Project_maphia 보고서

20180391 이지훈

1. only citizen

우선 시민의 수와 마피아의 수를 입력 받아서, 이 두 입력 값을 이용하여 Gamer라는 새로운 배열로 정의하였다. 만약 시민의 수가 7, 마피아의 수가 3이면, [1,2,3,4,5,6,7,8,9,10]이다. 여기서 마피아는 8 9 10에 해당한다. 시민과 마피아를 구별하는 중요 값은 시민과 마피아의 경계인 "7" 이라고 할 수 있다. 이 "7"은 입력 받은 시민의 수를 의미하기도 한다. 이 중요값을 checking_number라는 변수에 저장하였다.

시민(아무 역할이 없는)과 마피아만 있는 게임의 핵심은, 어떻게 투표를 통해 Gamer(마피아+시민) 중에 한명을 죽이고 어떻게 마피아의 습격으로 Gamer 중 시민만 골라 죽이는지에 있다고 생각했다.

우선 투표를 통해 아무나 골라 죽이는 알고리즘은 import random을 해서 여기의 choice 함수를 사용하였다. die_people = choice(Gamer)를 통해 구현하였다. 단순히 전체 Gamer의 배열에서 하나를 선택하는 코드이다. 그리고 Gamer.remove(die_people)을 통해 Gamer의 배열에 해당 사망자를 remove하는 형태로 구현하였다.

밤에 마피아가 시민만 골라 죽이는 알고리즘은, 앞서 말한 checking_number 변수를 사용하여 구현하였다. 어떠한 형태로든 시민의 수가 줄면 checking_number가 하나 줄도록 구현한 후 die_people = choice(Gamer[:checing_number])로 구현하였다. 이렇게 하면 마피아가 시민만 골라 죽일 수 있게 된다.

전체 코드의 "상태기계"는, is_exist_citizens, is_exist_maphias 라고 할 수 있다.

이름에서 볼 수 있듯 전체 Gamer의 배열에서 시민 혹은 마피아가 존재하는지 판단해주는 변수라고 할 수 있다. 이를 위해 checking_exist_citizens , checking_exist_maphia 함수를 선언해주었다. 역시 이름 그대로 Gamer배열과 citizens_number를 받아서 배열을 돌며 citizens_number보다 작은값(시민) 혹은 큰 값(마피아)가 있는지를 검사해주는 함수이다. 존재한다면 True, 아니면 False를 리턴하게 해주었다.

최초 is_exist_citizens, is_exist_maphias는 True로 설정하였고, 투표가 끝난 후와 마피아의 습격이 끝난 후에 checking_exist함수를 콜하여 상태기계를 최신화하도록 하였다. 만약 is_exist_citizens 혹은 is_exist_maphias가 false라면 바로 무한while루프를 빠져나오게 구현하였다.

또한 만약 is_exist_citizens가 false라면 who_is_winner를 maphia로 저장한 후 최종적으로 이 who_is_winner를 리턴시키도록 구현하였다. 반대의 경우도 마찬가지이다

.

코드의 마지막에는

일종의 main 함수 역할을 하는 game_start 함수에서 받은 리턴값(who_is_winner)를 받아서, 이 게임의 승리팀의 통계를 내는 코드를 구현하였다. Total_citizen_win, total_maphia_win 변수를 설정하여 각각의 승리팀의 승리횟수를 저장하도록 구현하였다.

최대한 정확한 통계값을 얻기 위하여, 위 코드를 for루프로 구현하였다. 이를 통해 각각의 경우에 대해 10만번의 데이터를 구할 수 있었다. (통계값 분석은 마지막에 소개한다)

코드와 최종 결과화면을 좀 더 실감나게 구현하기 위해, 최대한 함수나 변수명을 가독성 있게 구현하였고 생존자들을 출력할 때 잠시 Gamer 배열을 string형태의 배열로 바꾸어 주고 임의의 사람이름을 각각 넣어주게 구현함으로써 진짜 게임처럼 진행되도록 구현하였다.

2. with ugly police

이후의 4단계 로직은 only citizen 코드에서 몇 가지를 추가하는 개념이다.

Police를 위해 메인코드에 "target"이라는 변수를 설정하였다. 그리고 0으로 초기화하였다.

먼저 ugly police라는, 자신이 조사했던 대상을 기억못하는(대상을 중복으로 검사하는) 경찰을 구현하였다. Set_police_among_citizens 함수를 통해 시민 중 무작위로 하나를 골라 경찰의 직업을 주는 코드를 구현하였다. (무작위로 골라진 시민은 0이 됨). 하지만 ugly police도 자기자신은 조사하지 않도록 하기 위하여, police_work_uglycase 함수에서 만약 자기 자신을 고르면 다시 고르도록 코드를 구현하였다. Police_work_uglycase의 인자에 array값과 citizens_number를 주어, 만약 고른 대상이 citizens_number 보다 작거나 같으면 시민이므로 target에 0을 넣어 리턴하고(그대로 0을 유지하기 위해) citizens_number보다 크면 마피아이므로 target에 그 마피아값을 넣어 리턴하였다.

