Cairo University

Faculty of Computers & Artificial Intelligence

Operations Research & Decision Support Dept.

جامعة القاهره

كلية الحاسبات و الذكاء الإصطناعي

قسم بحوث العمليات و دعم القرار

GAME THEORY

TABLE OF CONTENT

PART (1)

A)

This section requires you to discuss three papers about Game Theory's application in real life in a certain industry; we'll talk about it in the economic field and how different companies compete with each other.

To begin, we must understand that economists utilize Game Theory to study economic competition, economic phenomena such as bargaining, mechanism design, auctions, and voting theory, as well as experimental economics, political economy, and behavioral economics. Game Theory provides useful tools for resolving strategy issues. Many company strategies are short- or long-term methods for achieving long-term profitability. With the appropriate strategy, a company may frequently successfully position itself in the market, whereas with the wrong plan, a company would suffer in the long run.

One of the most important games that we're going to discuss is the "Prisoners' Dilemma". This game was formalized by Albert Tucker in 1950, he described it as two criminal gang members who have been captured and imprisoned. Each prisoner is kept in solitary confinement and is unable to communicate with the other. Prosecutors say there isn't enough evidence to prosecute the two on the main accusation. On a lesser charge, they aim to get both of them sentenced to a year in prison. At the same time, the prosecutors make a deal with each prisoner. Each inmate has the option of betraying the other by testifying that the other committed the crime, or cooperating with the other by remaining silent. The offer is:

- If A and B both betray each other, they will each be sentenced to two years in prison.
- If A betrays B but B remains silent, A is released while B is sentenced to three years in prison (and vice versa)

- If both A and B remain silent, they will only be sentenced to one year in prison (on the lesser charge)

The dilemma is normally presented in a payoff matrix like the following:

A&B	**A (Silent)**	**A (Betray)**
A (Silent)	-1, -1	-3,0
A (Betray)	0, -3	-2, -2

FIRST GAME THEORY APPLICATION

In “**Games People Play: Game Theory in Life, Business, and Beyond**” for **Scott P. Stevens**. Out author has a very famous quote that says “*If you keep your eyes and mind open, you’re going to find a lot of other places that our ideas apply*”.

Our author discussed essential issues in Game Theory in this book. In the Game Theory application section that he mentions in his book, cigarette corporations, he discusses why they were pleased when they were banned from broadcasting on television. We'll look at a circumstance when allowing someone to blackmail and it might be in your best interests. We'll see how a $98 stock bid can beat a $102 stock bid. We'll discover how insisting on losing ties could be the difference between winning and losing.

Game Theory can explain why the industry earned an unexpected advantage When tobacco businesses were banned from advertising on television in 1971. Cigarette advertisement is revealed to be a prisoner's dilemma after a simplification. Assume there are two rival cigarette businesses on the market, each with a $500 million advertising budget. If a company advertises, it expands its market by 5%, while if the other company does not advertise, it captures 80% of the market. If neither company advertises or neither advertises, the market is split evenly. Advertising is a prominent strategy for both organizations, even though they each make $1.15

billion from advertising instead of the $1.5 billion they could make if neither is advertised. By restricting advertising, the government removed the "defection" alternatives from the game, leaving the tobacco businesses with their Pareto-optimal payoff.

SECOND GAME THEORY APPLICATION

In this section, we'll study a paper published by Elvis Picardo, a portfolio manager with over 25 years of experience in a variety of capital markets. He was talking about the Prisoner's Dilemma, one of the most well-known games in Game Theory. In his article, he discussed the competition among the most well-known soft drink businesses in the United States. This fight between Coca-Cola and PepsiCo has resulted in countless case studies in business schools.

Consider Coca-Cola vs. PepsiCo, and assume the former is considering lowering the price of its renowned soda. If that happens, Pepsi may have no choice but to follow suit if it wants to keep its cola on the market. Both corporations may see a considerable loss in profits as a result of this.

A price cut by either company could be interpreted as a breach of contract since it violates an implicit agreement to maintain prices high and maximize profits. As a result, if Coca-Cola lowers its price while Pepsi maintains its high price, the former is defecting and the latter is collaborating (by sticking to the spirit of the implicit agreement). Coca-Cola may gain market share and profit margins by selling more colas in this scenario.

Assume the following incremental earnings for Coca-Cola and Pepsi:

- If both companies keep their pricing high, each company's profits will rise by $500 million (because of normal growth in demand).
- If one cuts prices (i.e., defects) while the other does not (cooperates), the former's profits increase by $750 million due to increased market share, while the latter's profits remain unchanged.

- If both companies drop their pricing, the increase in soft drink consumption compensates for the lower price, resulting in a $250 million profit boost for each company.

Then, the Payoff Matrix will be represented as follows:

Coca-Cola & PepsiCo	**PepsiCo (Cooperate)**	**PepsiCo (Cooperate)**
Coca-Cola (Cooperate)	500, 500	0, 750
Coca-Cola (Defect)	750, 0	250, 250

THIRD GAME THEORY APPLICATION

In the last section, we are going to discuss one of the most famous competitions in the world between two companies, Lyft and Uber.

As an introduction, we are going to discuss some details about their current competition. Lyft stock increased 8.7% on its first day of trading on March 29, 2019. It's a once-in-a-lifetime opportunity for investors to participate in the largest and most high-profile technology IPO since Snap's launch in 2017.

According to the Wall Street Journal, Lyft had a difficult time when its major competitor Uber tried to cut off its cash supply from private equity and venture capital organizations. However, Lyft's success has boded well for Uber, which will begin its IPO in April.

In recent years, Lyft has been able to chip away at Uber's supremacy, particularly in terms of finance and business strategy. According to Second Measure, as reported by the Wall Street Journal, the company has increased its market share in the United States from 15% to about 30% in the last three years. Uber controls around 70% of the market.

Lyft reported that it has approximately 39 percent of the US market in December 2018, up from 22 percent in 2016, which is a significant increase given that the US is also where Uber operates. The company has exhibited in its roadshow that the industry was ripe for a “natural monopoly” in which a company could “set price to maximize profits”. However, neither Lyft nor Uber have ever had a natural monopoly scenario. Uber and Lyft have found themselves in a classic prisoner's dilemma situation. Because of their primary competitive strategy, ride-hailing businesses must continue to subsidize drivers and users to gain market dominance. If one of them stopped doing that, it could generate money, but it would almost certainly fail.

According to the Game Theory Prisoner’s Dilemma, the payoff matrix will be as follows:

Lyft & Uber	**Uber (Low)**	**Uber (High)**
Lyft (Low)	**Low, Low**	**Low, High**
Lyft (High)	**High, Low**	**High, High**

The equilibrium result has been (High, High), implying that both parties are constantly burning money to compete with each other and are afraid of losing market share. In such cases, the party with the most money usually wins the game in the end. However, because Uber's brand was tarnished by PR disasters in early 2017, which led to the resignation of its then-CEO, Lyft was able to grow by capitalizing on that opportunity and successfully raising USD 600 million to seize market share at the time. Uber has made numerous mistakes in terms of corporate culture, and Lyft has eventually established itself as a more ethical company. As the market has stabilized, the two companies have reverted to a duopoly situation. They're still losing money on each ride, and they can't make money as long as they don't want to see their market eat them.

B)

In this part, we are going to analyze a strategic situation and try to apply one of the Game Theory approaches to it.

In both intervals before and after the COVID-19 pandemic, many organizations faced a difficult challenge that they were unprepared to address: increasing employee isolation and loneliness.

PricewaterhouseCoopers PwC reported in a 2019 study that "close work relationships boost employee satisfaction by 50%, and people with a best friend at work are seven times more likely than others to engage fully in their work." However, new models of flexible working can reduce opportunities for interactions and relationships in the workplace today. Only half of the respondents in a survey at the time agreed that their employers had successfully developed a robust virtual social platform. This has gotten worse in recent years. Many companies' entire staffs quickly became remote, and the days of team lunches, onsite gyms, happy hours, and hallway chats vanished. That company culture vanished overnight. Even though some people returned to work several weeks or months later, many others did not. As companies implement permanent remote or hybrid working environments, there are fewer opportunities to build relationships with colleagues in person.

According to CNBC, one of the reasons so many people choose to leave their jobs is the loss of work friendships. And success and creativity suffer among those who stay.

In a recent study, Yasin Rofcanin, a professor of management at the University of Bath in the United Kingdom, and a group of colleagues discovered that friendship among coworkers is the most important factor in improving employee performance. Isolation has the greatest impact on mental and emotional health. Stress and anxiety are inextricably linked to feelings of isolation. Colleagues may struggle to find the resiliency they require to face each workday if they do not have others to lean on.

Imperative, a peer coaching platform, investigated the state of work relationships in its 2022 Workforce Purpose Index. Nearly half (46%) of those polled said it's difficult to make work friends, and more than half (57%) said their managers aren't helping. But there is some good news. The vast majority of workers (87%) stated that they are willing to devote more time to developing these relationships.

Despite the current challenges, organizations can take steps to facilitate and encourage the formation of work relationships. Peer coaching is a key method for empowerment in which two colleagues help each other reflect on experiences, offer support, build skills, and match their work to their sense of purpose.

Conclusion

As a result of the preceding, we now have the following information with statistics regarding the healthy relationships between employees inside the company:

- When an employee believes he or she has a positive working relationship with others, he or she will be 50% more satisfied than the average.
- When an employee has a best friend at work, he will be seven times (7X) more committed and engaged at work.
- 46% of employees find it difficult to form positive relationships with their coworkers and teams.
- 57% of employees believe their bosses don't assist them in forming friendly ties at work.
- Employees are ready to form good relationships with their coworkers in 87%of cases.

We can conclude from the foregoing that it is in the company's best interests to improve internal relations among its employees so that they are more committed to their work and duties.

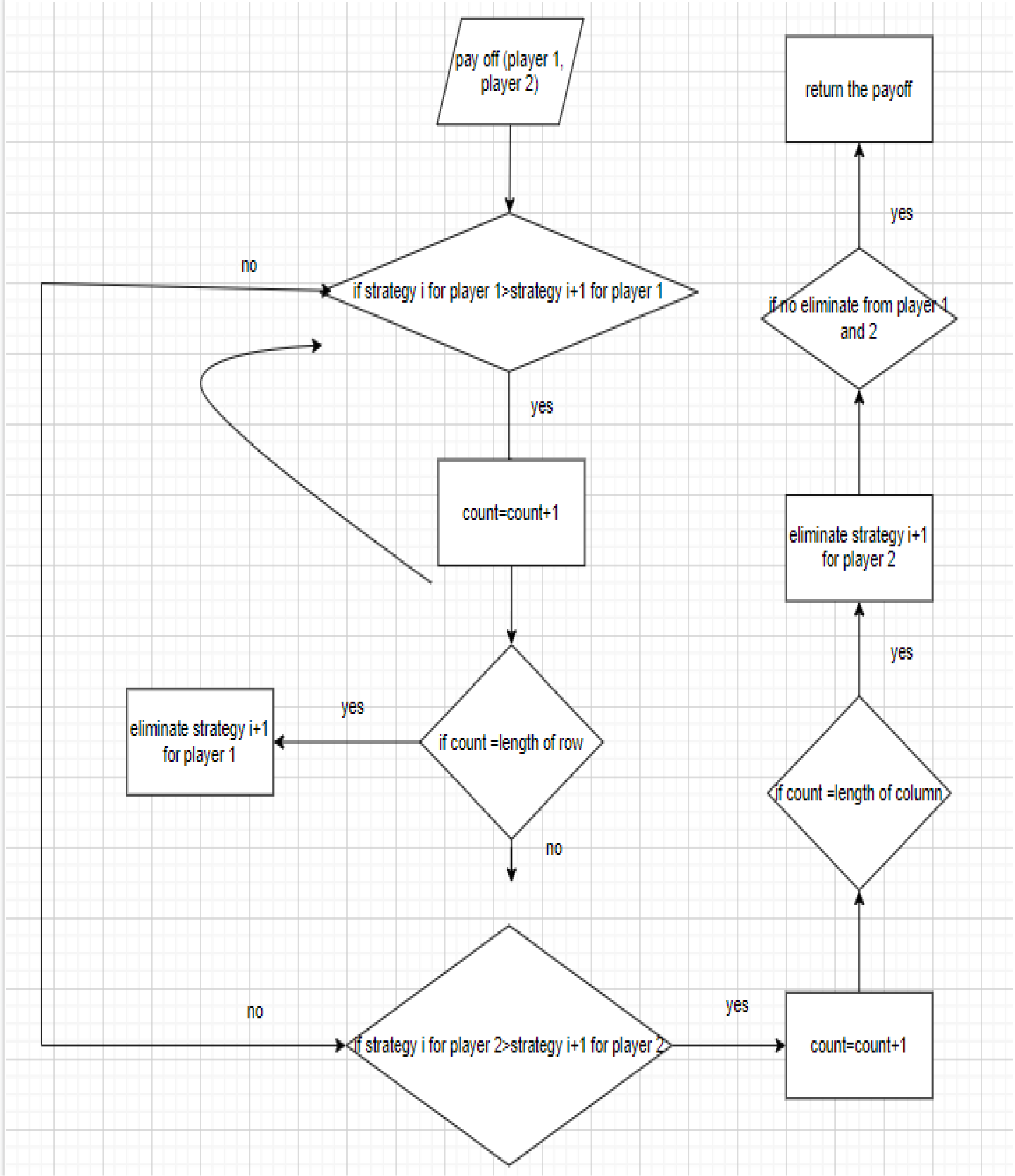
We'll assume that managers try their hardest (High) to enhance internal interactions among their employees in order to increase work quality and that if they don't, they'll suffer performance issues resulting from the employees' isolation. And we'll suppose that the employees will do their best (High) to solve it so that they are more satisfied, but if they don't work on their difficulties to improve their internal relations, they will fail and be under more stress.

Managers & Employees	**Managers (Low)**	**Managers (High)**
Employees (Low)	Low, Low	Low, High
Employees (High)	High, Low	High, High

The equilibrium result was (High, High), meaning that both parties must address present difficulties in order to improve both the company's state and the contentment and comfort of its personnel.

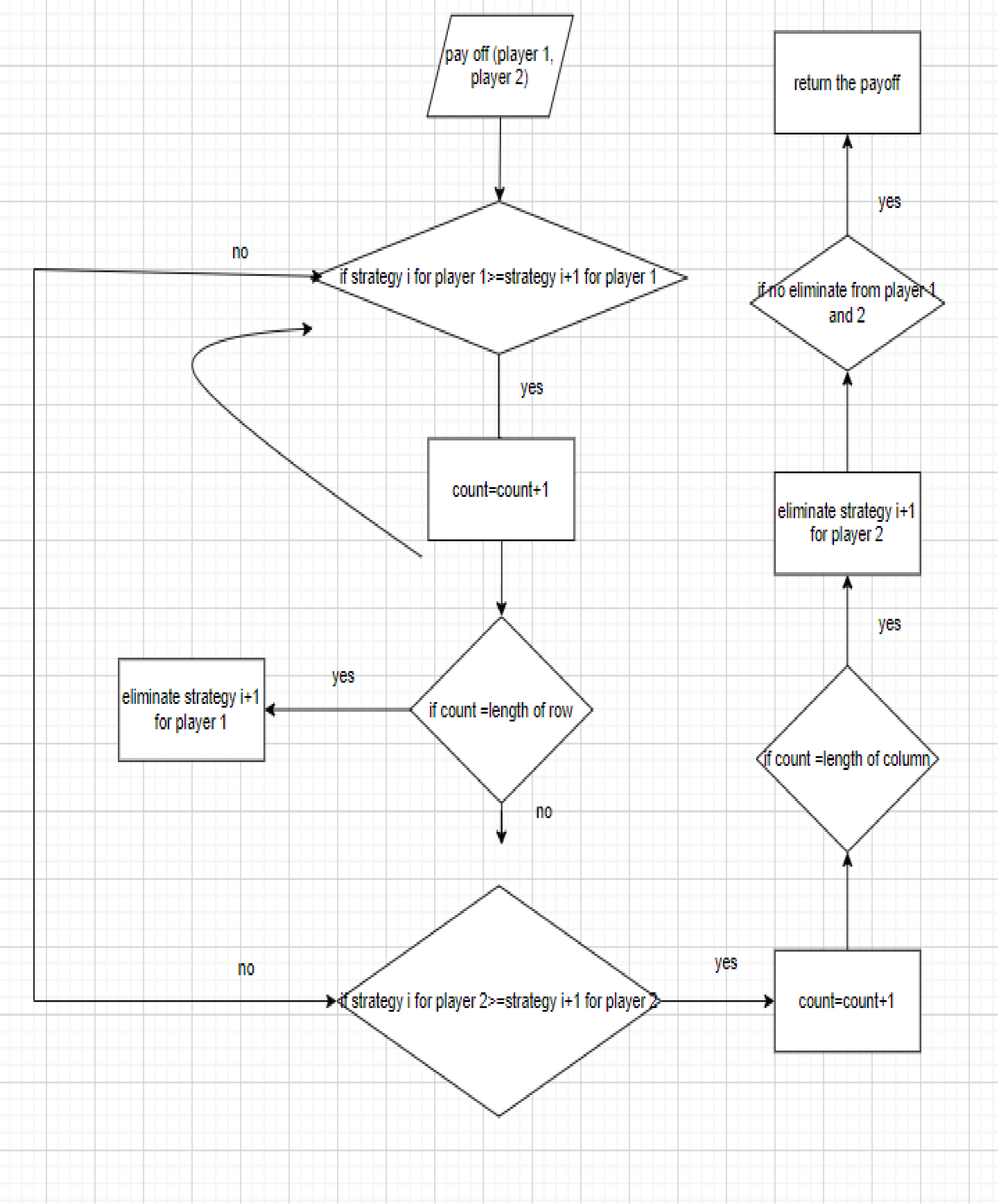
PART (2)

1. A strictly dominated strategy is a strategy that always delivers a worse outcome than an alternative strategy, regardless of what strategy the opponent chooses.

