement path of the remaining infected people, and the secondary probability of the main disease is obtained through the infected person of another disease.

- test result screen shot



```
0.54
0.3600000000000004
45
0.36
0.24
30
0.12
0.12
12
```

5. Changes in Comparison to the Plan

1) **Function 3** Notify others

- **(1) Detailed function 1** The time overlapping with the travel route of a user infected with an infectious disease one day ago is calculated, and the probability of infection is set as 30% for 1 hour, 60% for 2 hours, and 90% for 3 hours, respectively. (There is no accurate information on the correlation between contact time and infection probability, so the probability is arbitrarily set) Notify other users. Users who overlap for more than 3 hours are designated as preliminary infections and repeat the above function once more. At this time, the probability of secondary user infection is (90%*probability for contact time).

- **(1) Detailed function 1** The time overlapping with the travel route of a user infected with an infectious disease one day ago is calculated, and the probability of infection is set as 30% for 1 hour, 60% for 2 hours, and 90% for 3 hours, respectively. (There is no accurate information on the correlation between contact time and infection probability, so the probability is arbitrarily set) Notify other users.

(2) Detailed function 2 In order to calculate the probability of secondary infection, a third party, not a user, seeks the time overlapping with the infected person,

Calculate the time when the third party and the user overlap. (20% per hour was calculated)

- The detailed functions of function 3 are divided into two, and the secondary infection is not only causing the user to infect others

In consideration of the possibility of infection of the user, it is modified so that it can be calculated for all users,

The calculation for the probability of secondary infection was slightly changed.

6. Lessons Learned & Feedback

I had a hard time learning Python for a semester, but it was good because I felt like I was getting a lot of things. If I work on a similar project next time, I would like to try other functions that I haven't used this time.