메인코드에 경찰의 구현을 위해, 상태기계에 is_exist_police를 추가하였다.

경찰은 밤에 활동을 하므로, 밤에 is_exist_police가 true라면 police_work_uglycase를 수행하도록 하였다. 수행결과를 앞서 말한 target에 저장해놓고, 만약 그 target이 마피아이고(target != 0이고) 다음날 아침에도 경찰이 살아있으면(is_exist_police = True) 투표로 마피아를 죽이게끔 코드를 구현하였다. 그렇지 않으면 원래대로 무작위로 한명을 투표로 죽이게끔 구현하였다. 마피아를 죽이는데 성공하면 다시 target을 초기값인 0으로 초기화하였다.

경찰도 시민이므로 경찰이 죽으면 checking_number가 1 작아지도록 구현하였다.

3. with smart police

Ugly police와 똑 같은 구현이다. 다만 smart police는 자신이 조사했던 대상을 기억하는(대상을 중복으로 검사하지 않는) 경찰이다. Ugly smart police와 전체코드는 동일하나 police work을 police_work_smartcase로 다르게 구현하였다. 전에 조사했던 내용은 다시 조사하지 않도록 하기 위하여, 조사했던 내용을 저장해주는 배열 array_for_smartpolice를 선언해주었다.

Police_work_smartcase의 인자로 받은 array에서, 이 배열 array_for_smartpolice의 값을 (array에 해당 값이 있다면) 빼주어 이 배열에서 무작위로 하나를 골라 target하는 알고리즘을 구현하려 했으나, 파이썬의 얕은 복사의 개념으로 원래 array인 Gamer의 값이 손상됨을 확인할 수 있었다.

그래서 import copy를 해서 deepcopy 함수를 이용하여 깊은 복사의 개념을 사용하였다.

그리고 최초 array_for_smartpolice 배열에 0을 넣어 초기화해줌으로써 경찰 입장에서 절대 자기 자신을 조사하지 않도록 구현할 수 있었다. 이후는 같은 방법으로 target을 최신화해주었다.

그리고 최대한 array_for_smartpolice의 배열을 크게 만들지 않게 하기 위해 조사한 key가 array_for_smartpolice에 없을때에만 append 하도록 구현하였다.

또한 다른 메인코드는 모두 동일하지만, 맨 마지막의 확률 분석 코드에서, 새로운 게임이 시작될 때마다 다시 array_for_smartpolice 배열을 [0]을 초기화해줌으로써 모든 파생되는 버그를 해결할 수 있었다.

4. with smart police, doctor

(3)의 코드에 doctor가 추가된 코드이다. Doctor를 위해 메인코드에 "saver"라는 변수를 설정하고 -10으로 초기화하였다. (사용하지 않는 음수값으로 초기화) . Set_doctor_among_citizen 함수를 통해 전체 시민중(경찰제외) 한명을 골라 의사라는 직업을 주었다(의사는 -1로 구현).

Doctor_work(array) 함수에선, 단순히 인자로 받은 array에서 한명을 골라 saver에 저장한 후 리턴하도록 구현하였다.

메인코드에 doctor구현을 위해 is_exist_doctor 상태변수를 추가하였다

의사 역시 밤에 활동하므로, 밤에 is_exist_doctor가 true이면 doctor_work을 수행하도록 하였다.

그리고 나서, 마피아가 죽일 예정인 대상 (die_people)과 비교해서 만약 saver와 같다면 그 대상을 remove 하지 않는 구조로 코드를 구현하였다. (if else 로 쉽게 구현)

의사가 일단 work를 하게 되면 살리는데 성공하든 안하든 다시 saver를 -10으로 초기화하였다.

의사도 시민이므로 의사가 죽으면 checking_number를 1 작아지도록 구현하였다.

5. with smart police, doctor, ARMY, NOBLE

단순히 시민을 0 마피아를 1로 구현하지 않고 12345..의 형태의 독립된 정수값으로 정의해줌으로써 이렇게 각 시민들에게 새로운 직업을 배치하는 작업이 단순했다.

그래서 난 남은 시민들 중 2명 에게 "군인" 과 "귀족"의 직업을 주었다.

먼저 군인은, 마피아의 습격으로부터 한번 살게 해주는 직업이고,

귀족은 투표로부터 한번 살게 해주는 직업으로 정의했다.

(군인과 귀족은) (경찰과 의사)처럼 is_exist_의 상태기계까지 줄 필요는 없다고 생각이 들었다.

특정한 행동을 하는게 아니고 단순히 특정 상황에 대해 목숨이 하나 더 있는 것이기 때문이다.