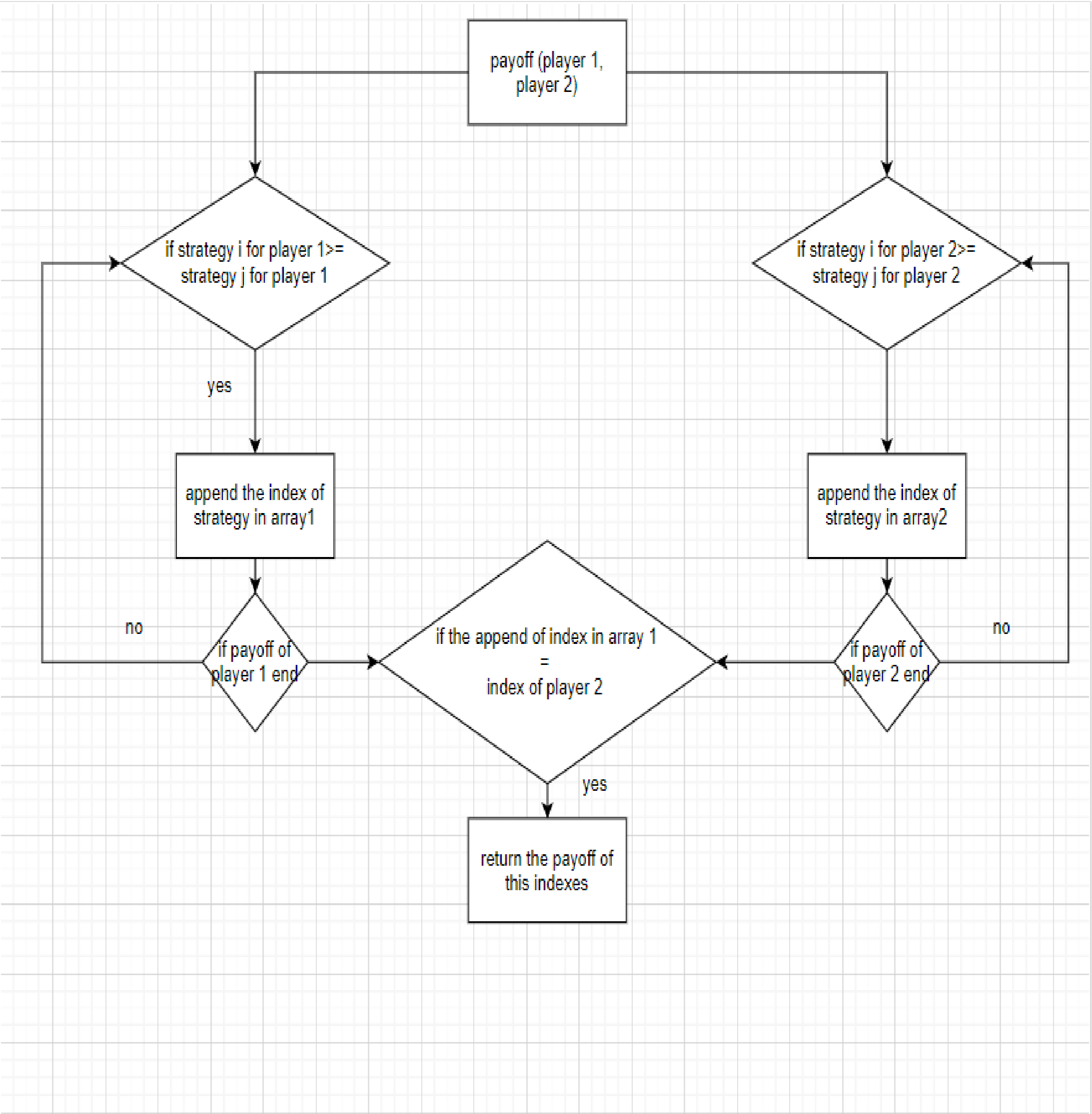


pay off (player 1, player 2)
return the payoff
yes
no
if strategy i for player 1>strategy i+1 for player 1
if no eliminate from player 1 and 2
yes
count=count+1
eliminate strategy i+1 for player 2
yes
yes
eliminate strategy i+1 for player 1
if count =length of row
if count =length of column
no
no
yes
if strategy i for player 2>strategy i+1 for player 2
count=count+1

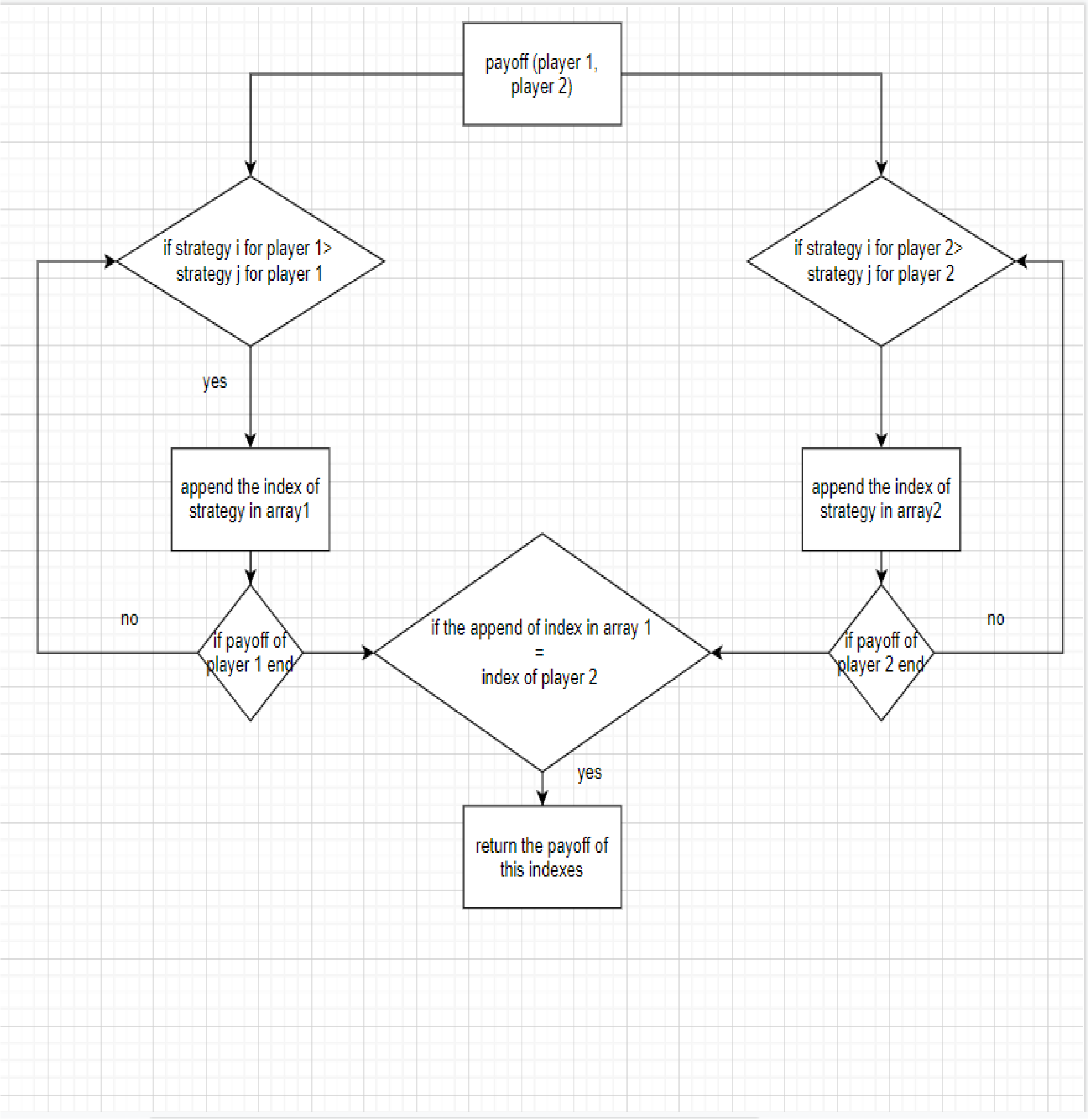
2. A weakly dominated strategy is a strategy that delivers an equal or worse outcome than an alternative strategy.


pay off (player 1, player 2)
return the payoff
yes
no
if strategy i for player 1>=strategy i+1 for player 1
if no eliminate from player 1 and 2
yes
count=count+1
eliminate strategy i+1 for player 2
yes
yes
eliminate strategy i+1 for player 1
if count =length of row
if count =length of column
no
no
yes
if strategy i for player 2>=strategy i+1 for player 2
count=count+1

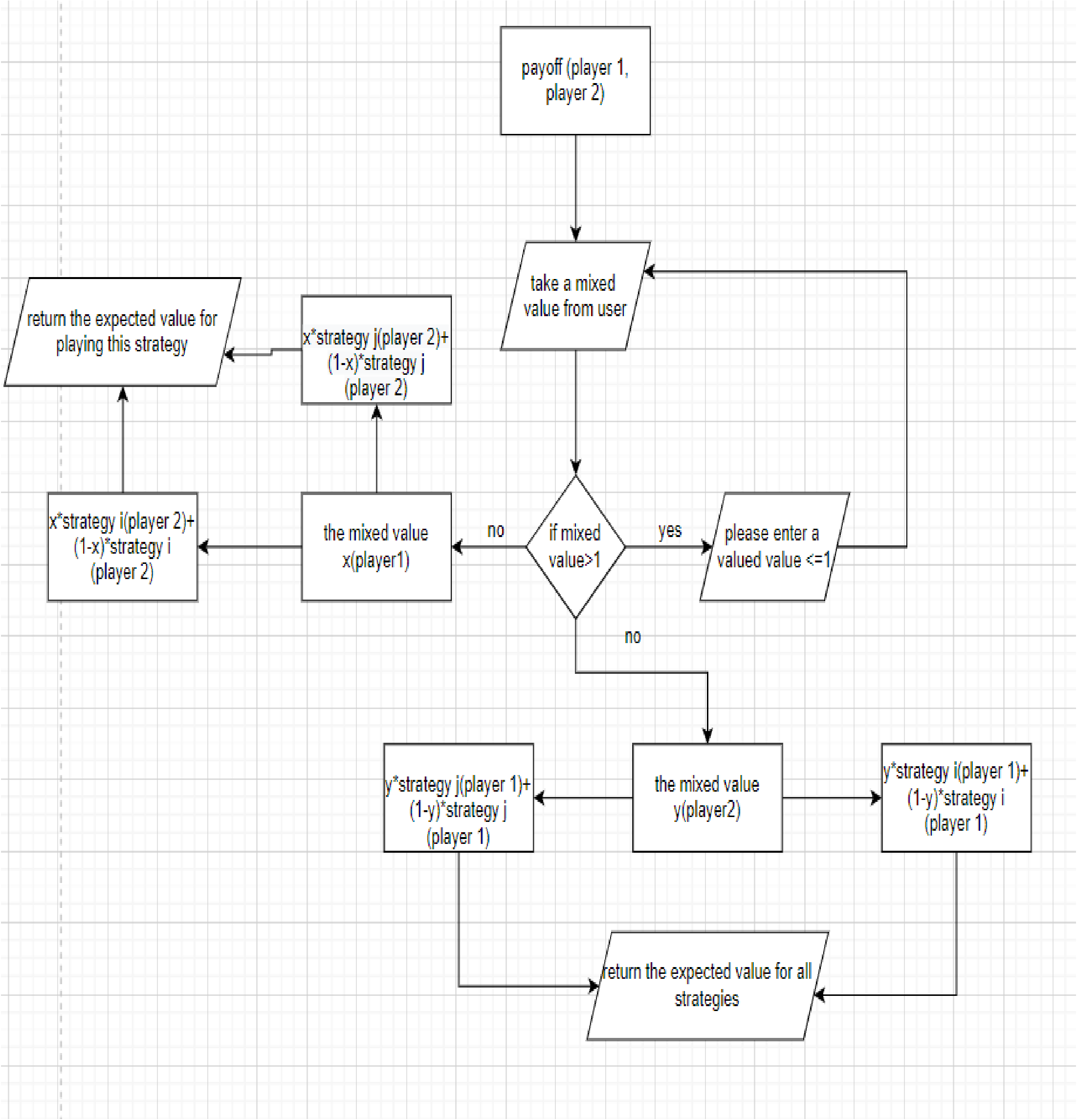
3. A pure-strategic approach When every other player j follows aj, Nash equilibrium is an action profile with the feature that no one player I may receive a larger reward by doing an action other than ai. A pure-strategic approach When every other player j follows aj, Nash equilibrium is an action profile with the feature that no one player I may receive a larger reward by doing an action other than ai.


payoff (player 1, player 2)
if strategy i for player 1>= strategy j for player 1
if strategy i for player 2>= strategy j for player 2
yes
append the index of strategy in array1
append the index of strategy in array2
no
if payoff of player 1 end
if the append of index in array 1 = index of player 2
if payoff of player 2 end
no
yes
return the payoff of this indexes

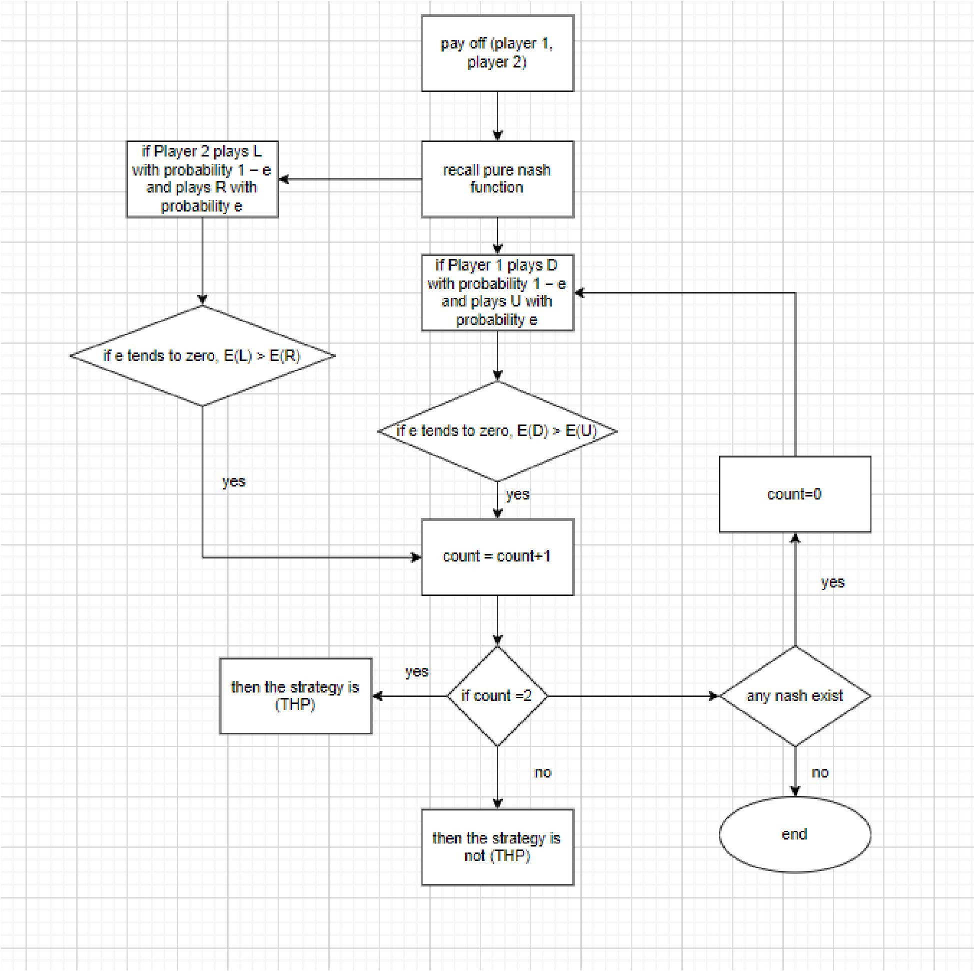
4. strict Nash equilibrium requires that all such deviations be costly.


payoff (player 1, player 2)
if strategy i for player 1> strategy j for player 1
if strategy i for player 2> strategy j for player 2
yes
append the index of strategy in array1
append the index of strategy in array2
no
if payoff of player 1 end
if the append of index in array 1 = index of player 2
if payoff of player 2 end
no
yes
return the payoff of this indexes

5. Expected payoff: Each player calculates what he will play by calculating the amount of the other player's game in certain proportions.

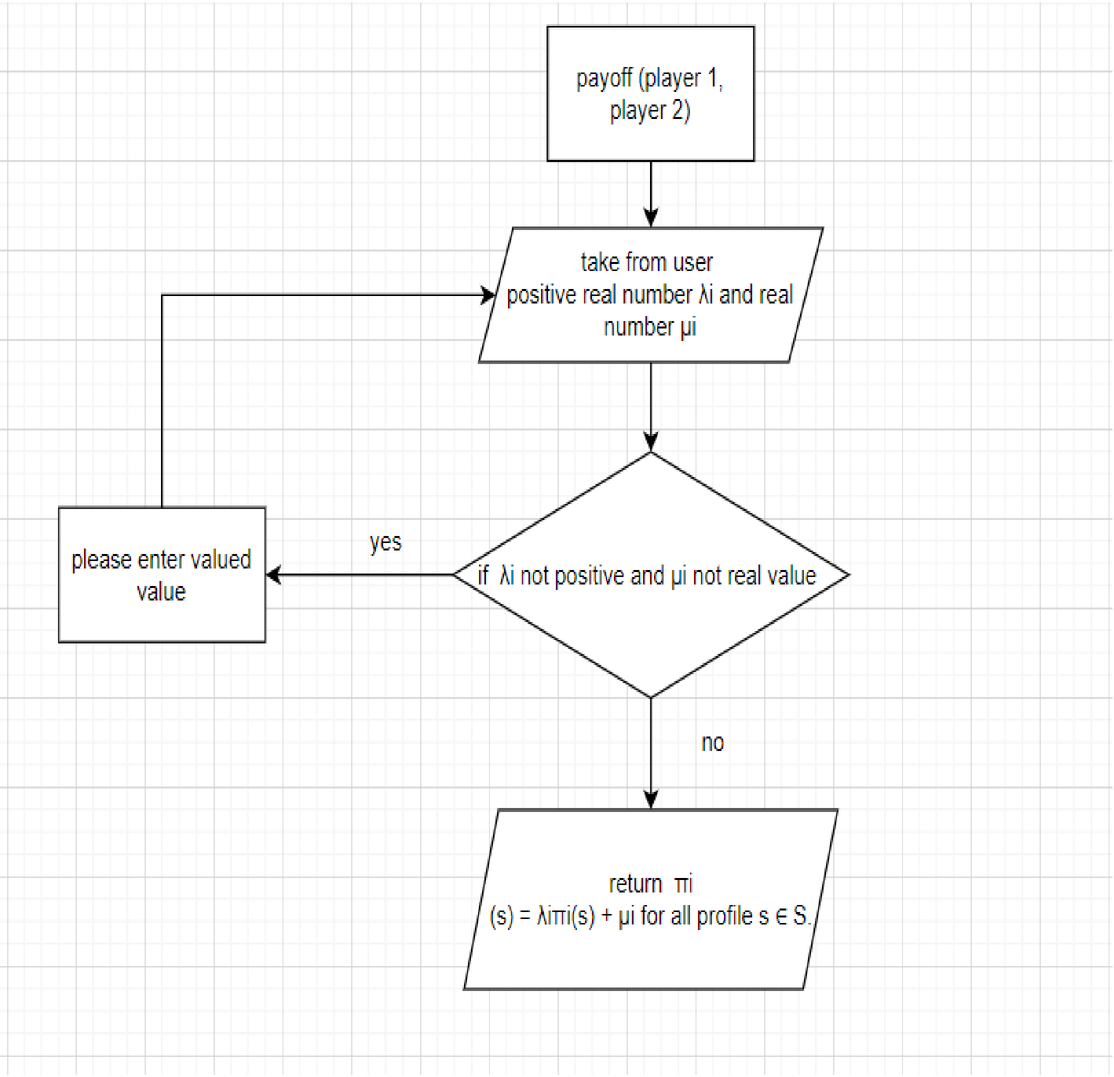

payoff (player 1, player 2)
take a mixed value from user
return the expected value for playing this strategy
x*strategy j(player 2)+ (1-x)*strategy j (player 2)
if mixed value>1
yes
please enter a valued value <=1
no
x*strategy i(player 2)+ (1-x)*strategy i (player 2)
the mixed value x(player1)
no
the mixed value y(player2)
y*strategy j(player 1)+ (1-y)*strategy j (player 1)
y*strategy i(player 1)+ (1-y)*strategy i (player 1)
return the expected value for all strategies

6. The trembling hand Perfect equilibrium, as defined in game theory, is a scenario or state that takes into account the probability of a player making an unintentional move by accident. The chances of this sort of game happening are quite slim, thus deciding whether or not to use this notion in such a situation might be difficult. A refinement of the Nash equilibrium gave rise to this notion.