일단 set__among 함수로 시민 중 한명에게 각 직업을 랜덤으로 배분해주었고(중복되지 않게 구현함) (군인은 -2 귀족은 -3)

코드의 일관성을 위해 army_work와 noble_work를 간단하게나마 구현하였다.

메인코드에 army와 noble을 위해 army_life, noble_life라는 변수를 2로 초기화해주어 선언해주었고, 앞서 말한 work 함수가 발동되면 각각의 life변수가 1로 바뀌게 코드를 구현하였다.

각각의 work 함수가 발동되는 조건은, 군인의 경우 밤에 마피아가 죽일 대상인 die_people이

-2이고 army_life가 2일 경우이다. 여기서 한가지 염두해야 할 점은, 의사의 존재이다.

만약 의사가 살릴 saver와 마피아가 죽일 die_people이 "군인"으로 일치한다면 우선적으로 의사가 살리는 개념으로 수행되도록 구현하였다. 살짝 복잡할 수 있었지만 if else 구문으로 쉽게 해결할 수 있었다.

귀족의 경우는 더 간단했다. 단순히 투표로 죽일 대상인 die_people이 -3이고 noble_life가 2일 때 work 함수가 발동되도록 구현하였다.

각각의 Work 함수에서 각각의 life 변수값이 재할당되기 때문에 따로 다시 초기화할 필요는 없었다. 두 경우 모두 한번 work 함수가 발동되면 다시는 work가 발동되지 못하도록 함으로써 목숨이 2개인 구조를 구현할 수 있었다.

군인과 귀족 모두 시민이므로, 이 둘이 각각 죽으면 checking_number가 1 줄어들도록 구현하였다.

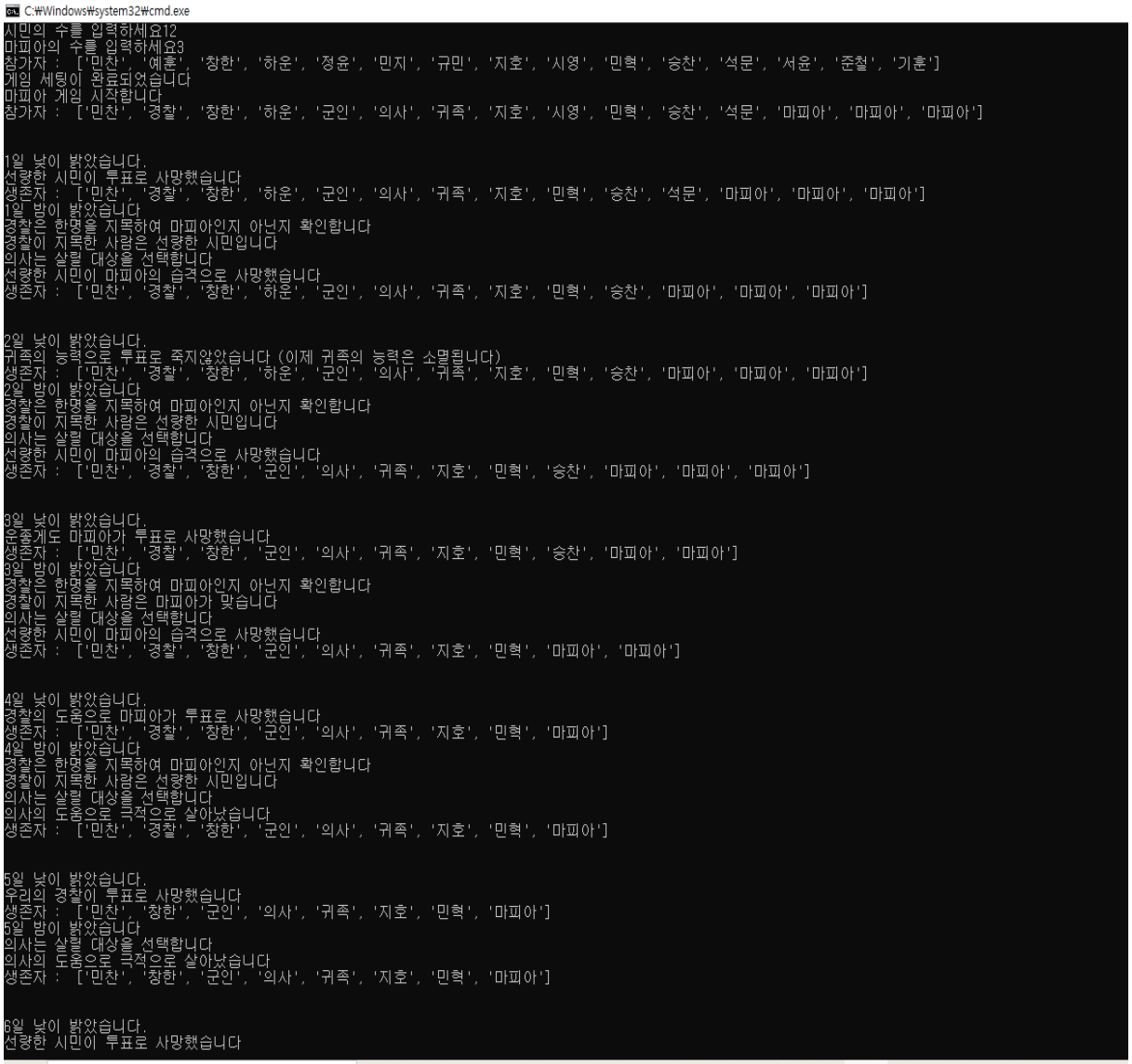
이렇게 함으로써 경찰, 의사 말고도 군인, 귀족의 직업을 추가하여 생각해 볼 수 있었다. 내 코드의 구조상 경찰과 의사의 수를 늘린다거나 또다른 직업을 부여하는 것도 쉽게 가능할 거 같다는 생각이 들었다. 최대한 수정이 쉽도록 구현하였고(하드코딩이 되지 않도록) 이후 추가적인 작업이 필요하다면 간편히 구현할 수 있을 것 같다.