pay off (player 1, player 2)
recall pure nash function
if Player 2 plays L with probability 1 – e and plays R with probability e
if Player 1 plays D with probability 1 – e and plays U with probability e
if e tends to zero, E(L) > E(R)
if e tends to zero, E(D) > E(U)
yes
yes
count=0
count = count+1
yes
then the strategy is (THP)
yes
if count =2
any nash exist
no
no
then the strategy is not (THP)
end

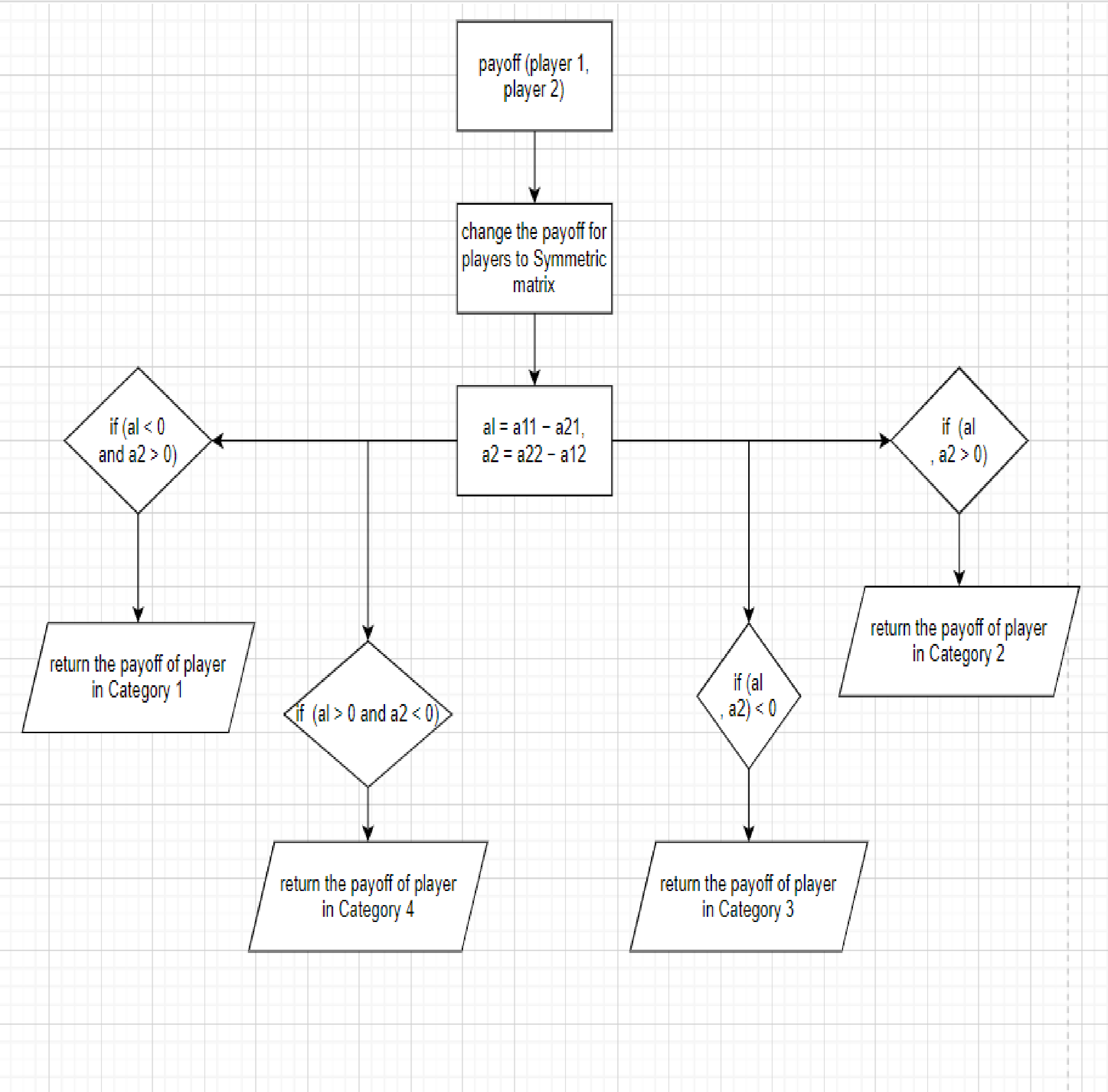
7. Positive affine transformation: The inferred preference order of a utility function that can be subjected to a positive affine transformation without changing it. The modified utility function U* = a + bU is obtained by applying a

positive affine transformation to the initial utility function U, where b > 0. If the utility functions U and U* describe the same set of underlying preferences, the utility function U is cardinal. An anticipated utility function is an example of cardinal utility.


payoff (player 1, player 2)
take from user positive real number λi and real number μi
if λi not positive and μi not real value
yes
please enter valued value
no
return πi (s) = λiπi(s) + μi for all profile s ϵ S.

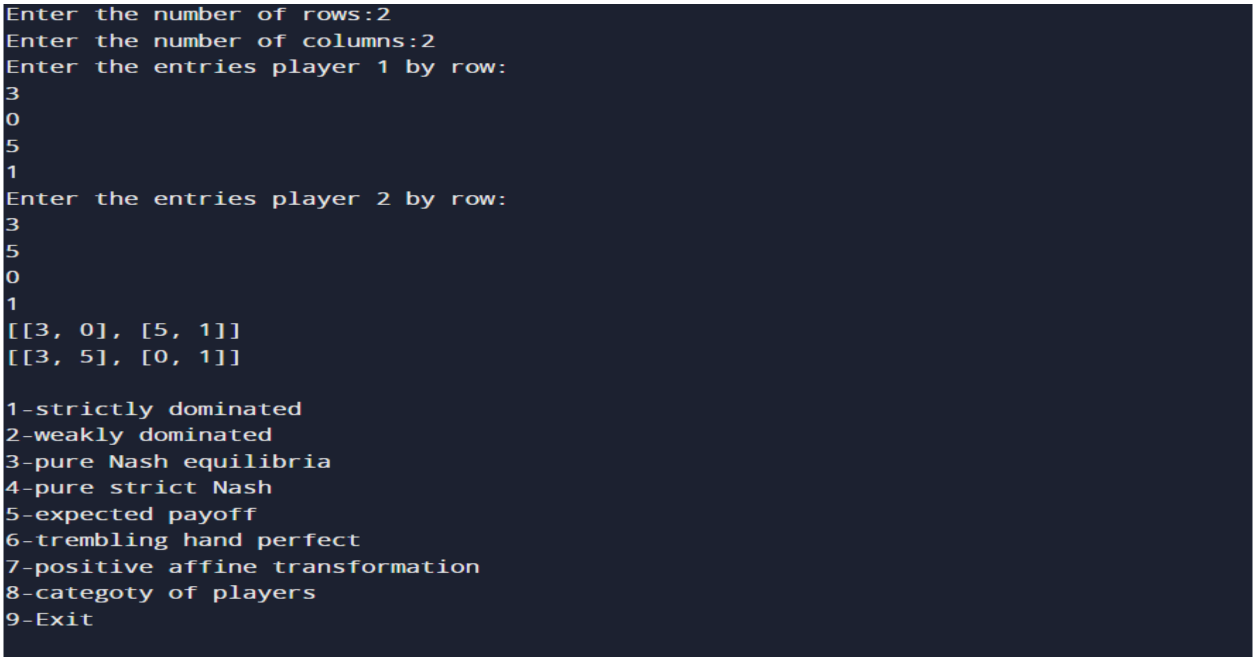
8. Symmetric game: A symmetric two-player game G = (I, S, π), one assumes: Two player positions, Each position has the same number of pure strategies, The payoff to any strategy is independent of which player position it is applied

to and change the matrix to symmetric matrix by: al = a11 − a21 and a2 = a22 − a12.

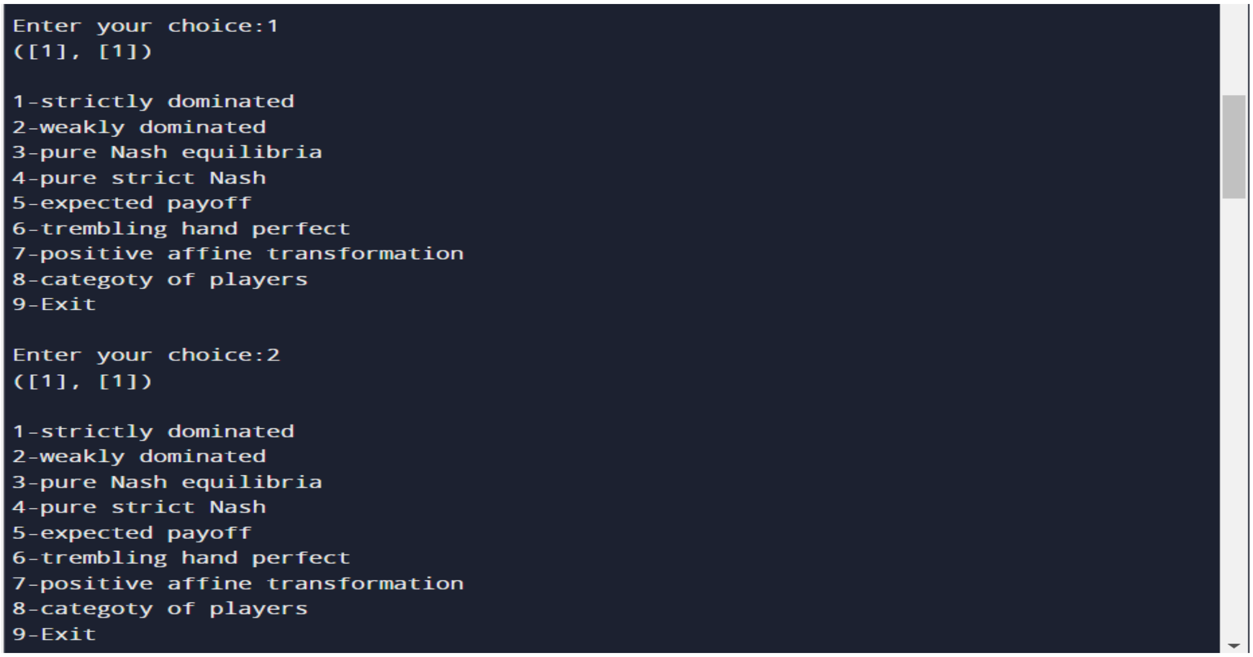

payoff (player 1, player 2)
change the payoff for players to Symmetric matrix
al = a11 − a21, a2 = a22 − a12
if (al < 0 and a2 > 0)
if (al , a2 > 0)
return the payoff of player in Category 1
if (al > 0 and a2 < 0)
if (al , a2) < 0
return the payoff of player in Category 2
return the payoff of player in Category 4
return the payoff of player in Category 3

Code run:

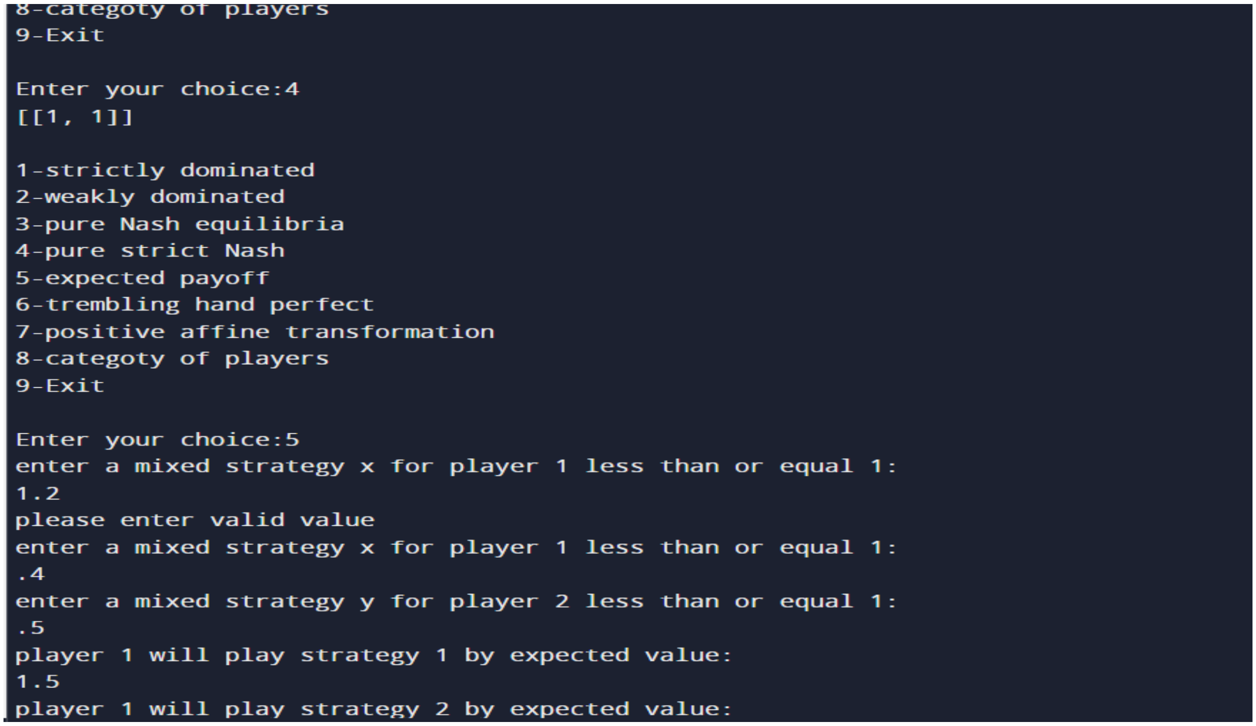
Example 1: player 1 = $\begin{matrix} 3 & 0 \\ 5 & 1 \end{matrix}$, player 2 = $\begin{matrix} 3 & 5 \\ 0 & 1 \end{matrix}$



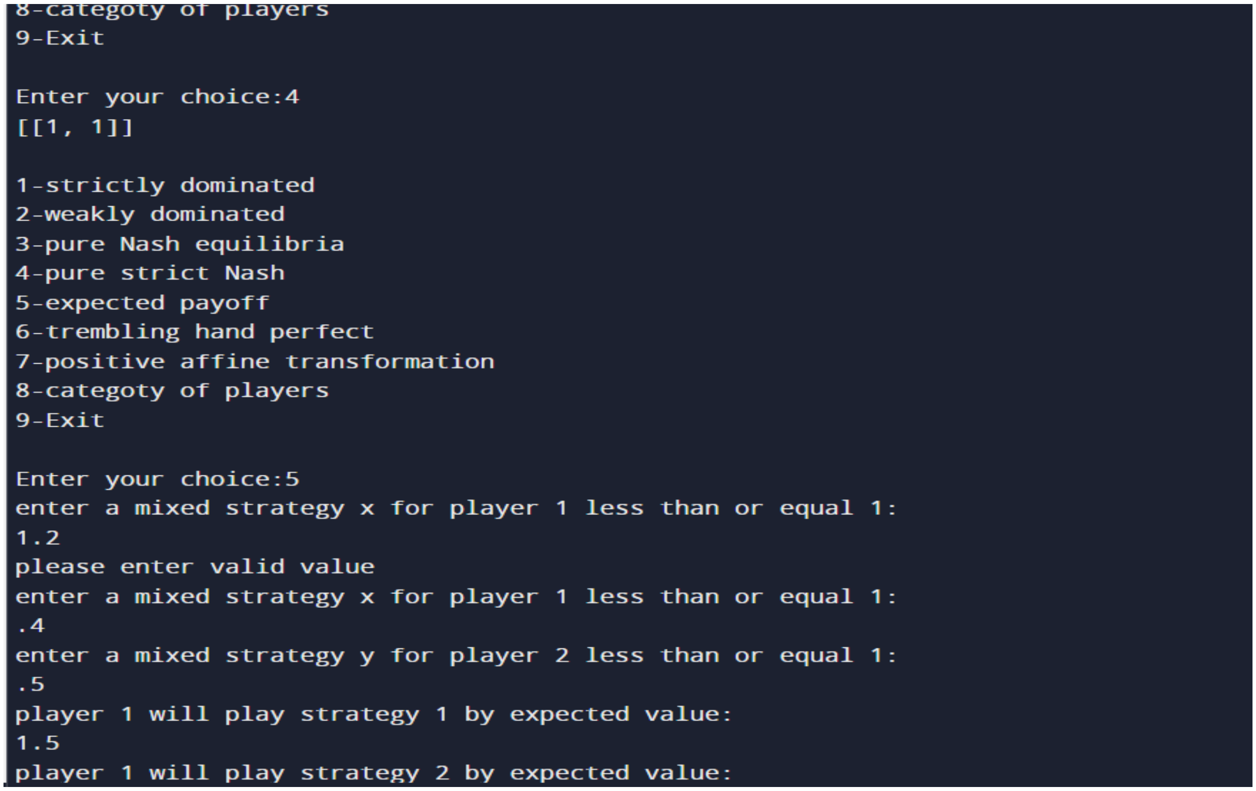
```
Enter the number of rows:2
Enter the number of columns:2
Enter the entries player 1 by row:
3
0
5
1
Enter the entries player 2 by row:
3
5
0
1
[[3, 0], [5, 1]]
[[3, 5], [0, 1]]

1-strictly dominated
2-weakly dominated
3-pure Nash equilibria
4-pure strict Nash
5-expected payoff
6-trembling hand perfect
7-positive affine transformation
8-categoty of players
9-Exit
```



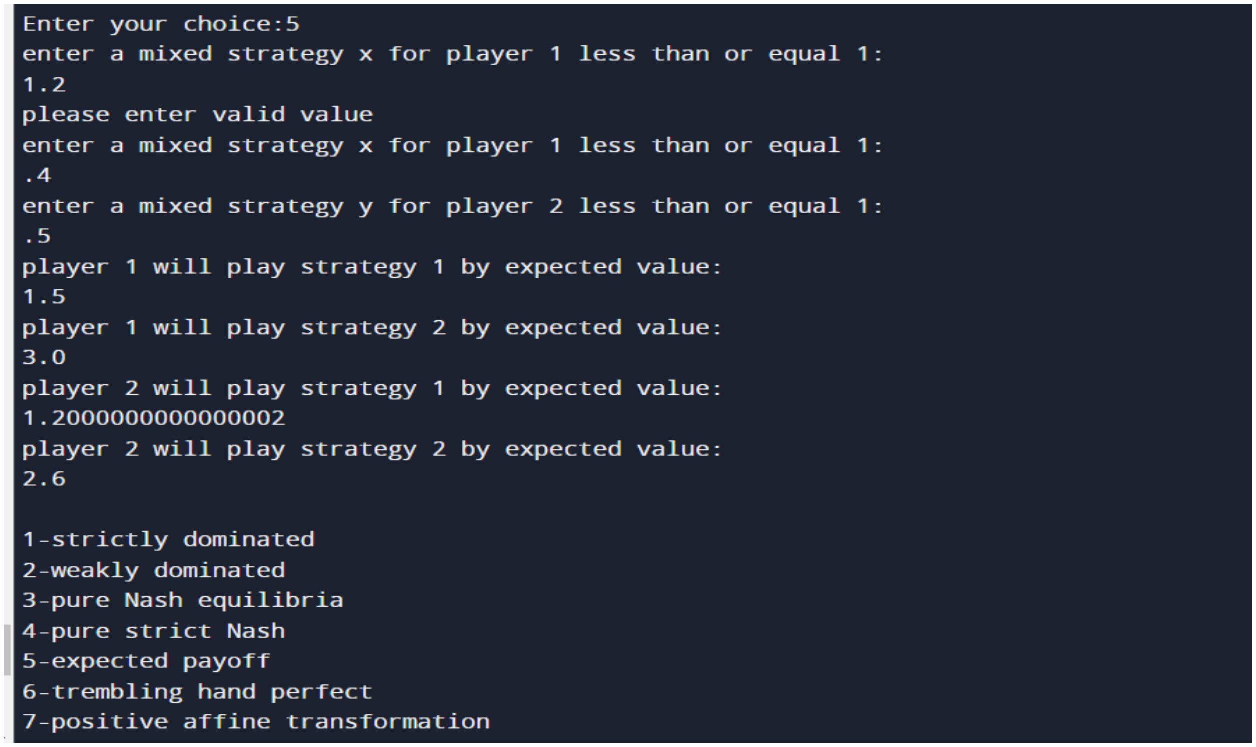
```
Enter your choice:1
([1], [1])

1-strictly dominated
2-weakly dominated
3-pure Nash equilibria
4-pure strict Nash
5-expected payoff
6-trembling hand perfect
7-positive affine transformation
8-categoty of players
9-Exit

Enter your choice:2
([1], [1])

1-strictly dominated
2-weakly dominated
3-pure Nash equilibria
4-pure strict Nash
5-expected payoff
6-trembling hand perfect
7-positive affine transformation
8-categoty of players
9-Exit
```



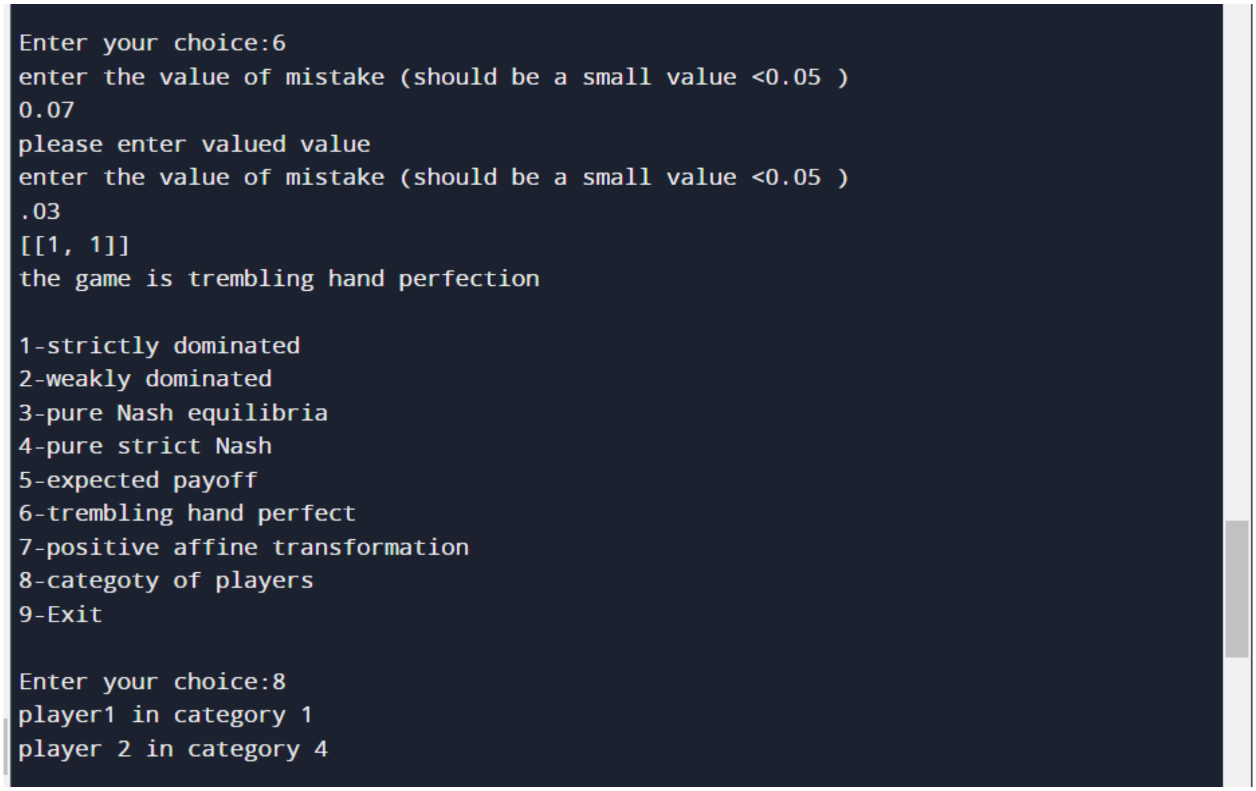
```
8-categoty of players
9-Exit

Enter your choice:4
[[1, 1]]

1-strictly dominated
2-weakly dominated
3-pure Nash equilibria
4-pure strict Nash
5-expected payoff
6-trembling hand perfect
7-positive affine transformation
8-categoty of players
9-Exit

Enter your choice:5
enter a mixed strategy x for player 1 less than or equal 1:
1.2
please enter valid value
enter a mixed strategy x for player 1 less than or equal 1:
.4
enter a mixed strategy y for player 2 less than or equal 1:
.5
player 1 will play strategy 1 by expected value:
1.5
player 1 will play strategy 2 by expected value:
```



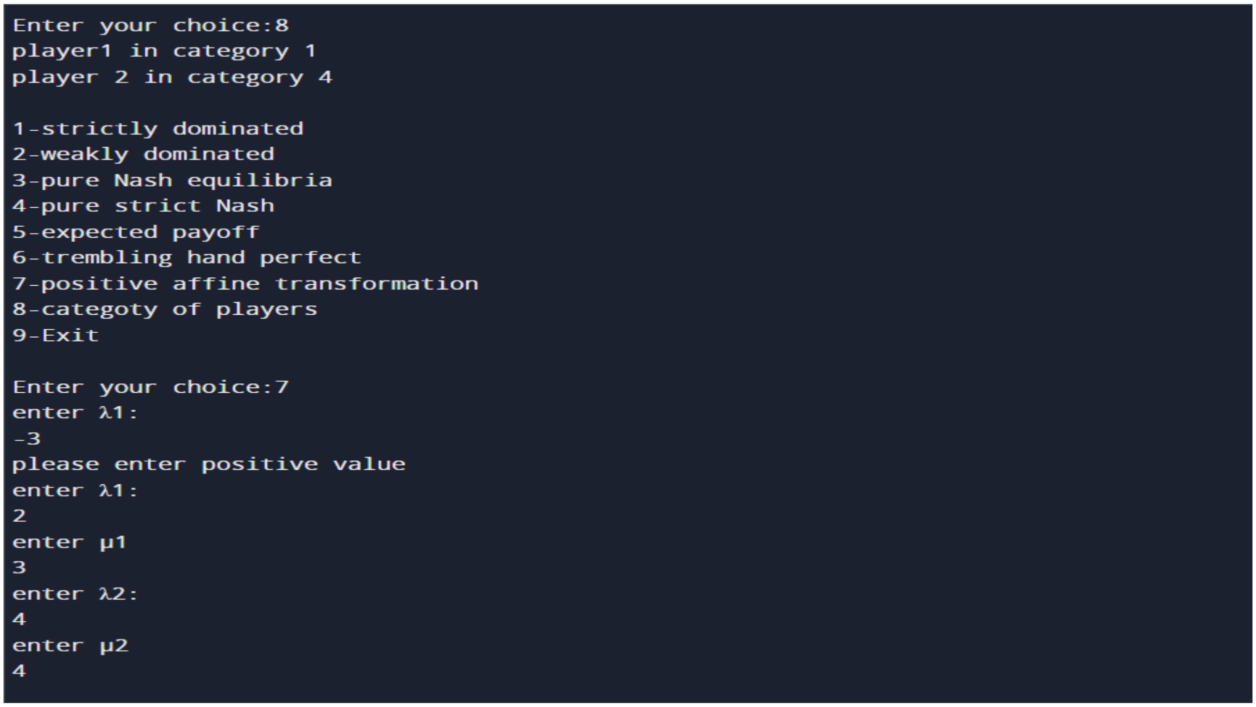
```
8-categoty of players
9-Exit

Enter your choice:4
[[1, 1]]

1-strictly dominated
2-weakly dominated
3-pure Nash equilibria
4-pure strict Nash
5-expected payoff
6-trembling hand perfect
7-positive affine transformation
8-categoty of players
9-Exit

Enter your choice:5
enter a mixed strategy x for player 1 less than or equal 1:
1.2
please enter valid value
enter a mixed strategy x for player 1 less than or equal 1:
.4
enter a mixed strategy y for player 2 less than or equal 1:
.5
player 1 will play strategy 1 by expected value:
1.5
player 1 will play strategy 2 by expected value:
```



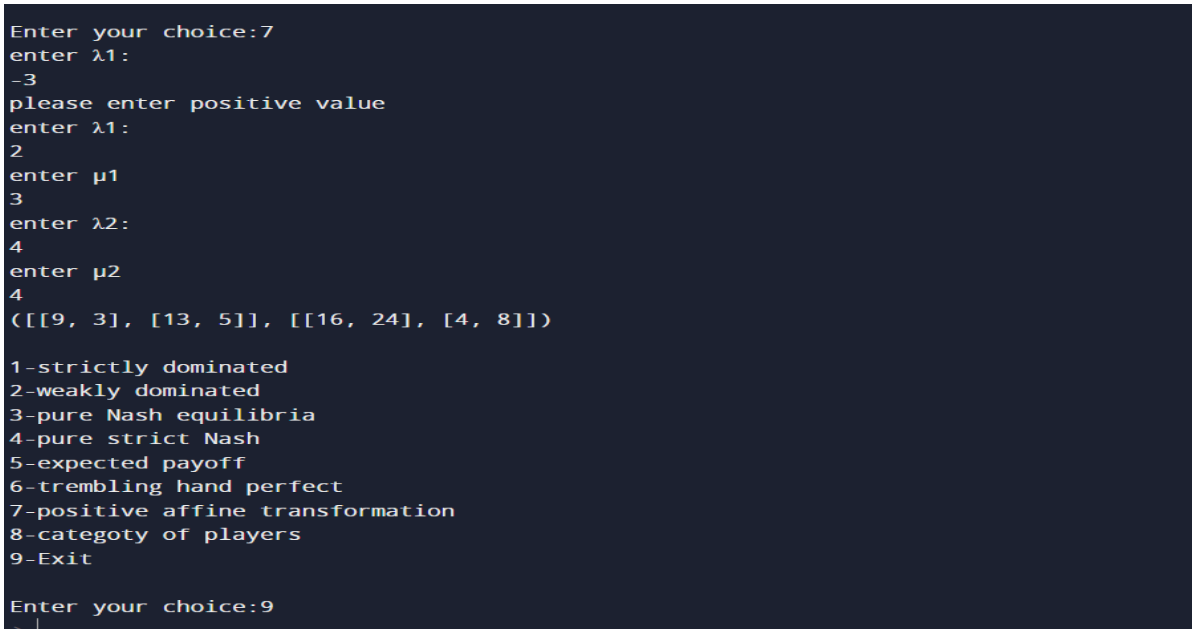
```
Enter your choice:5
enter a mixed strategy x for player 1 less than or equal 1:
1.2
please enter valid value
enter a mixed strategy x for player 1 less than or equal 1:
.4
enter a mixed strategy y for player 2 less than or equal 1:
.5
player 1 will play strategy 1 by expected value:
1.5
player 1 will play strategy 2 by expected value:
3.0
player 2 will play strategy 1 by expected value:
1.2000000000000002
player 2 will play strategy 2 by expected value:
2.6

1-strictly dominated
2-weakly dominated
3-pure Nash equilibria
4-pure strict Nash
5-expected payoff
6-trembling hand perfect
7-positive affine transformation
```



```
Enter your choice:6
enter the value of mistake (should be a small value <0.05 )
0.07
please enter valued value
enter the value of mistake (should be a small value <0.05 )
.03
[[1, 1]]
the game is trembling hand perfection

1-strictly dominated
2-weakly dominated
3-pure Nash equilibria
4-pure strict Nash
5-expected payoff
6-trembling hand perfect
7-positive affine transformation
8-categoty of players
9-Exit

Enter your choice:8
player1 in category 1
player 2 in category 4
```

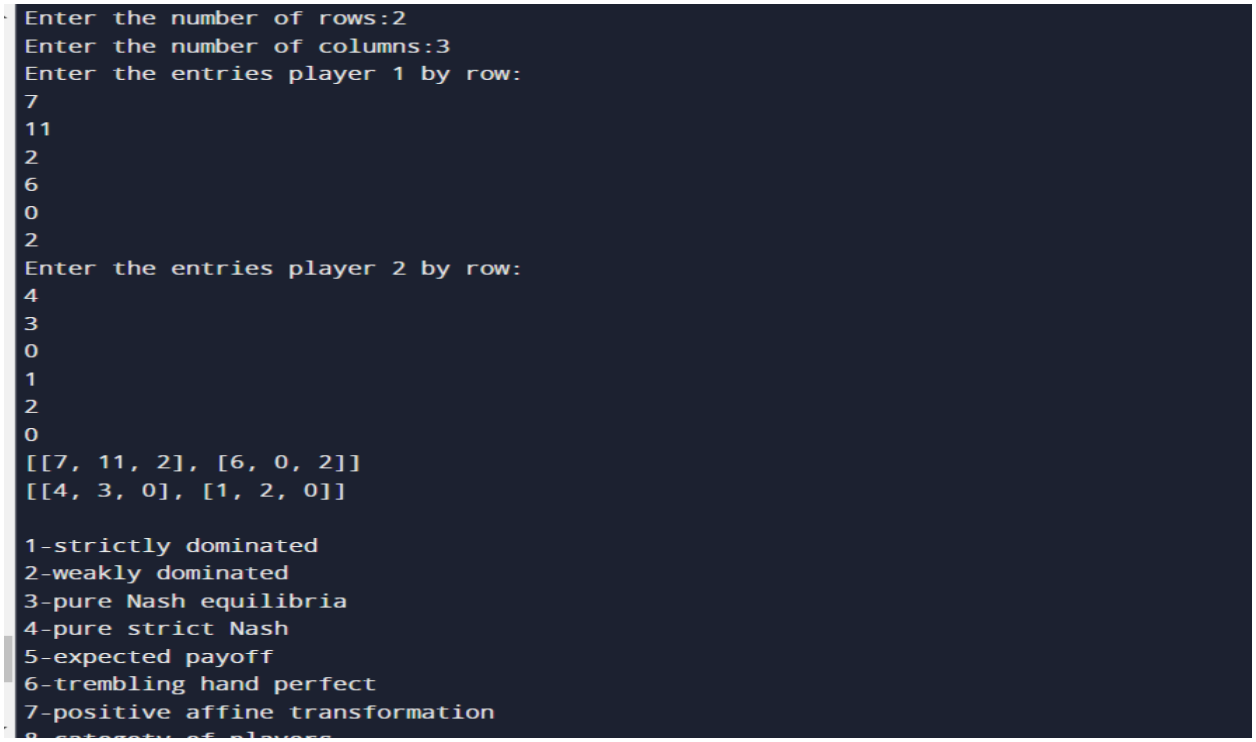


```
Enter your choice:8
player1 in category 1
player 2 in category 4

1-strictly dominated
2-weakly dominated
3-pure Nash equilibria
4-pure strict Nash
5-expected payoff
6-trembling hand perfect
7-positive affine transformation
8-categoty of players
9-Exit

Enter your choice:7
enter λ1:
-3
please enter positive value
enter λ1:
2
enter μ1
3
enter λ2:
4
enter μ2
4
```

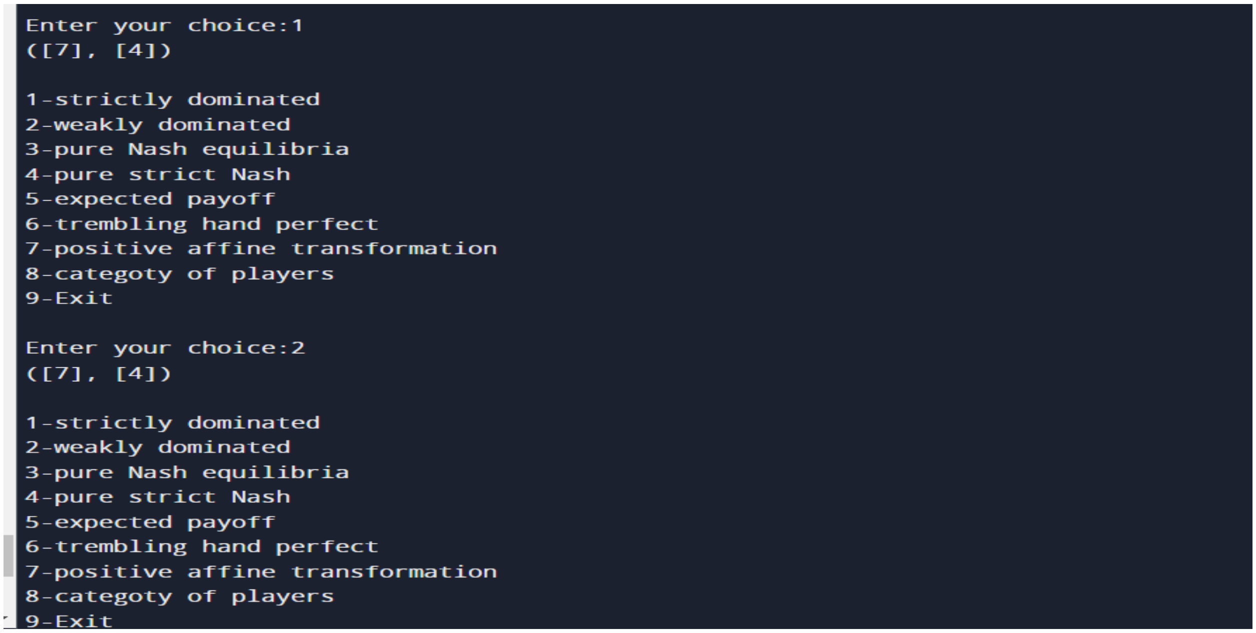


```
Enter your choice:7
enter λ1:
-3
please enter positive value
enter λ1:
2
enter μ1
3
enter λ2:
4
enter μ2
4
([[9, 3], [13, 5]], [[16, 24], [4, 8]])

1-strictly dominated
2-weakly dominated
3-pure Nash equilibria
4-pure strict Nash
5-expected payoff
6-trembling hand perfect
7-positive affine transformation
8-categoty of players
9-Exit

Enter your choice:9
```