#아래는 중간결과 test case 중 하나입니다(with smart police, doctor)

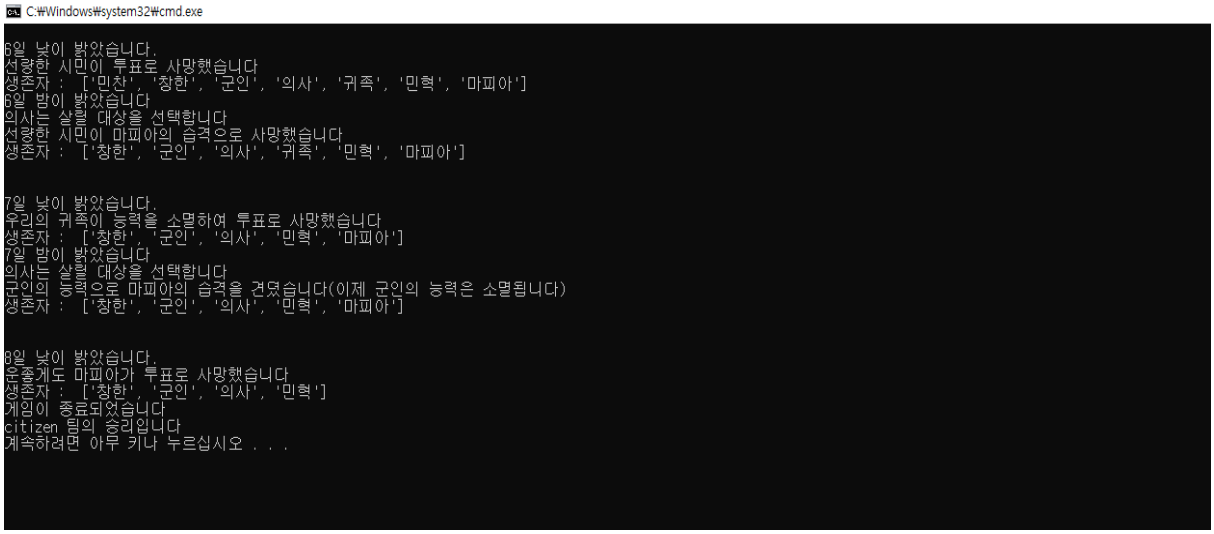
```
C:₩Windows₩system32₩cmd.exe
시민의 수를 입력하세요11
마피아의 수를 입력하세요4
참가자 :  ['민찬', '예훈', '창한', '하운', '정윤', '민지', '규민', '지호', '시영', '민혁', '승찬', '석문', '서윤', '준철', '기훈']
게임 세팅이 완료되었습니다
마피아 게임 시작합니다
참가자 :  ['경찰', '예훈', '창한', '하운', '정윤', '민지', '규민', '지호', '시영', '의사', '승찬', '마피아', '마피아', '마피아', '마피아']


1일 낮이 밝았습니다.
선량한 시민이 투표로 사망했습니다
생존자 :  ['경찰', '예훈', '창한', '하운', '민지', '규민', '지호', '시영', '의사', '승찬', '마피아', '마피아', '마피아', '마피아']
1일 밤이 밝았습니다
경찰은 한명을 지목하여 마피아인지 아닌지 확인합니다
경찰이 지목한 사람은 마피아가 맞습니다
의사는 살릴 대상을 선택합니다
의사의 도움으로 극적으로 살아났습니다
생존자 :  ['경찰', '예훈', '창한', '하운', '민지', '규민', '지호', '시영', '의사', '승찬', '마피아', '마피아', '마피아', '마피아']


2일 낮이 밝았습니다.
경찰의 도움으로 마피아가 투표로 사망했습니다
생존자 :  ['경찰', '예훈', '창한', '하운', '민지', '규민', '지호', '시영', '의사', '승찬', '마피아', '마피아', '마피아']
2일 밤이 밝았습니다
경찰은 한명을 지목하여 마피아인지 아닌지 확인합니다
경찰이 지목한 사람은 마피아가 맞습니다
의사는 살릴 대상을 선택합니다
선량한 시민이 마피아의 습격으로 사망했습니다
생존자 :  ['경찰', '예훈', '창한', '하운', '민지', '지호', '시영', '의사', '승찬', '마피아', '마피아', '마피아']


3일 낮이 밝았습니다.
경찰의 도움으로 마피아가 투표로 사망했습니다
생존자 :  ['경찰', '예훈', '창한', '하운', '민지', '지호', '시영', '의사', '승찬', '마피아', '마피아']
3일 밤이 밝았습니다
경찰은 한명을 지목하여 마피아인지 아닌지 확인합니다
경찰이 지목한 사람은 선량한 시민입니다
의사는 살릴 대상을 선택합니다
의사의 도움으로 극적으로 살아났습니다
생존자 :  ['경찰', '예훈', '창한', '하운', '민지', '지호', '시영', '의사', '승찬', '마피아', '마피아']


4일 낮이 밝았습니다.
선량한 시민이 투표로 사망했습니다
생존자 :  ['경찰', '예훈', '창한', '하운', '민지', '지호', '의사', '승찬', '마피아', '마피아']
4일 밤이 밝았습니다
경찰은 한명을 지목하여 마피아인지 아닌지 확인합니다
경찰이 지목한 사람은 선량한 시민입니다
의사는 살릴 대상을 선택합니다
선량한 시민이 마피아의 습격으로 사망했습니다
생존자 :  ['경찰', '예훈', '창한', '하운', '지호', '의사', '승찬', '마피아', '마피아']


5일 낮이 밝았습니다.
선량한 시민이 투표로 사망했습니다
생존자 :  ['경찰', '예훈', '창한', '하운', '의사', '승찬', '마피아', '마피아']
5일 밤이 밝았습니다
경찰은 한명을 지목하여 마피아인지 아닌지 확인합니다
경찰이 지목한 사람은 선량한 시민입니다
의사는 살릴 대상을 선택합니다
의사의 도움으로 극적으로 살아났습니다
생존자 :  ['경찰', '예훈', '창한', '하운', '의사', '승찬', '마피아', '마피아']
```

```
6일 낮이 밝았습니다.
우리의 의사가 투표로 사망했습니다
생존자 :  ['경찰', '예훈', '창한', '하운', '승찬', '마피아', '마피아']
6일 밤이 밝았습니다
경찰은 한명을 지목하여 마피아인지 아닌지 확인합니다
경찰이 지목한 사람은 마피아가 맞습니다
선량한 시민이 마피아의 습격으로 사망했습니다
생존자 :  ['경찰', '창한', '하운', '승찬', '마피아', '마피아']


7일 낮이 밝았습니다.
경찰의 도움으로 마피아가 투표로 사망했습니다
생존자 :  ['경찰', '창한', '하운', '승찬', '마피아']
7일 밤이 밝았습니다
경찰은 한명을 지목하여 마피아인지 아닌지 확인합니다
경찰이 지목한 사람은 선량한 시민입니다
우리의 경찰이 마피아의 습격으로 사망했습니다
생존자 :  ['창한', '하운', '승찬', '마피아']


8일 낮이 밝았습니다.
선량한 시민이 투표로 사망했습니다
생존자 :  ['창한', '승찬', '마피아']
8일 밤이 밝았습니다
선량한 시민이 마피아의 습격으로 사망했습니다
생존자 :  ['창한', '마피아']


9일 낮이 밝았습니다.
운좋게도 마피아가 투표로 사망했습니다
생존자 :  ['창한']
게임이 종료되었습니다
citizen 팀의 승리입니다
계속하려면 아무 키나 누르십시오 . . .
```