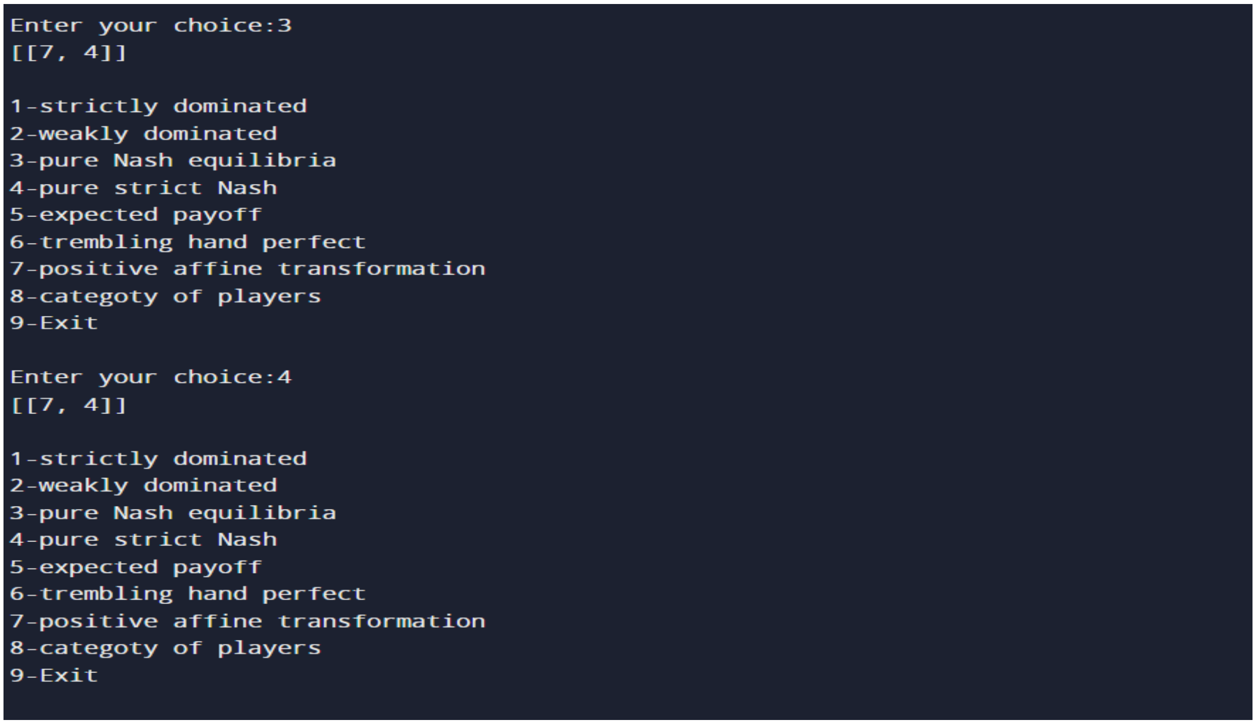
Example 2: player 1 = $\begin{matrix} 7 & 11 & 2 \\ 6 & 0 & 2 \end{matrix}$, player 2 = $\begin{matrix} 4 & 3 & 0 \\ 1 & 2 & 0 \end{matrix}$



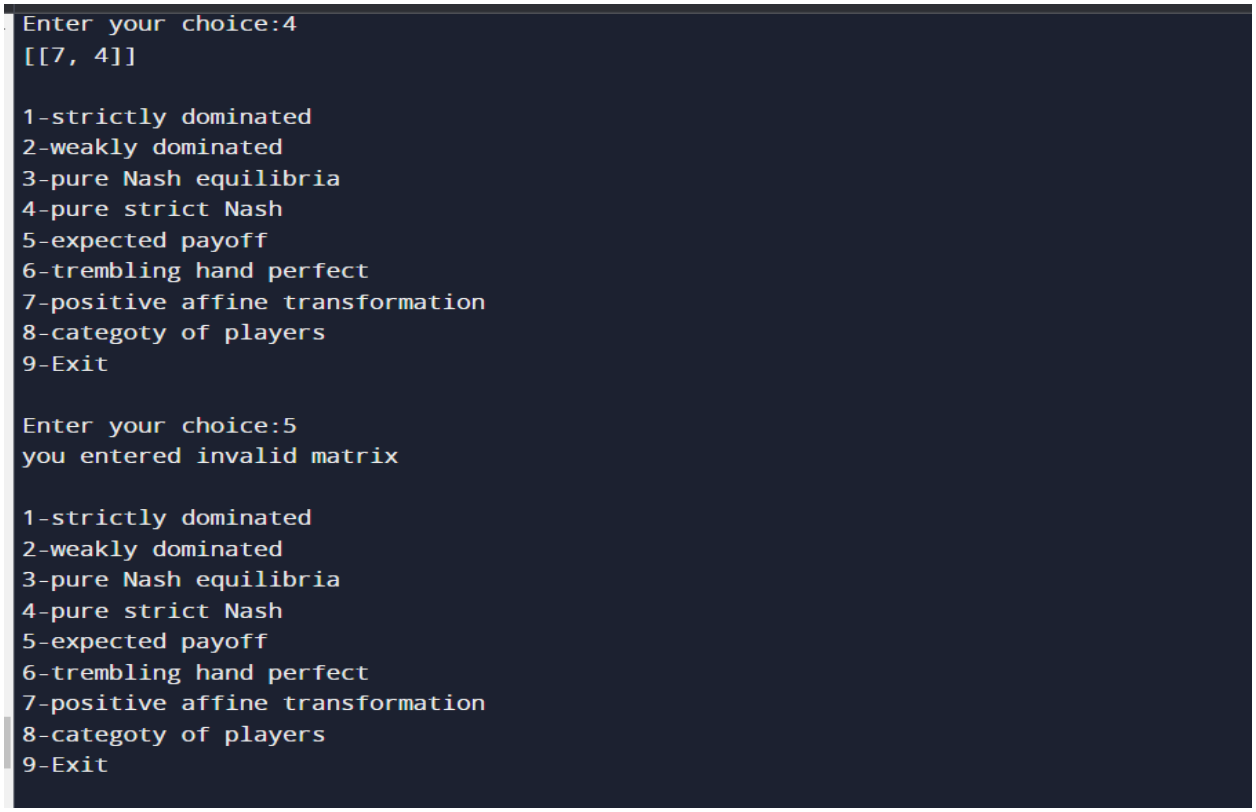
```
Enter the number of rows:2
Enter the number of columns:3
Enter the entries player 1 by row:
7
11
2
6
0
2
Enter the entries player 2 by row:
4
3
0
1
2
0
[[7, 11, 2], [6, 0, 2]]
[[4, 3, 0], [1, 2, 0]]

1-strictly dominated
2-weakly dominated
3-pure Nash equilibria
4-pure strict Nash
5-expected payoff
6-trembling hand perfect
7-positive affine transformation
8-categoty of players
```



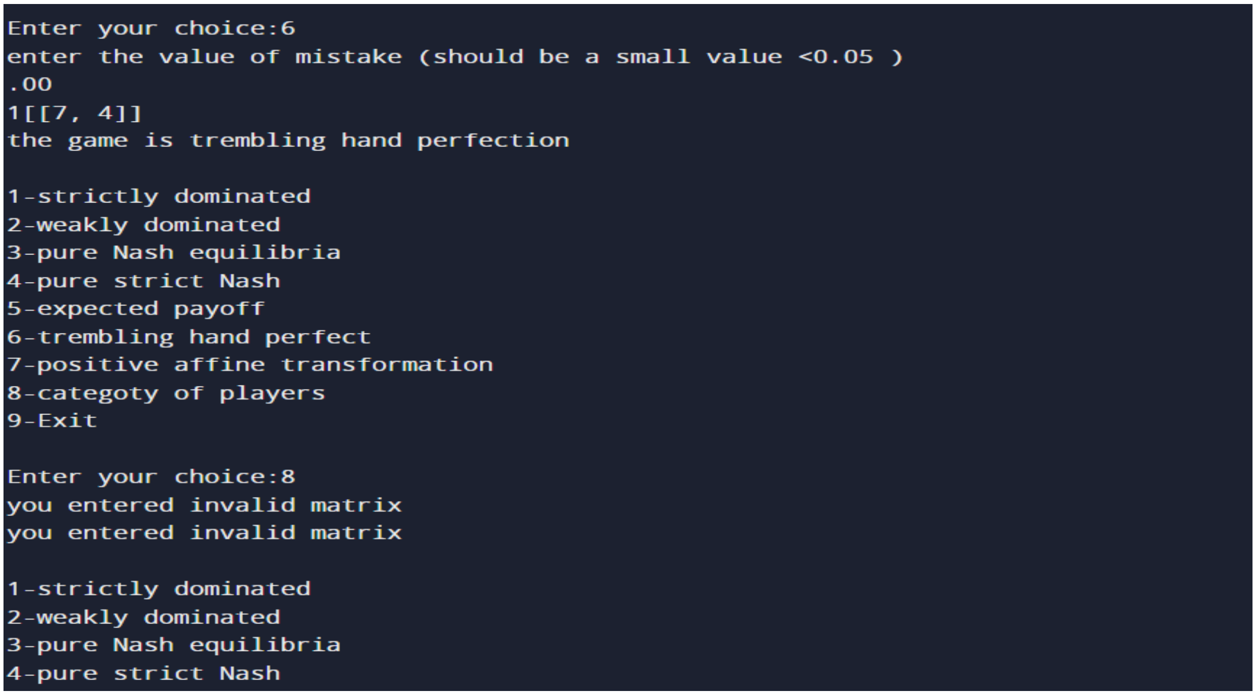
```
Enter your choice:1
([7], [4])

1-strictly dominated
2-weakly dominated
3-pure Nash equilibria
4-pure strict Nash
5-expected payoff
6-trembling hand perfect
7-positive affine transformation
8-categoty of players
9-Exit

Enter your choice:2
([7], [4])

1-strictly dominated
2-weakly dominated
3-pure Nash equilibria
4-pure strict Nash
5-expected payoff
6-trembling hand perfect
7-positive affine transformation
8-categoty of players
9-Exit
```



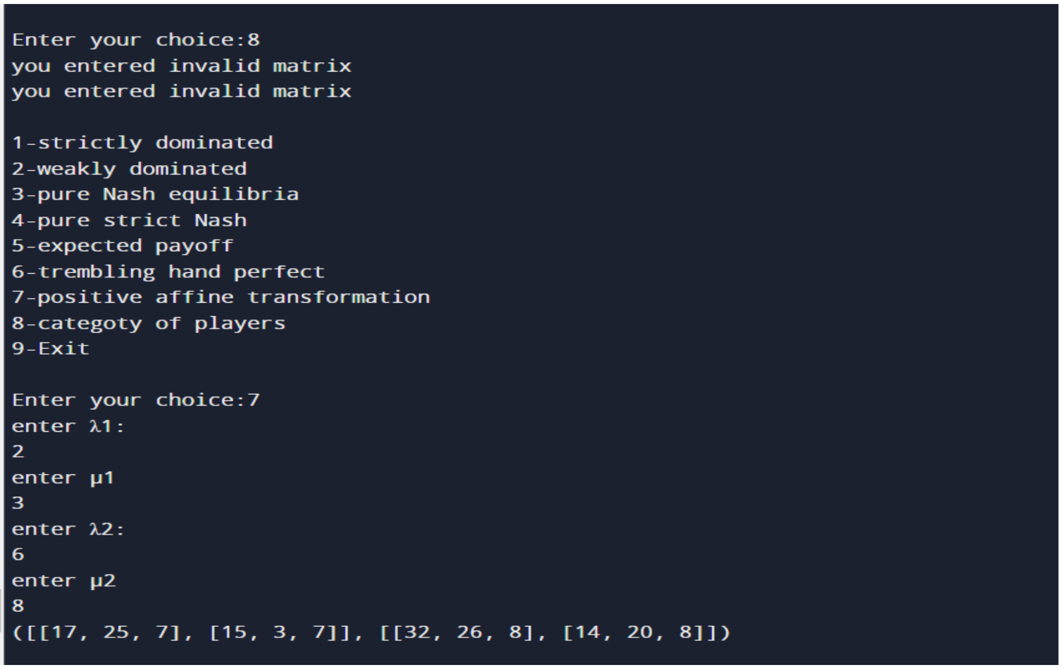
```
Enter your choice:3
[[7, 4]]

1-strictly dominated
2-weakly dominated
3-pure Nash equilibria
4-pure strict Nash
5-expected payoff
6-trembling hand perfect
7-positive affine transformation
8-categoty of players
9-Exit

Enter your choice:4
[[7, 4]]

1-strictly dominated
2-weakly dominated
3-pure Nash equilibria
4-pure strict Nash
5-expected payoff
6-trembling hand perfect
7-positive affine transformation
8-categoty of players
9-Exit
```



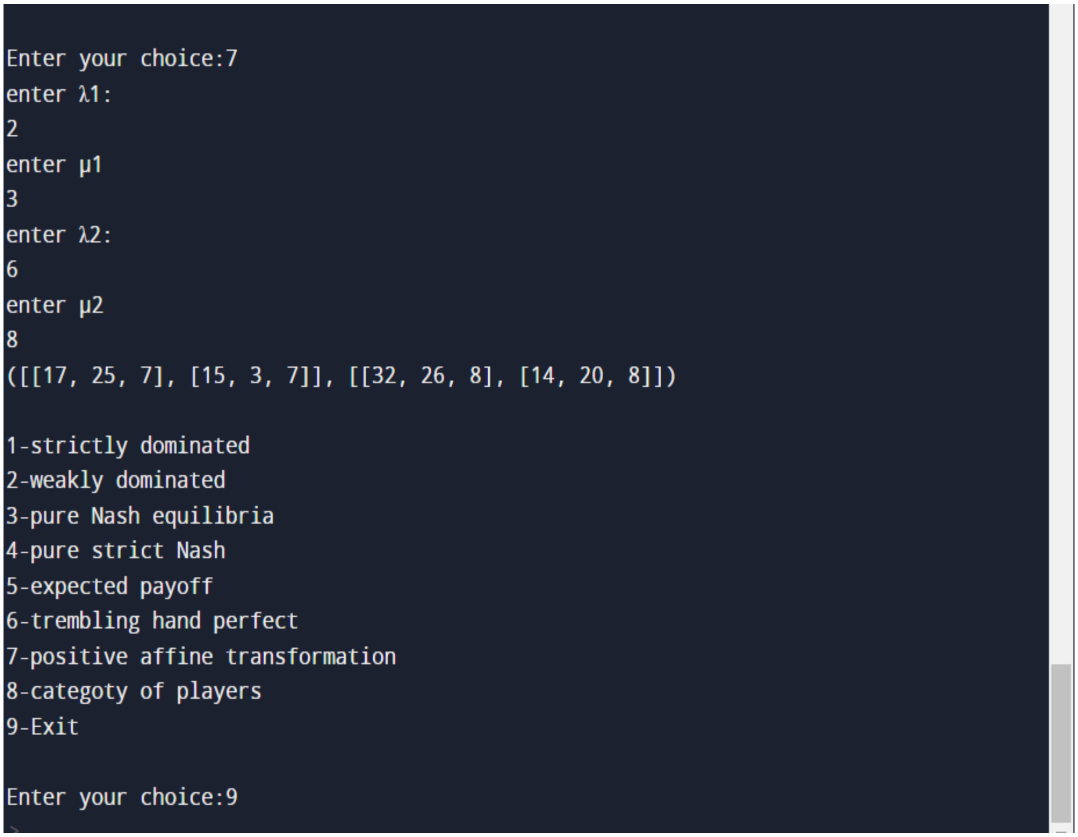
```
Enter your choice:4
[[7, 4]]

1-strictly dominated
2-weakly dominated
3-pure Nash equilibria
4-pure strict Nash
5-expected payoff
6-trembling hand perfect
7-positive affine transformation
8-categoty of players
9-Exit

Enter your choice:5
you entered invalid matrix

1-strictly dominated
2-weakly dominated
3-pure Nash equilibria
4-pure strict Nash
5-expected payoff
6-trembling hand perfect
7-positive affine transformation
8-categoty of players
9-Exit
```



```
Enter your choice:6
enter the value of mistake (should be a small value <0.05 )
.00
1[[7, 4]]
the game is trembling hand perfection

1-strictly dominated
2-weakly dominated
3-pure Nash equilibria
4-pure strict Nash
5-expected payoff
6-trembling hand perfect
7-positive affine transformation
8-categoty of players
9-Exit

Enter your choice:8
you entered invalid matrix
you entered invalid matrix

1-strictly dominated
2-weakly dominated
3-pure Nash equilibria
4-pure strict Nash
```



```
Enter your choice:8
you entered invalid matrix
you entered invalid matrix

1-strictly dominated
2-weakly dominated
3-pure Nash equilibria
4-pure strict Nash
5-expected payoff
6-trembling hand perfect
7-positive affine transformation
8-categoty of players
9-Exit

Enter your choice:7
enter λ1:
2
enter μ1
3
enter λ2:
6
enter μ2
8
([[17, 25, 7], [15, 3, 7]], [[32, 26, 8], [14, 20, 8]])
```



```
Enter your choice:7
enter λ1:
2
enter μ1
3
enter λ2:
6
enter μ2
8
([[17, 25, 7], [15, 3, 7]], [[32, 26, 8], [14, 20, 8]])

1-strictly dominated
2-weakly dominated
3-pure Nash equilibria
4-pure strict Nash
5-expected payoff
6-trembling hand perfect
7-positive affine transformation
8-categoty of players
9-Exit

Enter your choice:9
```

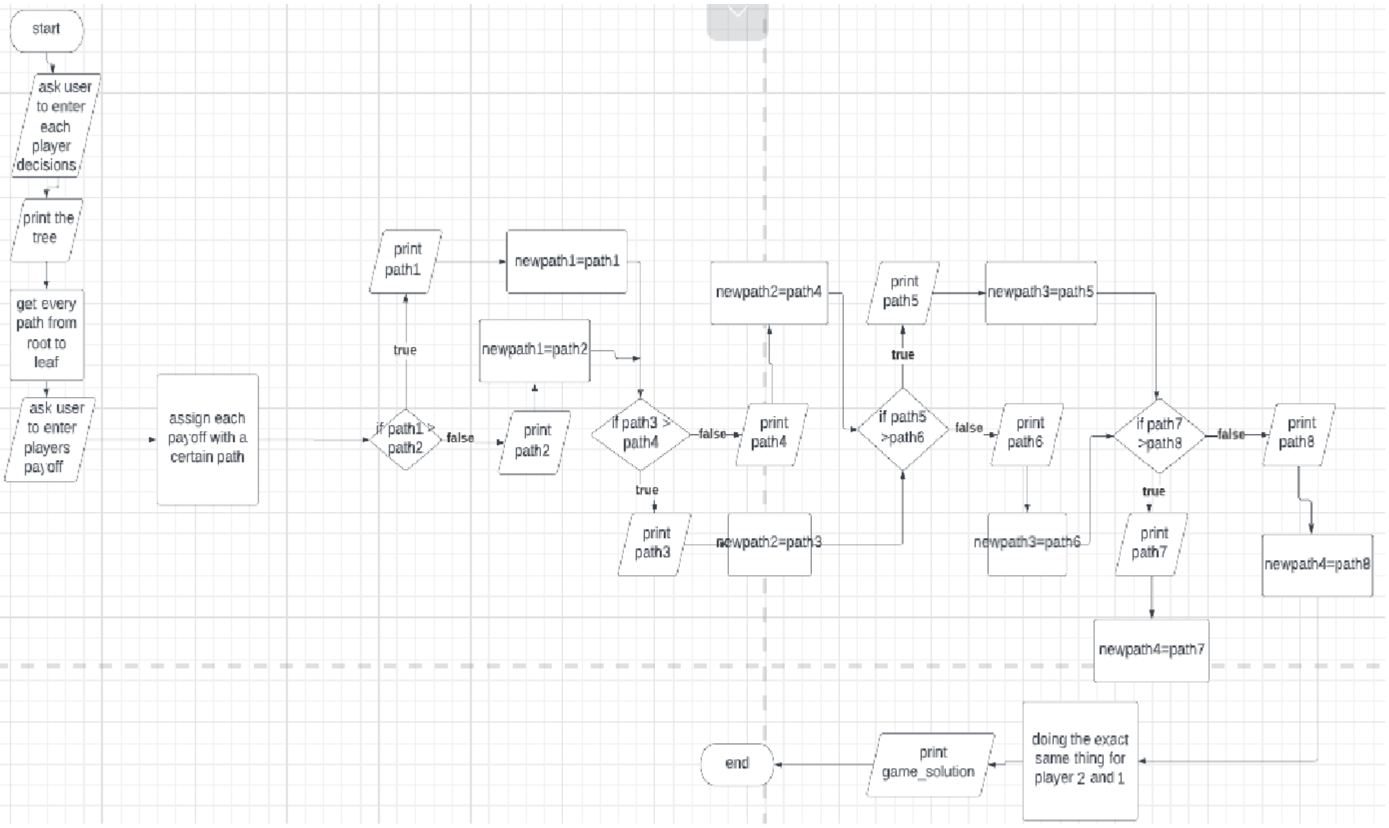
PART (3)

In this part we will solve a 3-player sequential game where each player has two actions at each decision node.