#아래는 최종결과 test case 중 하나입니다(with smart police, doctor, army, noble)



```
C:₩Windows₩system32₩cmd.exe
시민의 수를 입력하세요12
마피아의 수를 입력하세요3
참가자 :  ['민찬', '예훈', '창한', '하운', '정윤', '민지', '규민', '지호', '시영', '민혁', '승찬', '석문', '서윤', '준철', '기훈']
게임 세팅이 완료되었습니다
마피아 게임 시작합니다
참가자 :  ['민찬', '경찰', '창한', '하운', '군인', '의사', '귀족', '지호', '시영', '민혁', '승찬', '석문', '마피아', '마피아', '마피아']


1일 낮이 밝았습니다.
선량한 시민이 투표로 사망했습니다
생존자 :  ['민찬', '경찰', '창한', '하운', '군인', '의사', '귀족', '지호', '민혁', '승찬', '석문', '마피아', '마피아', '마피아']
1일 밤이 밝았습니다
경찰은 한명을 지목하여 마피아인지 아닌지 확인합니다
경찰이 지목한 사람은 선량한 시민입니다
의사는 살릴 대상을 선택합니다
선량한 시민이 마피아의 습격으로 사망했습니다
생존자 :  ['민찬', '경찰', '창한', '하운', '군인', '의사', '귀족', '지호', '민혁', '승찬', '마피아', '마피아', '마피아']


2일 낮이 밝았습니다.
귀족의 능력으로 투표로 죽지않았습니다 (이제 귀족의 능력은 소멸됩니다)
생존자 :  ['민찬', '경찰', '창한', '하운', '군인', '의사', '귀족', '지호', '민혁', '승찬', '마피아', '마피아', '마피아']
2일 밤이 밝았습니다
경찰은 한명을 지목하여 마피아인지 아닌지 확인합니다
경찰이 지목한 사람은 선량한 시민입니다
의사는 살릴 대상을 선택합니다
선량한 시민이 마피아의 습격으로 사망했습니다
생존자 :  ['민찬', '경찰', '창한', '군인', '의사', '귀족', '지호', '민혁', '승찬', '마피아', '마피아', '마피아']


3일 낮이 밝았습니다.
운좋게도 마피아가 투표로 사망했습니다
생존자 :  ['민찬', '경찰', '창한', '군인', '의사', '귀족', '지호', '민혁', '승찬', '마피아', '마피아']
3일 밤이 밝았습니다
경찰은 한명을 지목하여 마피아인지 아닌지 확인합니다
경찰이 지목한 사람은 마피아가 맞습니다
의사는 살릴 대상을 선택합니다
선량한 시민이 마피아의 습격으로 사망했습니다
생존자 :  ['민찬', '경찰', '창한', '군인', '의사', '귀족', '지호', '민혁', '마피아', '마피아']


4일 낮이 밝았습니다.
경찰의 도움으로 마피아가 투표로 사망했습니다
생존자 :  ['민찬', '경찰', '창한', '군인', '의사', '귀족', '지호', '민혁', '마피아']
4일 밤이 밝았습니다
경찰은 한명을 지목하여 마피아인지 아닌지 확인합니다
경찰이 지목한 사람은 선량한 시민입니다
의사는 살릴 대상을 선택합니다
의사의 도움으로 극적으로 살아났습니다
생존자 :  ['민찬', '경찰', '창한', '군인', '의사', '귀족', '지호', '민혁', '마피아']


5일 낮이 밝았습니다.
우리의 경찰이 투표로 사망했습니다
생존자 :  ['민찬', '창한', '군인', '의사', '귀족', '지호', '민혁', '마피아']
5일 밤이 밝았습니다
의사는 살릴 대상을 선택합니다
의사의 도움으로 극적으로 살아났습니다
생존자 :  ['민찬', '창한', '군인', '의사', '귀족', '지호', '민혁', '마피아']


6일 낮이 밝았습니다.
선량한 시민이 투표로 사망했습니다
```



```
C:₩Windows₩system32₩cmd.exe

6일 낮이 밝았습니다.
선량한 시민이 투표로 사망했습니다
생존자 :  ['민찬', '창한', '군인', '의사', '귀족', '민혁', '마피아']
6일 밤이 밝았습니다
의사는 살릴 대상을 선택합니다
선량한 시민이 마피아의 습격으로 사망했습니다
생존자 :  ['창한', '군인', '의사', '귀족', '민혁', '마피아']


7일 낮이 밝았습니다.
우리의 귀족이 능력을 소멸하여 투표로 사망했습니다
생존자 :  ['창한', '군인', '의사', '민혁', '마피아']
7일 밤이 밝았습니다
의사는 살릴 대상을 선택합니다
군인의 능력으로 마피아의 습격을 견뎌습니다(이제 군인의 능력은 소멸됩니다)
생존자 :  ['창한', '군인', '의사', '민혁', '마피아']


8일 낮이 밝았습니다.
운좋게도 마피아가 투표로 사망했습니다
생존자 :  ['창한', '군인', '의사', '민혁']
게임이 종료되었습니다
citizen 팀의 승리입니다
계속하려면 아무 키나 누르십시오 . . .
```

#통계값 분석 (통계파일.exe는 따로 있습니다 각 case의 데이터는 총 10만번 돌린 결과값입니다)

1. only citizen

대체로 시민의 수가 동일할 때 마피아의 수를 늘리면 시민이 이길 확률이 줄어듦을 확인할 수 있었다. 하지만 마피아의 수가 동일할 때 시민의 수를 늘리면 항상 시민이 이길 확률이 올라간다는 사실이 상식적이지만, 10만번의 데이터 분석 결과 상식에 벗어나는 결과가 나왔다.

예로 마피아의 수가 2명일 때 시민의 수를 6명에서 7명으로 늘렸지만 오히려 시민이 이길 확률이 16퍼센트가량 감소했다.

가만히 들여다보니 시민의 수를 기준으로 짝수일때와 홀수일 때 아예 독립적으로 확률이 움직인다는 생각이 들었다. (이는 뒤의 4가지 케이스에서도 동일하게 일어남). 아예 시민의 수가 짝수일 경우와 홀수일 경우로 나뉘어 두 집단이 독립적으로 존재한다는 생각이 들었다. 확률적으로 접근하면 (시민기준) 짝수 집단과 홀수 집단 각각 다른 확률 분포를 가질 것 같았다.

솔직히 10만번이나 돌렸는데 내 데이터가 잘못된 것 같지는 않았다. 또한 파이썬의 랜덤함수는 괜찮은 함수이기 때문에 문제가 없을 것 같았다. 도대체 왜 이런 상식에 벗어나는 확률 패턴이 나오는 걸까.

생각해보니 오히려 시민의 수가 늘어나면 낮에 투표할 때 시민이 걸려 죽을 확률이 올라간다

6명에서 7명으로 늘게 되면 낮에 시민이 죽을 확률은 오히려 4/6에서 5/7로 증가한다.

또 가만 생각해보니 "투표의 방식"이 실제와 다르기 때문이라는 생각이 들었다.

실제 게임에서는 사실상 낮에 시민의 수와 마피아의 수가 동일하면 마피아의 승리이다.

마피아들이 단합하여 한 시민을 몰아버리면 못해도 동률이 나와 마피아가 투표로 죽을 일이 없기 때문이다. 하지만 이번 프로젝트의 코드 알고리즘은 "투표를 통해 무조건 한 사람이 무작위적으로 죽는다"이다. 또한 "마피아는 서로를 알고 있지만 사람을 죽이는 것은 무작위적이다" 라고 명시되어있다. 즉 이번 프로젝트에서 마피아들은 지능이 없는 ugly maphia에 해당한다고 생각한다.

그래서 마피아가 마피아를 지목하는 상황이 벌어지고 투표의 결과가 동률이 나오는 현상이 배제되는 현상이 생긴다. 결정적으로 이로 인해 마지막에 시민과 마피아 두 명만 남았을때도 사실상 무조건 마피아의 승리이지만 이번 알고리즘에서는 랜덤으로 둘 중 한 명이 죽는다.

이와 같은 경우들이 반영되어 위와 같은 상식에 벗어나는 확률적 경향이 일어난 것 같다.

의도를 바꾸어 투표의 결과가 중복이 되면 그냥 넘어가고, 마피아끼리는 서로 지목하지 못하는 smart maphia의 개념을 새로 도입한다면 보다 현실적인 결과가 나올 것이라는 결론을 내렸다.

아무튼 이번 프로젝트의 의도대로 분석한 결과는 위와 같았다.

이렇듯 모집단이 정확히 어떤 분포를 따르고 어떤 확률적인 수식을 가지는지는 알 수 없었지만 최대한 sample 데이터를 많이 뽑음으로써 population의 paramter를 estimation 할 수 있었다. 영원히 모집단의 확률은 알 수 없겠지만 10만번의 테스트를 거친 샘플 데이터로 최대한 할 수 있는 만큼 추정해보았다.

2. with ugly police

Only citizen의 확률과 비교해 보았을 때 거의 모든 case에 대해 5~15퍼센트 정도만큼 시민이 이길 확률이 올라갔다. 평균적으로 10퍼센트 정도 올라갔다. 중복을 허용하는 경찰이지만 확실히 마피아를 찾게 되고 그 다음날 경찰이 살아있으면 투표로 무조건 마피아가 죽기 때문에 확실히 시민 입장에서 도움이 되었다고 해석할 수 있다.

Vs only citizen : +10 percent

3. with smart police

Ugly police의 확률과 비교해 보았을 때 거의 모든 case에 대해 1~3퍼센트 정도만큼 시민이 이길 확률이 올라갔다. 평균적으로 2퍼센트 정도 올라갔다. 보다 심도 있는 분석을 위해 시민의 수를 20, 40, 100으로 늘리고 마피아의 수를 5 10 30 50으로 늘려서 통계를 내보았다.

확실히 집단(마피아 + 시민)의 수가 커질수록 smart police의 효과가 커졌다. 아무래도 smart police는 한번 검사한 대상은 다시는 검사하지 않으니 집단(특히 마피아)의 수가 커질수록 smart police의 효과가 커지는 것이라고 추정이 된다. 특히 시민이 100명 마피아가 50명일 때 ugly police와 smart police의 효과 차이가 컸다. Ugly의 경우 0.63퍼센트로 거의 0에 가까운 값이 나왔지만, smart의 경우 5.365퍼센트로 ugly의 경우보다 훨씬 높은 값이 나왔다. 역시 마피아의 수가 많을 때 효과가 크다는 사실을 확인할 수 있었다.