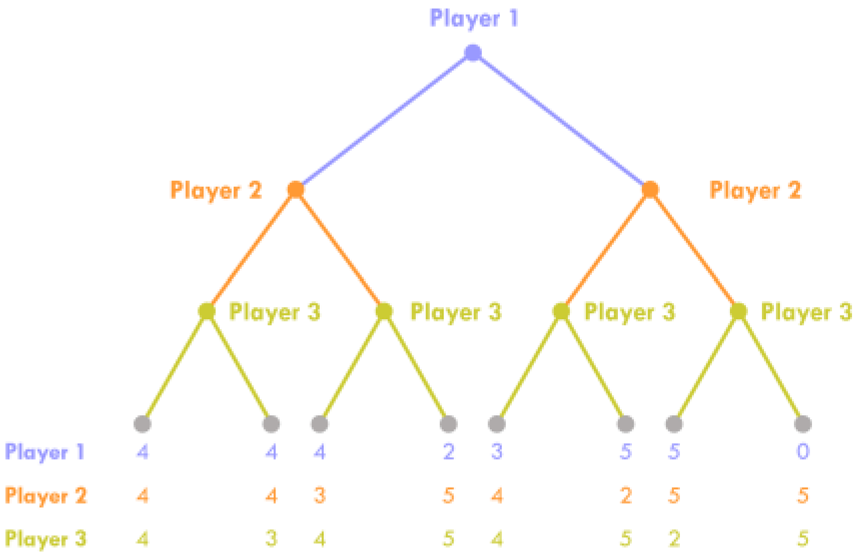
In the following steps our program will be explained:

- First, we will start by implementing a tree using treelib library.
- Ask the user to input the actions for each player.
- In get_path function we use a dictionary to link each action with its id in the tree then we get all the paths from root to leaf.
- In players_payoff function we ask the user to input each player payoff for a specific path then we append these payoffs to its path.
- In backward _induction function we start to compare player 3 payoffs to get only four paths, then we compare player 2 payoffs to get only 2 paths, then we compare player 1 payoffs to get only one path and that is our game solution.

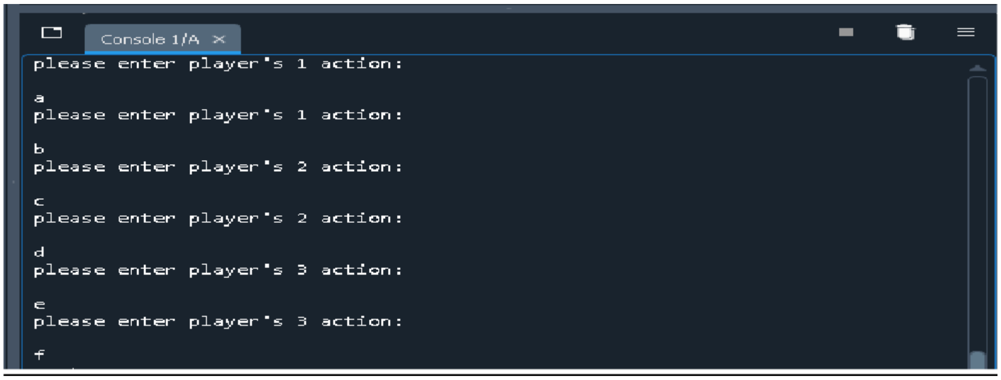
Here's a flowchart explaining the steps:



Example1:

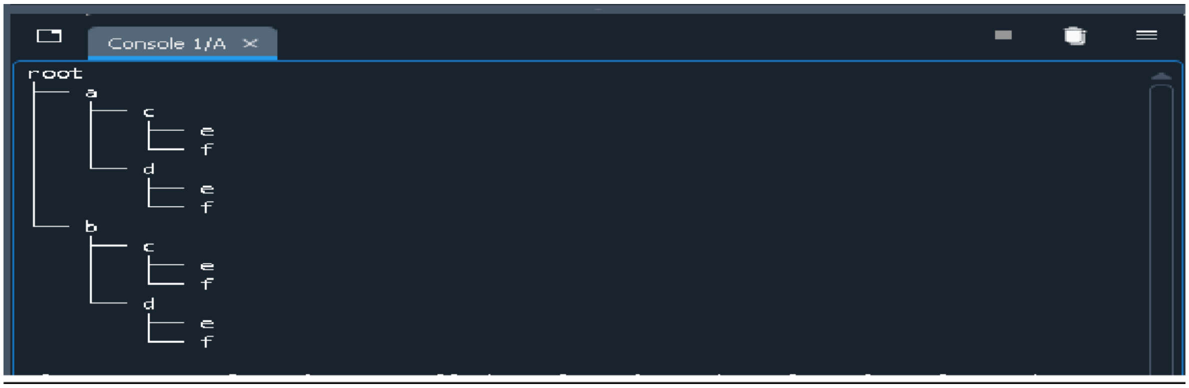


Here's the input of each player's decision:



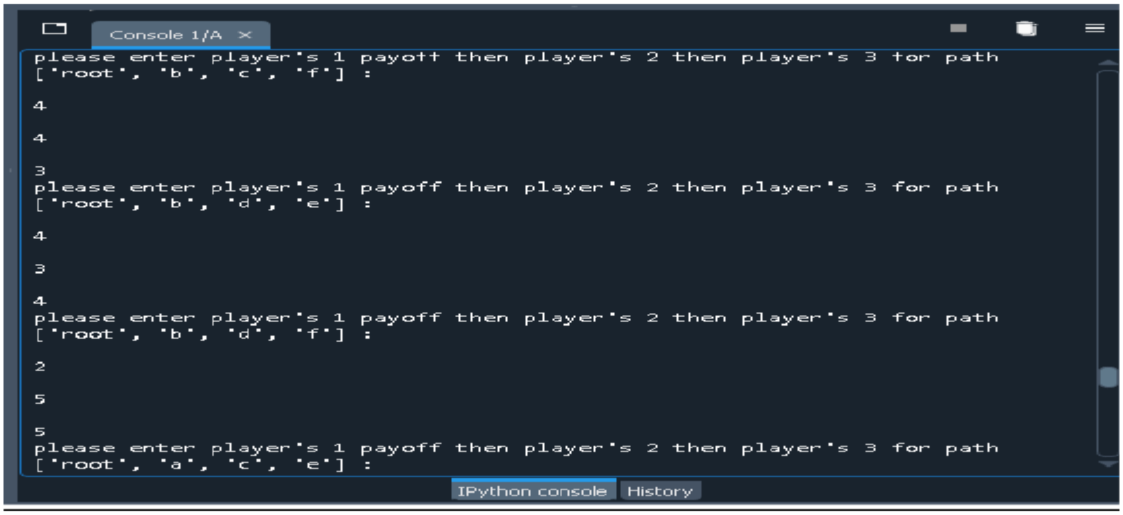
```
please enter player's 1 action:

a
please enter player's 1 action:

b
please enter player's 2 action:

c
please enter player's 2 action:

d
please enter player's 3 action:

e
please enter player's 3 action:

f
```

Here's the tree:

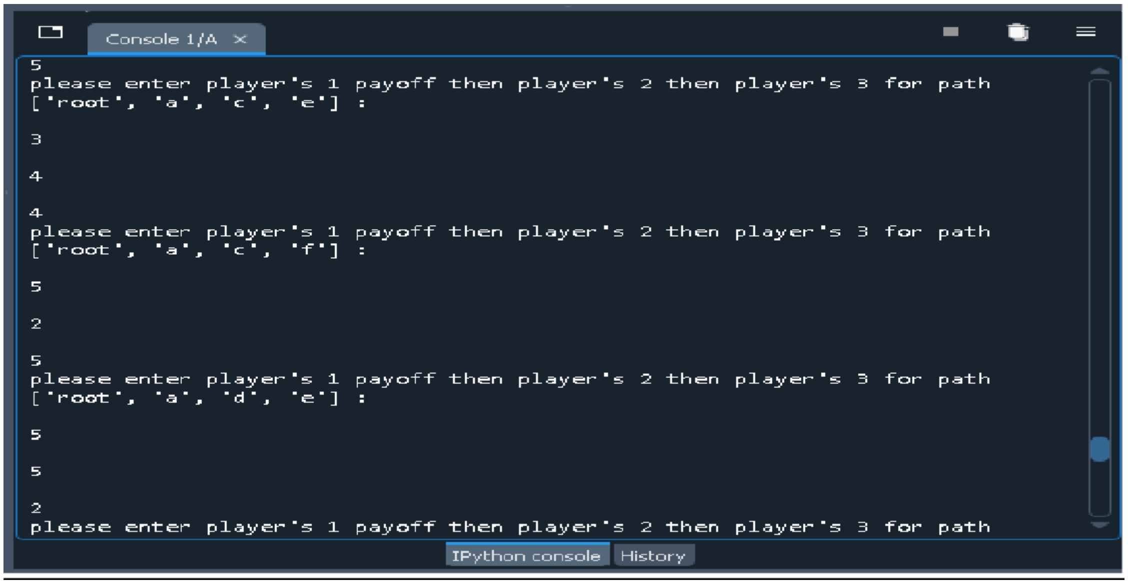


```
root
├── a
│   ├── c
│   │   ├── e
│   │   └── f
│   └── d
│       ├── e
│       └── f
└── b
    ├── c
    │   ├── e
    │   └── f
    └── d
        ├── e
        └── f
```

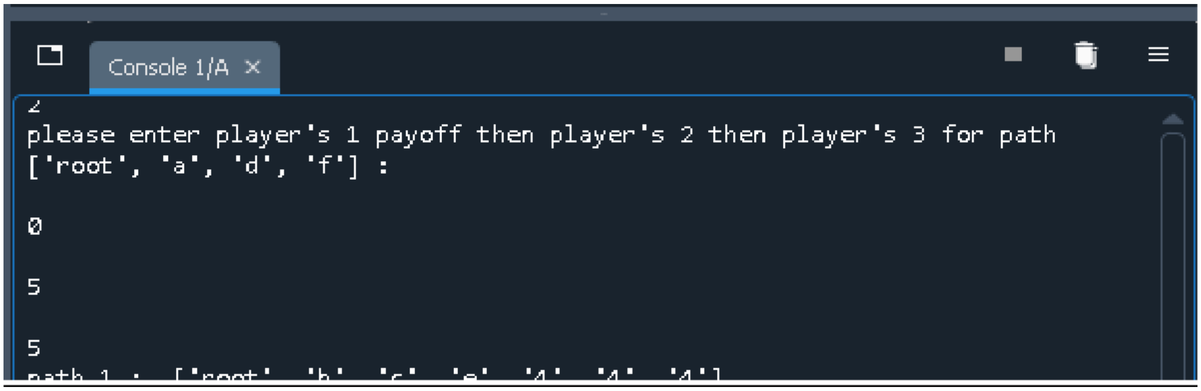
User entering each player payoff:



```
please enter player's 1 payoff then player's 2 then player's 3 for path
['root', 'b', 'c', 'f'] :

4

4

3
please enter player's 1 payoff then player's 2 then player's 3 for path
['root', 'b', 'd', 'e'] :

4

3

4
please enter player's 1 payoff then player's 2 then player's 3 for path
['root', 'b', 'd', 'f'] :

2

5

5
please enter player's 1 payoff then player's 2 then player's 3 for path
['root', 'a', 'c', 'e'] :
```

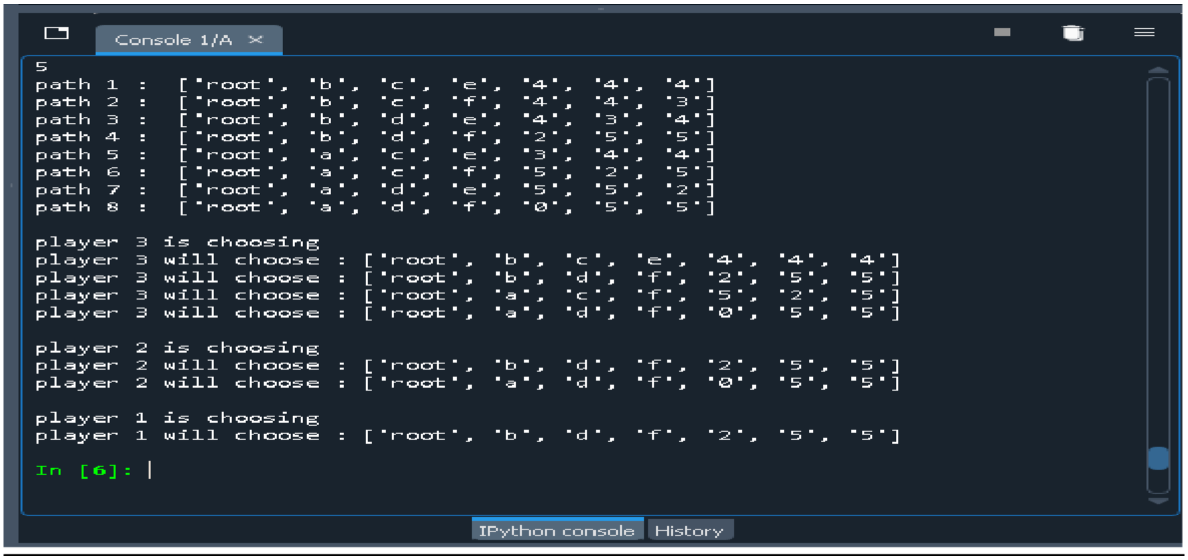


```
5
please enter player's 1 payoff then player's 2 then player's 3 for path
['root', 'a', 'c', 'e'] :

3

4

4
please enter player's 1 payoff then player's 2 then player's 3 for path
['root', 'a', 'c', 'f'] :

5

2

5
please enter player's 1 payoff then player's 2 then player's 3 for path
['root', 'a', 'd', 'e'] :

5

5

2
please enter player's 1 payoff then player's 2 then player's 3 for path
```



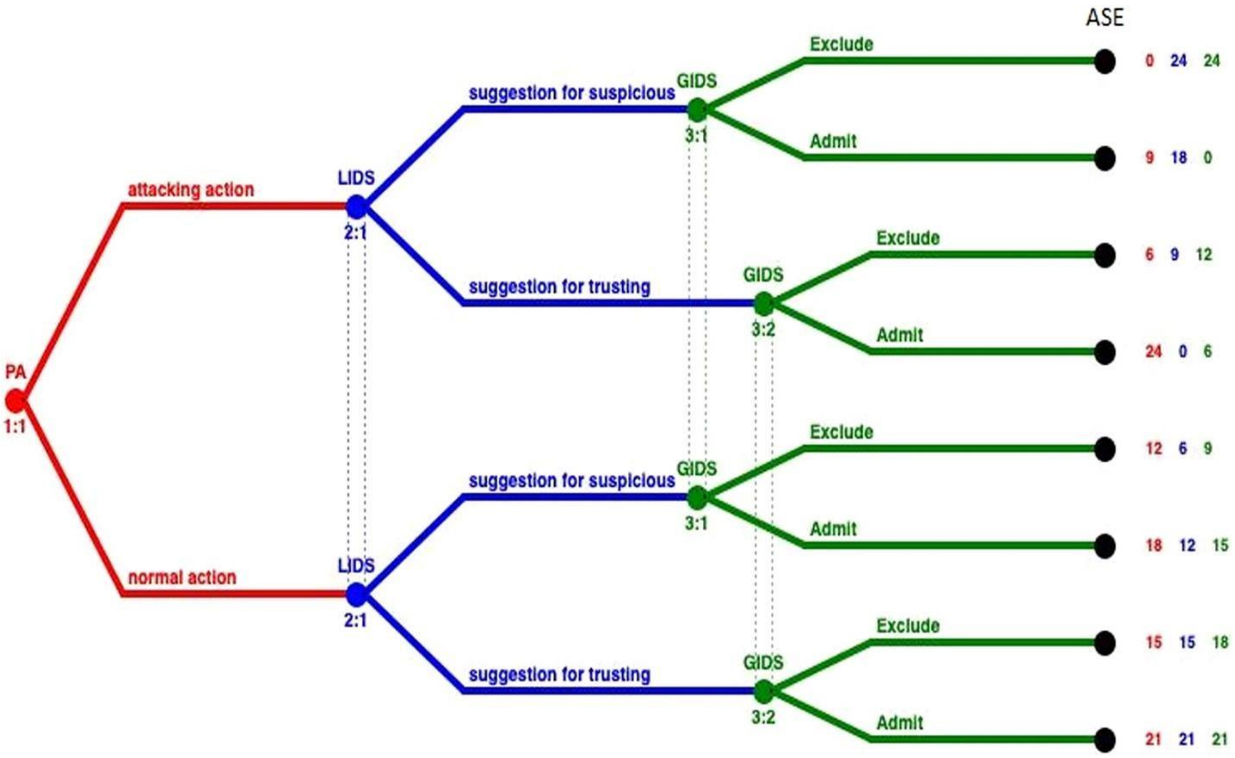
```
2
please enter player's 1 payoff then player's 2 then player's 3 for path
['root', 'a', 'd', 'f'] :

0

5

5
path 1 :  ['root', 'b', 'c', 'e', '4', '4', '4']
```

Printing the game solution using backward induction:

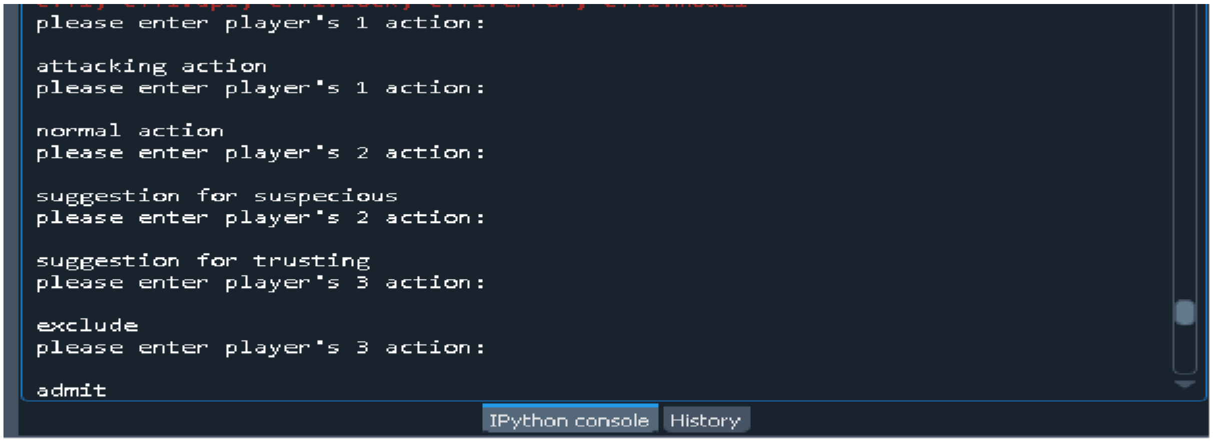


```
5
path 1 :  ['root', 'b', 'c', 'e', '4', '4', '4']
path 2 :  ['root', 'b', 'c', 'f', '4', '4', '3']
path 3 :  ['root', 'b', 'd', 'e', '4', '3', '4']
path 4 :  ['root', 'b', 'd', 'f', '2', '5', '5']
path 5 :  ['root', 'a', 'c', 'e', '3', '4', '4']
path 6 :  ['root', 'a', 'c', 'f', '5', '2', '5']
path 7 :  ['root', 'a', 'd', 'e', '5', '5', '2']
path 8 :  ['root', 'a', 'd', 'f', '0', '5', '5']

player 3 is choosing
player 3 will choose : ['root', 'b', 'c', 'e', '4', '4', '4']
player 3 will choose : ['root', 'b', 'd', 'f', '2', '5', '5']
player 3 will choose : ['root', 'a', 'c', 'f', '5', '2', '5']
player 3 will choose : ['root', 'a', 'd', 'f', '0', '5', '5']

player 2 is choosing
player 2 will choose : ['root', 'b', 'd', 'f', '2', '5', '5']
player 2 will choose : ['root', 'a', 'd', 'f', '0', '5', '5']

player 1 is choosing
player 1 will choose : ['root', 'b', 'd', 'f', '2', '5', '5']

In [6]: |
```

Example 2:

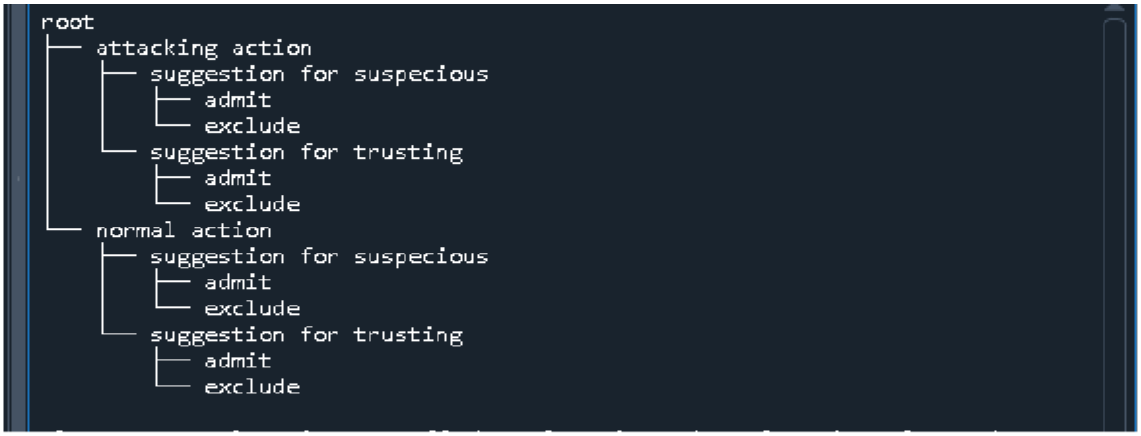


<u>Here's the input of each player's decision:</u>



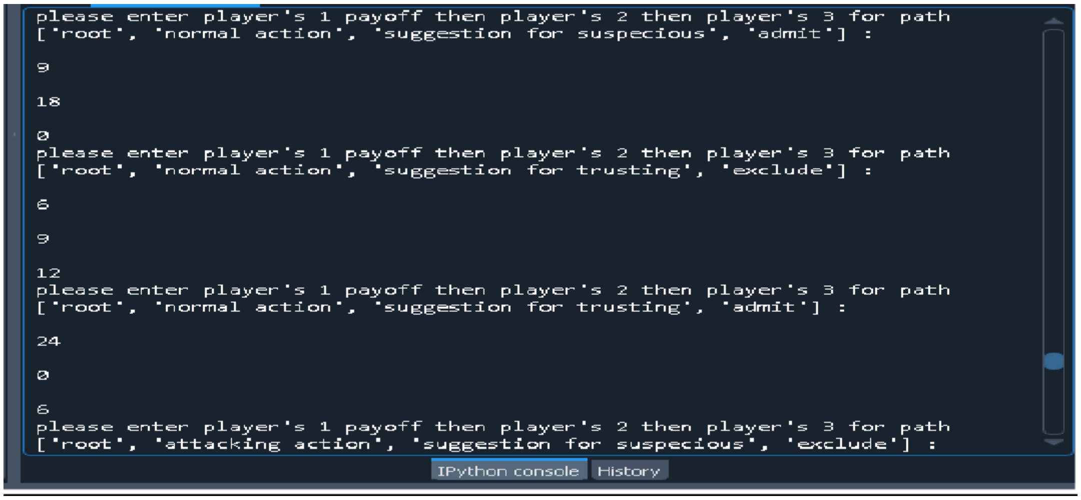
```
please enter player's 1 action:

attacking action
please enter player's 1 action:

normal action
please enter player's 2 action:

suggestion for suspecious
please enter player's 2 action:

suggestion for trusting
please enter player's 3 action:

exclude
please enter player's 3 action:

admit
```

<u>Here's the tree:</u>

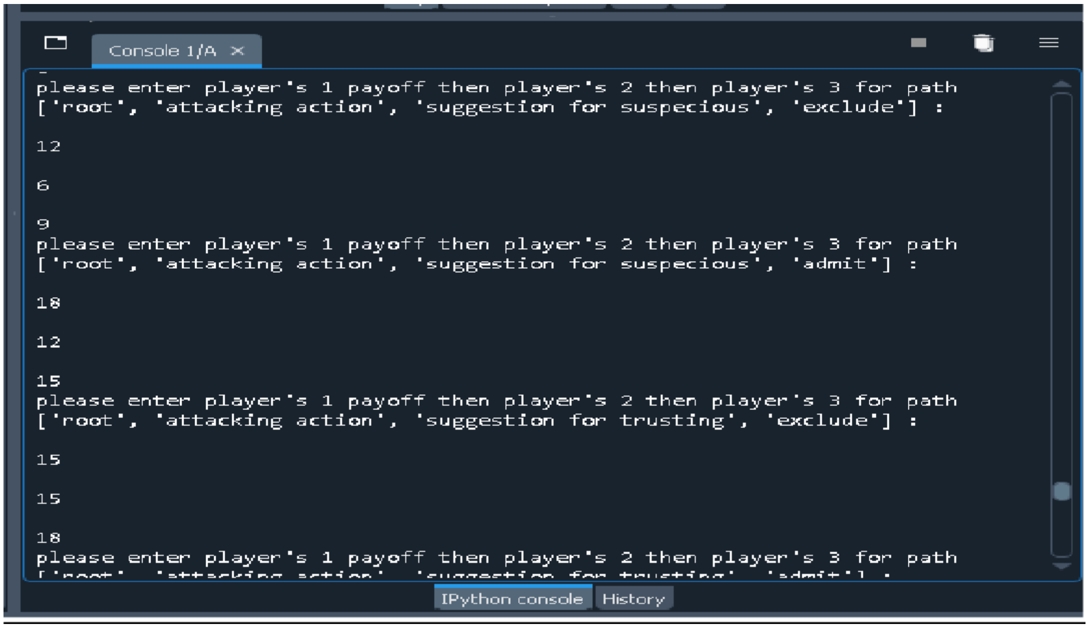


```
root
├── attacking action
│   ├── suggestion for suspecious
│   │   ├── admit
│   │   └── exclude
│   └── suggestion for trusting
│       ├── admit
│       └── exclude
└── normal action
    ├── suggestion for suspecious
    │   ├── admit
    │   └── exclude
    └── suggestion for trusting
        ├── admit
        └── exclude
```

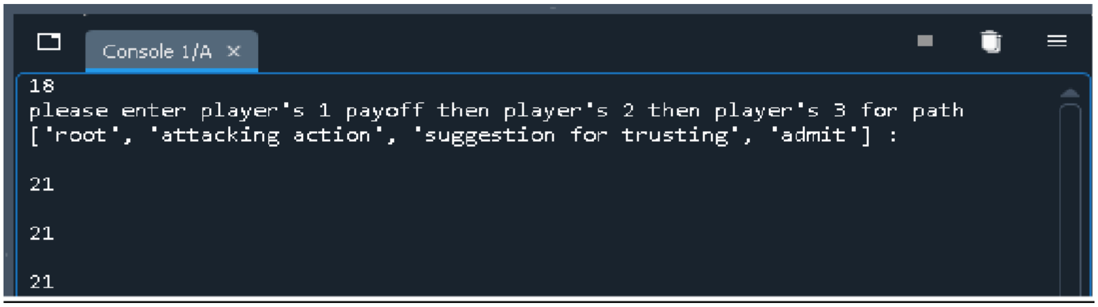
User entering each player payoff:



```
please enter player's 1 payoff then player's 2 then player's 3 for path
['root', 'normal action', 'suggestion for suspecious', 'admit'] :

9

18

0
please enter player's 1 payoff then player's 2 then player's 3 for path
['root', 'normal action', 'suggestion for trusting', 'exclude'] :

6

9

12
please enter player's 1 payoff then player's 2 then player's 3 for path
['root', 'normal action', 'suggestion for trusting', 'admit'] :

24

0

6
please enter player's 1 payoff then player's 2 then player's 3 for path
['root', 'attacking action', 'suggestion for suspecious', 'exclude'] :
```

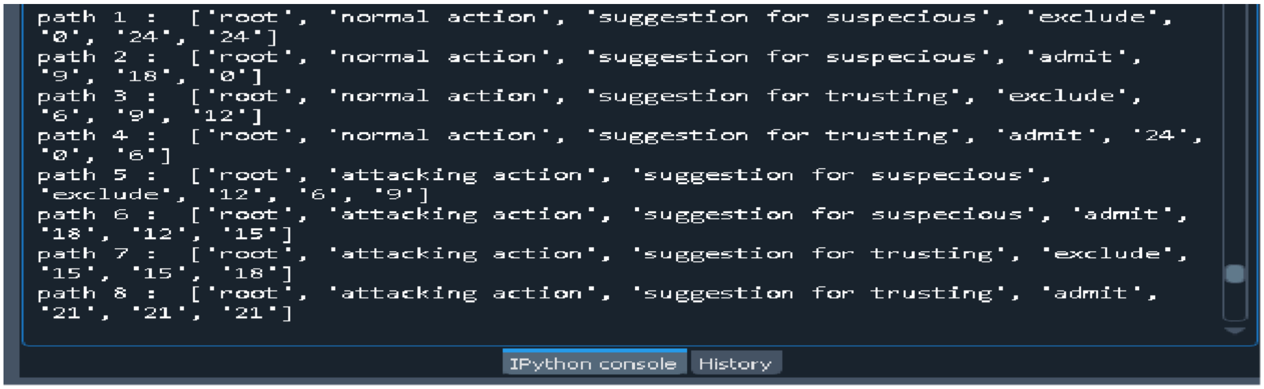


```
please enter player's 1 payoff then player's 2 then player's 3 for path
['root', 'attacking action', 'suggestion for suspecious', 'exclude'] :

12

6

9
please enter player's 1 payoff then player's 2 then player's 3 for path
['root', 'attacking action', 'suggestion for suspecious', 'admit'] :

18

12

15
please enter player's 1 payoff then player's 2 then player's 3 for path
['root', 'attacking action', 'suggestion for trusting', 'exclude'] :

15

15

18
please enter player's 1 payoff then player's 2 then player's 3 for path
['root', 'attacking action', 'suggestion for trusting', 'admit'] :
```

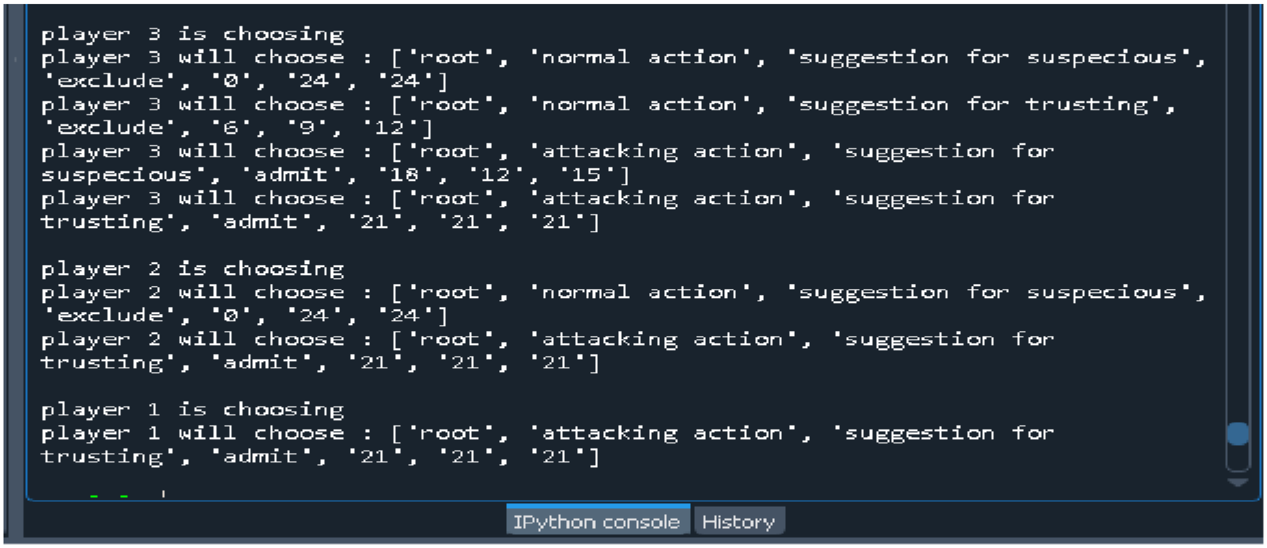


```
18
please enter player's 1 payoff then player's 2 then player's 3 for path
['root', 'attacking action', 'suggestion for trusting', 'admit'] :

21

21

21
```

Printing the game solution using backward induction:

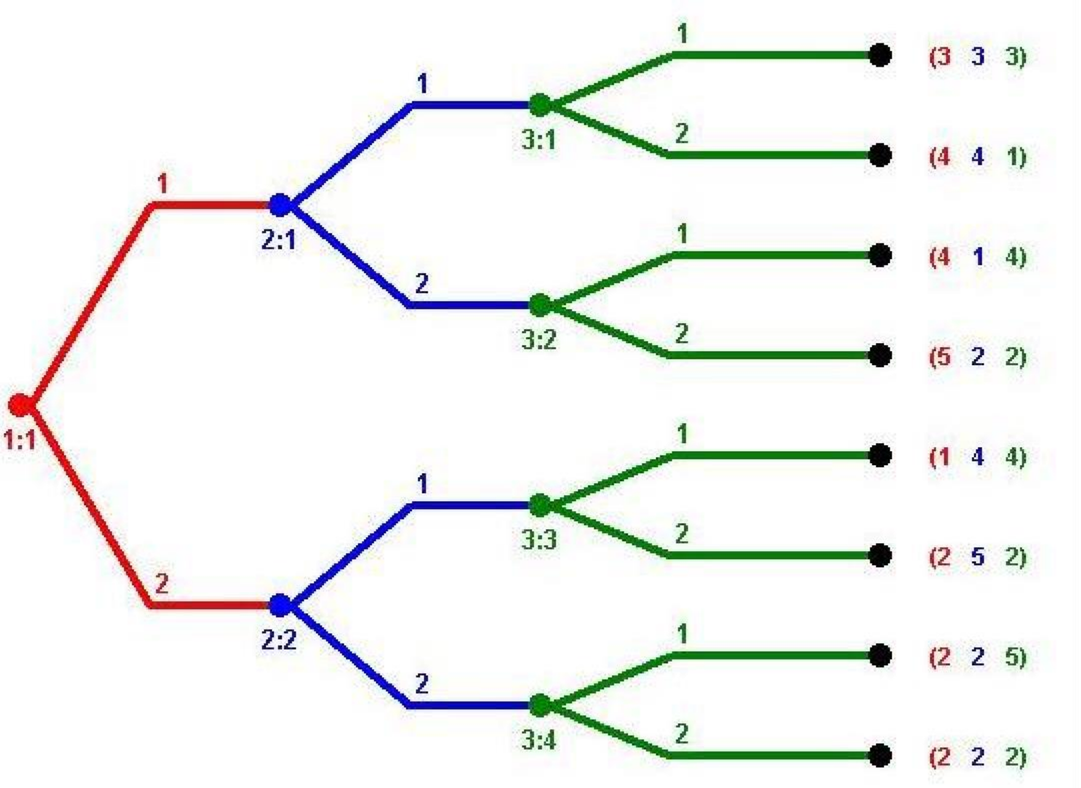


```
path 1 :  ['root', 'normal action', 'suggestion for suspecious', 'exclude',
'0', '24', '24']
path 2 :  ['root', 'normal action', 'suggestion for suspecious', 'admit',
'9', '18', '0']
path 3 :  ['root', 'normal action', 'suggestion for trusting', 'exclude',
'6', '9', '12']
path 4 :  ['root', 'normal action', 'suggestion for trusting', 'admit', '24',
'0', '6']
path 5 :  ['root', 'attacking action', 'suggestion for suspecious',
'exclude', '12', '6', '9']
path 6 :  ['root', 'attacking action', 'suggestion for suspecious', 'admit',
'18', '12', '15']
path 7 :  ['root', 'attacking action', 'suggestion for trusting', 'exclude',
'15', '15', '18']
path 8 :  ['root', 'attacking action', 'suggestion for trusting', 'admit',
'21', '21', '21']
```

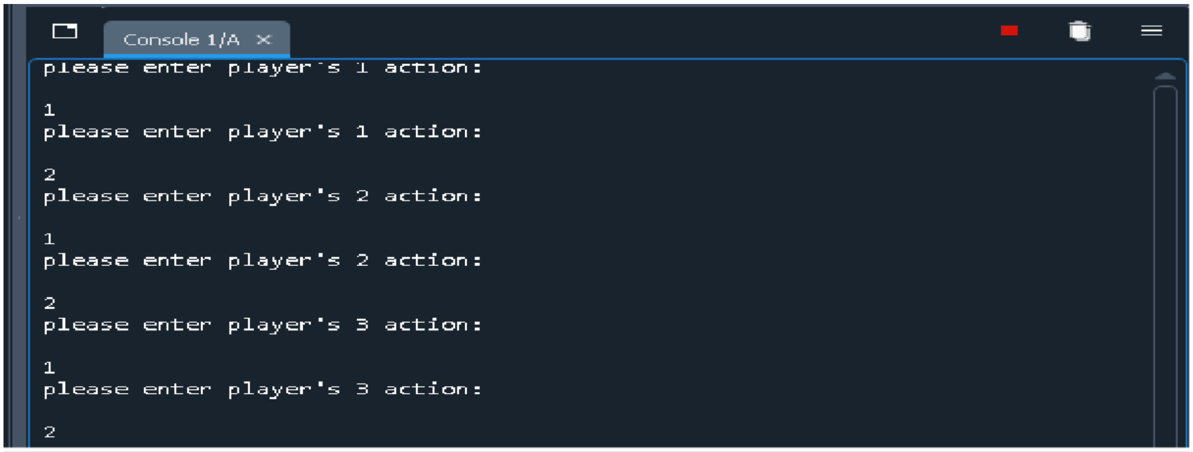


```
player 3 is choosing
player 3 will choose : ['root', 'normal action', 'suggestion for suspecious',
'exclude', '0', '24', '24']
player 3 will choose : ['root', 'normal action', 'suggestion for trusting',
'exclude', '6', '9', '12']
player 3 will choose : ['root', 'attacking action', 'suggestion for
suspecious', 'admit', '18', '12', '15']
player 3 will choose : ['root', 'attacking action', 'suggestion for
trusting', 'admit', '21', '21', '21']

player 2 is choosing
player 2 will choose : ['root', 'normal action', 'suggestion for suspecious',
'exclude', '0', '24', '24']
player 2 will choose : ['root', 'attacking action', 'suggestion for
trusting', 'admit', '21', '21', '21']

player 1 is choosing
player 1 will choose : ['root', 'attacking action', 'suggestion for
trusting', 'admit', '21', '21', '21']
```