Vs Only citizen : + 12 percent

Vs ugly police : +2 percent

4. with smart police, doctor

위의 3번째 경우에 의사가 추가된 경우이다. Smart police만 있을 때의 확률과 비교해 보았을 때, 앞의 경우와는 조금 다른 양상을 보였다.

전체 인원수(시민+ 마피아)가 짝수일때는 +- 0.5~1퍼센트(평균 0.3퍼센트)정도로 거의 효과가 없었지만

전체 인원수(시민 +마피아)가 홀수일때는 4~9퍼센트(평균 7퍼센트)정도 시민이 이길 확률이 올라갔다.

또한 전체 인원수가 점점 커질수록 효과가 미비해지는 경향이 있었다. 예로 시민이 100명이고 마피아가 10,30,50명인 각각의 케이스를 분석해보니 의사가 있고 없고의 효과 차이가 거의 없었다.

우선 인원수가 많아질수록 효과가 줄어드는 이유는 의사가 단 한 명이기 때문이라고 생각한다.

의사의 로직 자체가 전체 인원에서 아무나 한 명 선택하여 살리는 개념이기 때문이기 때문이다. 당연히 선택해야할 집단의 수가 커지면 의사가 target을 살릴 수 있는 가능성이 작아질 것이다.

만약 의사의 수를 늘린다면 효과는 더욱 극명해질 것이라 판단된다.

전체 인원수가 짝수일때와 홀수일 경우 효과가 극명히 갈리는 이유는 앞서 말한 ugly마피아의 성질과 투표를 통해 무조건 한 명은 죽는 알고리즘(예로 표가 2대 2의 동률이 나오는 상황 같은 걸 고려하지 않는다는 뜻. 이로 인해 낮에 마피아 1명과 시민 1명이 남아도 무조건 마피아가 승리하는게 아니라 랜덤으로 한명이 투표로 죽는다)의 영향이라고 생각한다.

또한 의사가 도입되면서 시민보다 마피아의 수가 많은 경우에도 시민이 이길 확률이 생겼음을 확인할 수 있었다. 예를 들어 시민이 4명이고 마피아가 5명일 때 확률이 0이었다가 5.251로 증가하였다. 이는 의사가 최소 한번은 밤에 시민을 살렸다는 것을 의미한다.

(citizen+maphia)의 수가 짝수일 때

Vs Only citizen : + 12 percent

Vs ugly police : +2 percent

Vs smart police : 0 percent

(citizen+maphia)의 수가 홀수일 때

Vs Only citizen : + 19 percent

Vs ugly police : +9 percent

Vs smart police : +7 percent

5. with smart police, doctor, noble, army

앞의 경우에 내가 만든 귀족과 군인이 추가된 경우이다. 역시 전체 시민의 수가 짝수일 경우와 홀수일 경우에 확률에 차이가 있었다.

전체 인원수(시민+ 마피아)가 짝수일때는 0.5~3퍼센트(평균 1퍼센트) 정도 시민이 이길 확률이 올라갔고

전체 인원수(시민 +마피아)가 홀수일때는 5~15퍼센트(평균 10퍼센트) 정도 시민이 이길 확률이 올라갔다.

또한 역시나 전체 인원수가 커질수록 효과가 미비해지는 경향이 있었다.

위의 경우와 마찬가지로 시민이 100명 마피아가 10 30 50 명인 경우를 분석해보니 군인과 귀족이 추가 된 들 하더라도 효과가 거의 없음을 확인할 수 있었다. 역시 마찬가지 이유였다. 시민이 100

명인데 그중 2명에게 특수 역할(시민이 이길 확률을 약간 올려줌)을 부여했을 뿐이기 때문이다. 특수 직업(시민이 이길 확률을 올려주는)을 추가하게 되면 이러한 부분은 조금씩 개선이 될 것이라고 생각한다.

전체 게임참가자의 명수가 짝수와 홀수일 때 확률이 다르게 나오는 것은 위와 마찬가지 이유에서 나온다고 해석하였다.

(citizen+maphia)의 수가 짝수일 때

Vs Only citizen : + 13 percent

Vs ugly police : +3 percent

Vs smart police : +1 percent

Vs smart police & doctor : +1 percent

(citizen+maphia)의 수가 홀수일 때

Vs Only citizen : + 29 percent

Vs ugly police : +19 percent

Vs smart police : +17 percent

Vs smart police & doctor : +10 percent

첨부한 코드는 2가지가 있습니다.

마피아프로젝트_최종완성본_제출용은 실제 게임처럼 결과화면이 나오도록 구현한 코드이고,

마피아프로젝트_최종완성본_분석용은 각 게임을 10만번 돌린 결과가 나오도록 구현한 코드입니다.

첨부한 엑셀파일도 있습니다.

통계파일은 각 case별 통계값들을 엑셀파일로 정리한 파일입니다.

감사합니다.