Example 3:

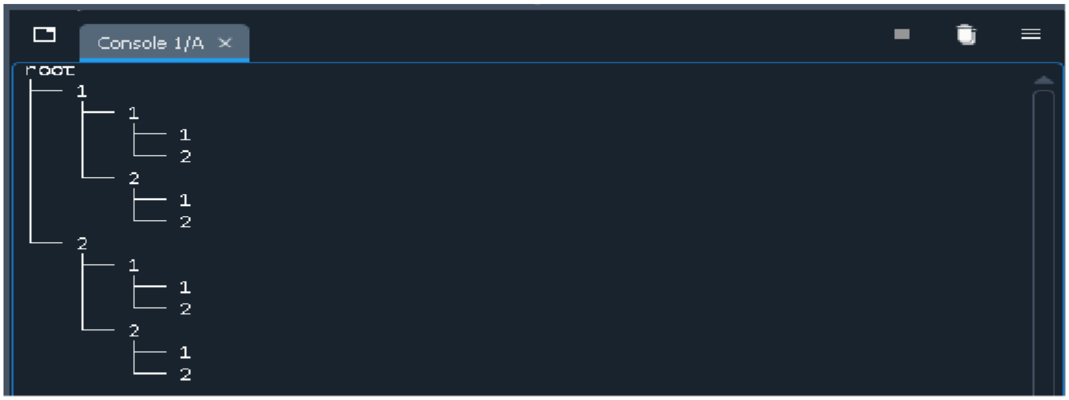


Here’s the input of each player’s decision:



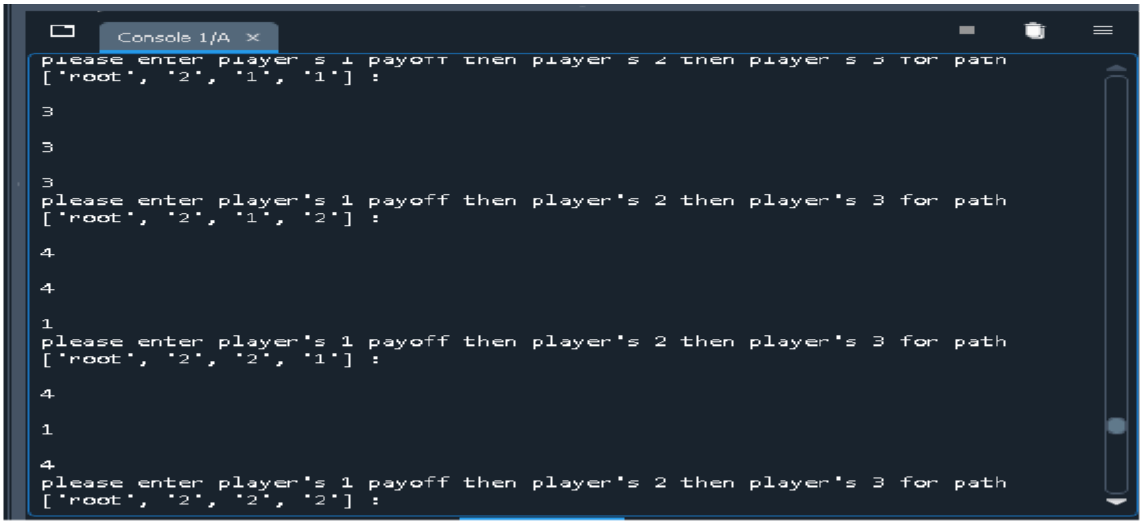
```
please enter player's 1 action:

1
please enter player's 1 action:

2
please enter player's 2 action:

1
please enter player's 2 action:

2
please enter player's 3 action:

1
please enter player's 3 action:

2
```

<u>Here’s the tree:</u>

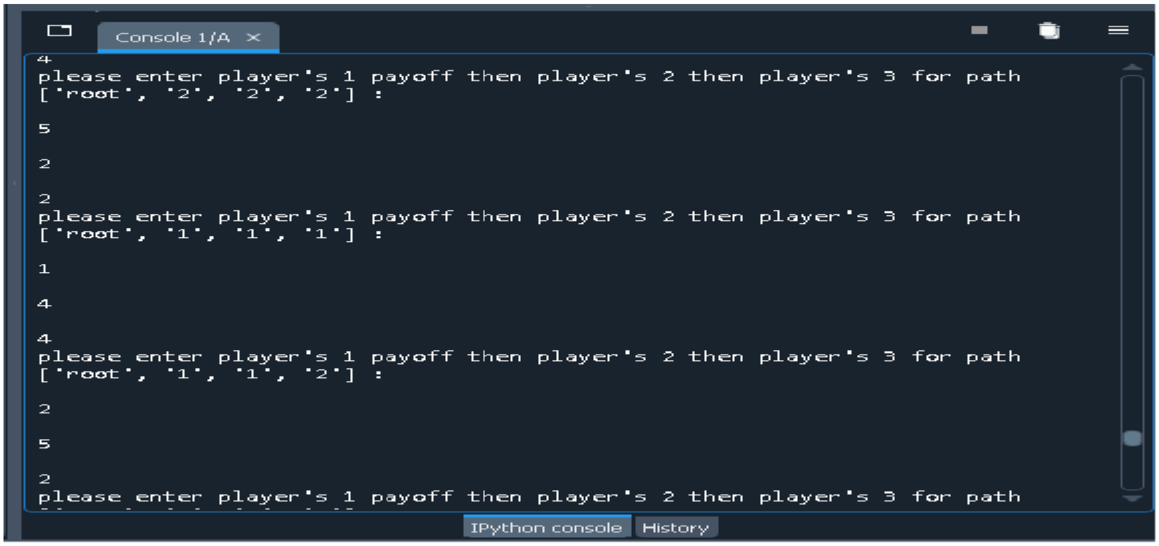


```
root
├── 1
│   ├── 1
│   │   ├── 1
│   │   └── 2
│   └── 2
│       ├── 1
│       └── 2
└── 2
    ├── 1
    │   ├── 1
    │   └── 2
    └── 2
        ├── 1
        └── 2
```

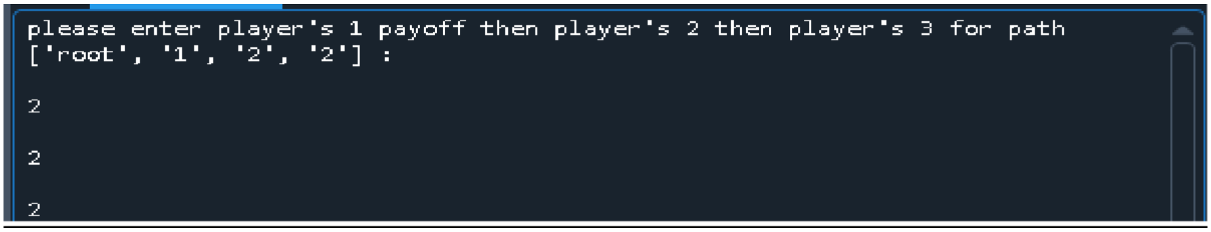
<u>User entering each player payoff:</u>



```
please enter player's 1 payoff then player's 2 then player's 3 for path
['root', '2', '1', '1'] :

3

3

3
please enter player's 1 payoff then player's 2 then player's 3 for path
['root', '2', '1', '2'] :

4

4

1
please enter player's 1 payoff then player's 2 then player's 3 for path
['root', '2', '2', '1'] :

4

1

4
please enter player's 1 payoff then player's 2 then player's 3 for path
['root', '2', '2', '2'] :
```

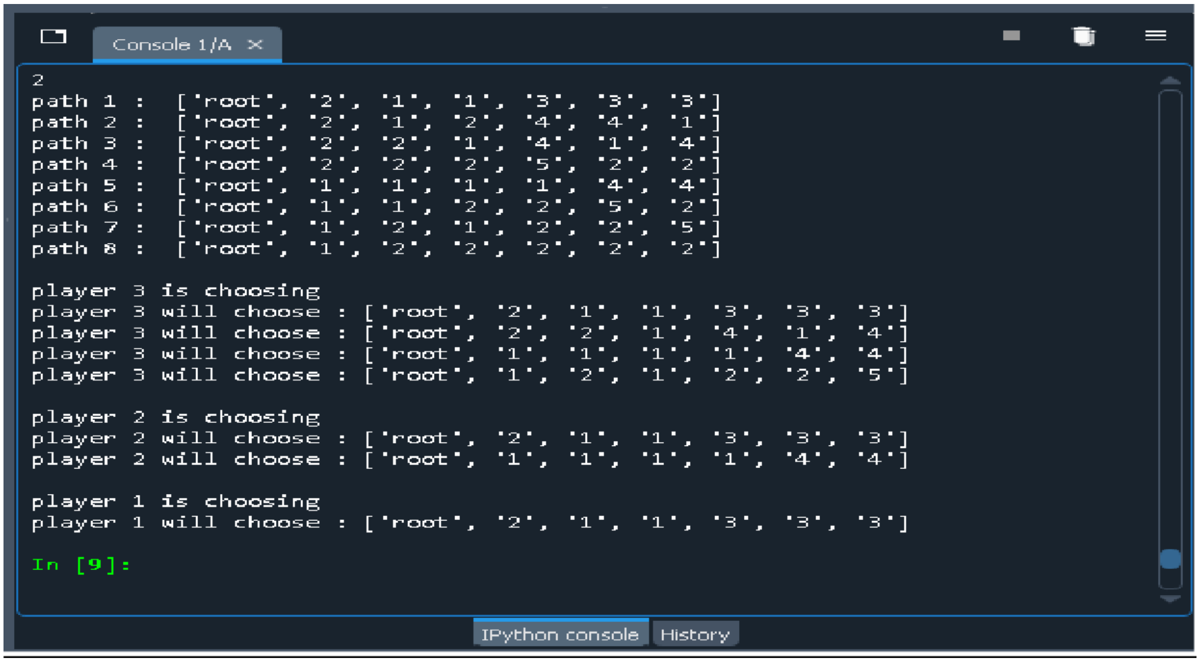


```
4
please enter player's 1 payoff then player's 2 then player's 3 for path
['root', '2', '2', '2'] :

5

2

2
please enter player's 1 payoff then player's 2 then player's 3 for path
['root', '1', '1', '1'] :

1

4

4
please enter player's 1 payoff then player's 2 then player's 3 for path
['root', '1', '1', '2'] :

2

5

2
please enter player's 1 payoff then player's 2 then player's 3 for path
```



```
please enter player's 1 payoff then player's 2 then player's 3 for path
['root', '1', '2', '2'] :

2

2

2
```

Printing the game solution using backward induction:



```
2
path 1 :  ['root', '2', '1', '1', '3', '3', '3']
path 2 :  ['root', '2', '1', '2', '4', '4', '1']
path 3 :  ['root', '2', '2', '1', '4', '1', '4']
path 4 :  ['root', '2', '2', '2', '5', '2', '2']
path 5 :  ['root', '1', '1', '1', '1', '4', '4']
path 6 :  ['root', '1', '1', '2', '2', '5', '2']
path 7 :  ['root', '1', '2', '1', '2', '2', '5']
path 8 :  ['root', '1', '2', '2', '2', '2', '2']

player 3 is choosing
player 3 will choose : ['root', '2', '1', '1', '3', '3', '3']
player 3 will choose : ['root', '2', '2', '1', '4', '1', '4']
player 3 will choose : ['root', '1', '1', '1', '1', '4', '4']
player 3 will choose : ['root', '1', '2', '1', '2', '2', '5']

player 2 is choosing
player 2 will choose : ['root', '2', '1', '1', '3', '3', '3']
player 2 will choose : ['root', '1', '1', '1', '1', '4', '4']

player 1 is choosing
player 1 will choose : ['root', '2', '1', '1', '3', '3', '3']

In [9]:
```

PART (4)

A

Definition:

Sprouts were created in 1967 by two mathematicians at the University of Cambridge in the United Kingdom, John H. Conway, and Michael S. Paterson. One of Martin Gardner's "Mathematical Games" sections in Scientific American popularized the game.

Conway says, "Here's a quotation from me:’’

The day after sprouts sprouted, it appeared like everyone was playing it; there were small groups of people peeping over absurd to amazing sprout placements at coffee or tea times."

Players: Two

The players take turns in joining dots according to simple rules, until one player cannot make a move.

Description:

Start by drawing two or more spots on a piece of paper.

Players then take turns to make a move,

Rules:

- Draw a line joining two spots or a single spot to itself.
- The line must not cross another line or pass through another spot.
- Draw a spot on the new line.
- No more than three lines can emerge from any spot.
- The last player to be able to move wins.

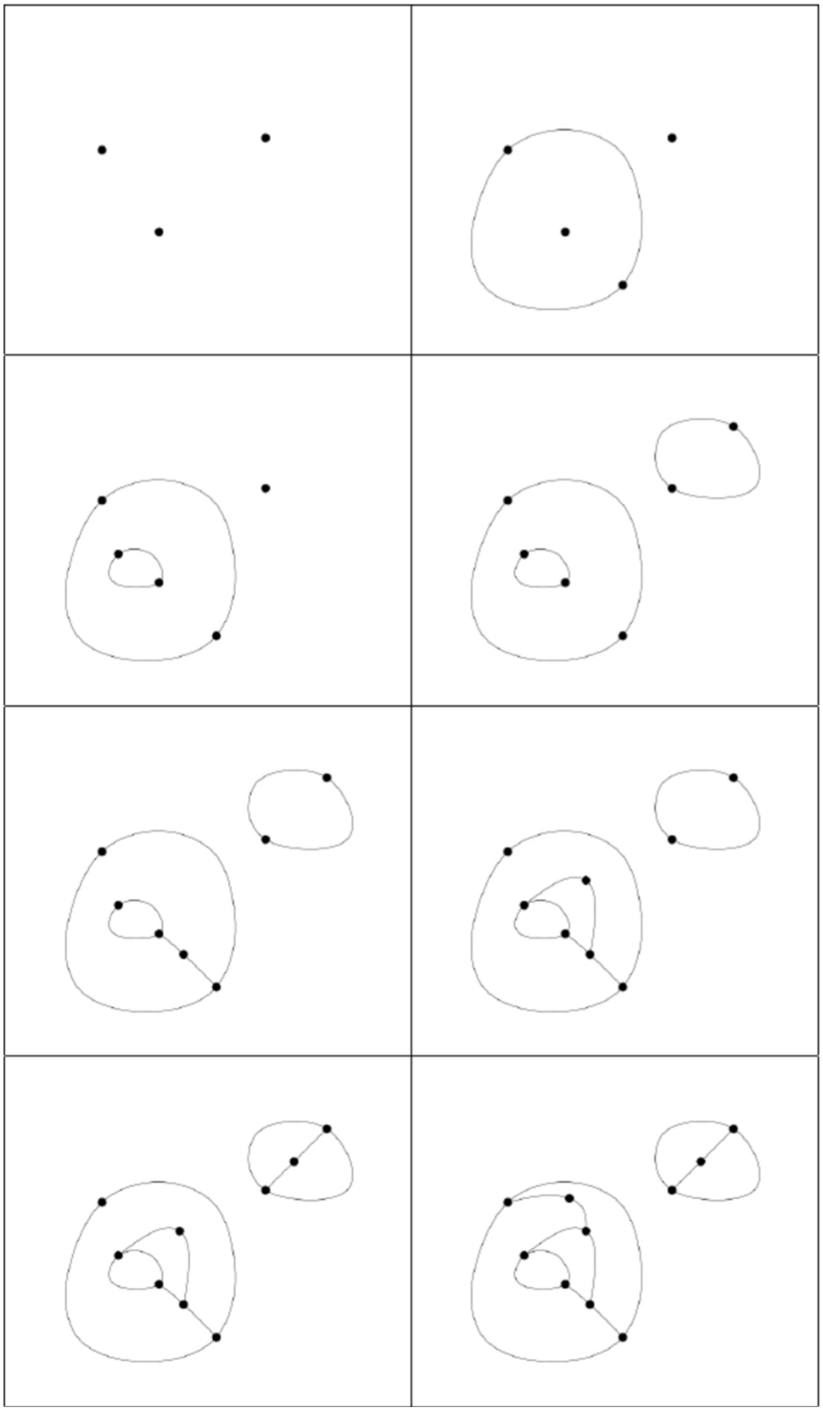
The maximum length of a game:

a Sprouts game with initial points cannot have more than $3n-1$ moves, n number of nodes

Minimum length of a game:

a Sprouts game will have at least $2n$ moves, n number of nodes

Example: Game of Sprouts starting with 3 points with the First player as winner



B) Zeimelo theorem can’t apply to the game

Because Zeimelo Theorem: Given:

○ Two players.

○ Perfect Information.

○ Finite Number of Nodes (strategies and rounds).

○ Three possible outcomes W, L, T

And the game doesn’t have a tie

C) The sprouts game have a winning player

Because the game end when the player can’t play at any nodes, the game rules is one of the player is winning

D) Winning strategy determine which player can force a win if they play perfectly.

References

- *https://www.revita.com.cy/prisoners-dilemma-interpretation-and-real-life-applications/*
- *https://www.investopedia.com/articles/investing/110513/utilizing-prisoners-dilemma-business-and-economy.asp#:~:text=The%20prisoner%E2%80%99s%20dilemma%20basically%20provides%20a%20framework%20for,political%20science%20to%20philosophy%2C%20psychology%2C%20biology%2C%20and%20sociology*
- *https://www.unife.it/economia/lm.economia/insegnamenti/economia-applicata-avanzata/mat-did/trade-scm/game-theory-in-business-applications2003-per-lezione.pdf*
- *https://blogs.cornell.edu/info2040/2017/09/11/the-business-applications-of-the-prisoners-dilemma/*
- *https://stratechery.com/2017/the-uber-dilemma/*
- Book: “Games People Play: Game Theory in Life, Business, and Beyond” for Scott P. Stevens.
- *Book Game Theory and Business Application 2001*
- *https://equalocean.com/analysis/201903301672#:~:text=Uber%20and%20Lyft%20have%20somehow%20been%20trapped%20in,make%20money%20but%20will%20soon%20definitely%20face%20failure*
- *https://proc.conisar.org/2019/pdf/5230.pdf*
- *https://www.strategy-business.com/article/Why-businesses-need-to-help-employees-build-friendships*