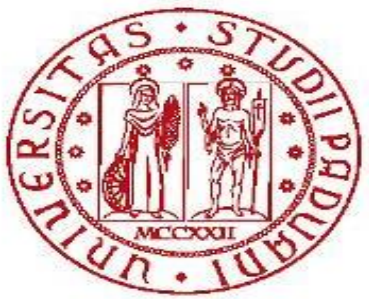


UNIVERSITÀ
DEGLI STUDI
DI PADOVA

ITERATIVE PRISONER'S DILEMMA

Venkata Avinash Jakkampudi (2005626)
Anand Kumar Tumpati (2009408)
Naga Divya Sunkara (1216017)
Poornima Peddu (1227739)

Prisoner’s Dilemma

- The prisoner's dilemma is a paradox in decision analysis in which two individuals acting in their own self-interests do not produce the optimal outcome.
- The typical prisoner's dilemma is set up in such a way that both parties choose to protect themselves at the expense of the other participant.
- As a result, both participants find themselves in a worse state than if they had cooperated with each other in the decision-making process.
- The prisoner's dilemma is one of the most well-known concepts in modern game theory.

Payoff Matrix

	Player 1		
Player 2		**Co-Operate**	**Defect**
	Co-Operate	R,R	S,T
	Defect	T,S	P,P

T, R, P, S are integers

T>R>P>S

2R > T+S

Payoff matrix example with values

- For example: T = 3, R = 2, P = 1, S = 0

	Player 1		
Player 2		**Co-Operate**	**Defect**
	Co-Operate	2,2	0,3
	Defect	3,0	1,1

Example of Prisoner's Dilemma

	Player 1		
Player 2		**Co-Operate**	**Defect**
	Co-Operate	2,2	0,3
	Defect	3,0	1,1

Iterative Prisoner’s Dilemma

- The iterated prisoner's dilemma is an extension of the general form except the game is repeatedly played by the same participants.
- An iterated prisoner's dilemma differs from the original concept of a prisoner's dilemma because participants can learn about the behavioural tendencies of their counterparty.

Strategies

- Nice Guy:

This strategy always co-operates with the opponent irrespective of the opponents moves, it does not betray or defect at least once in the tournament.

Ex: Consider two players P0 and P1

P0 be the player with strategy Nice Guy.

P1 be the player with random moves.

P0 = [C, C, C, C, C, C, C]

P1 = [C, D, D, C, C, D, C]

- Bad Guy:

This strategy always defects/betrays with the opponent irrespective of the opponents moves, it does not co-operate at least once in the tournament.

Ex: Consider Two players P0 and P1

P0 be the player with strategy Bad Guy.
P1 be the player with random moves.
P0 = [D, D, D, D, D, D, D]
P1 = [C, C, D, C, D, D, D]

- Mainly Nice:

This strategy mostly co-operates, the co-operative moves percentage is always greater than the defects/betray moves.

Ex: Consider Two Players P0 and P1

P0 be the player with strategy Mainly Nice.

P1 be the player with random moves.

P0 = [C, C, C, D, C, C, D]

P1 = [D, D, C, D, C, C, C]

- Mainly Bad:

This strategy mostly betrays, the betray moves percentage is always greater than the co-operative moves.

Ex: Consider Two Players P0 And P1

P0 be the player with strategy Mainly Bad.
P1 be the player with random moves.
P0 = [D, D, C, D, D, C, D]
P1 = [D, C, D, C, D, D, C]

- Tit For Tat:

  This strategy always repeats the previous move of the opponent.

  Ex: Consider Two players P0 and P1

  P0 be the player with random moves.
  P1 be the player with strategy Tit for Tat.
  P0 = [C, C, C, D, D, C, D]
  P1 = [C, C, C, D, D, C, D]

- Grudger:

This strategy co-operates till the opponent co-operates and if the opponent defects at least once, from then the grudger always defects irrespective of the next moves from opponent.

Ex: Consider Two players P0 and P1

P0 be the player with random moves.
P1 be the player with strategy Grudger.
P0 = [C, C, D, C, C, C, C]
P1 = [C, C, D, D, D, D, D]

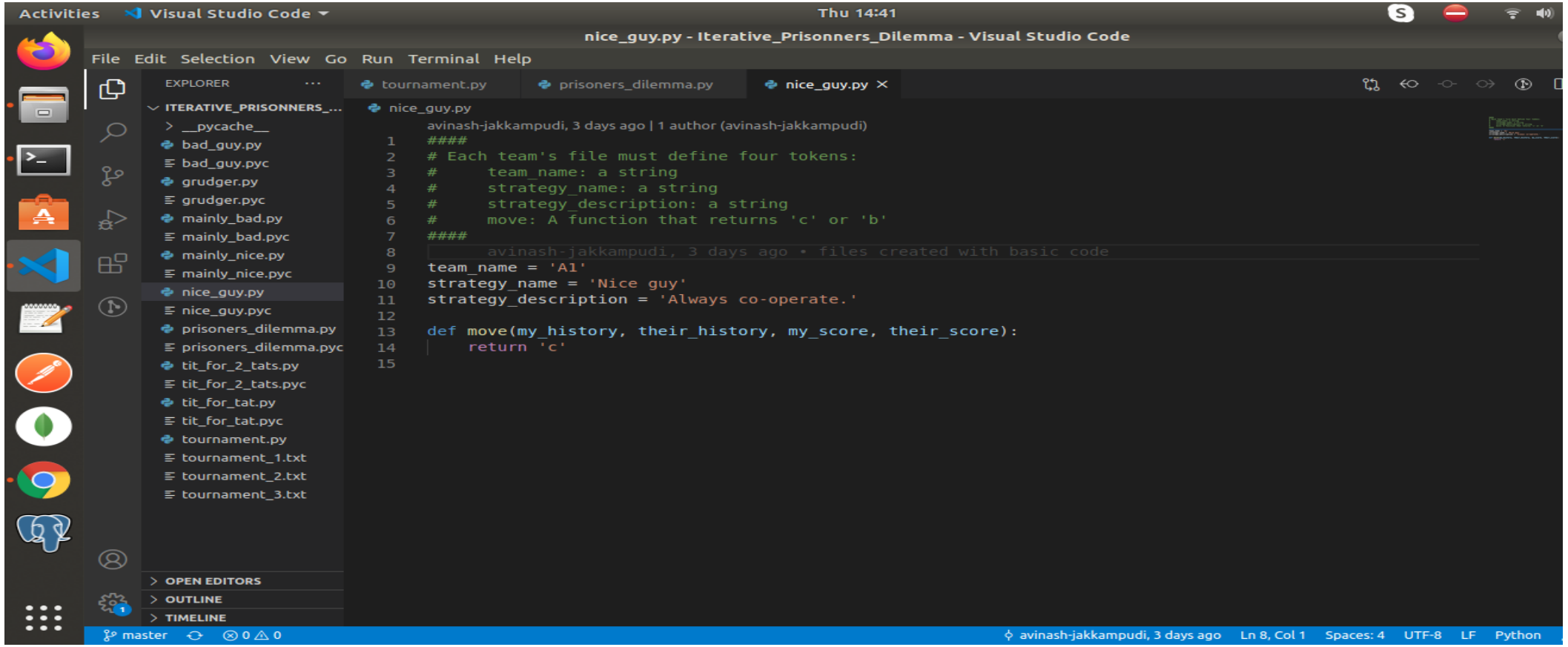
- Tit for two Tats:

  This strategy defects once whenever the opponent defects twice.

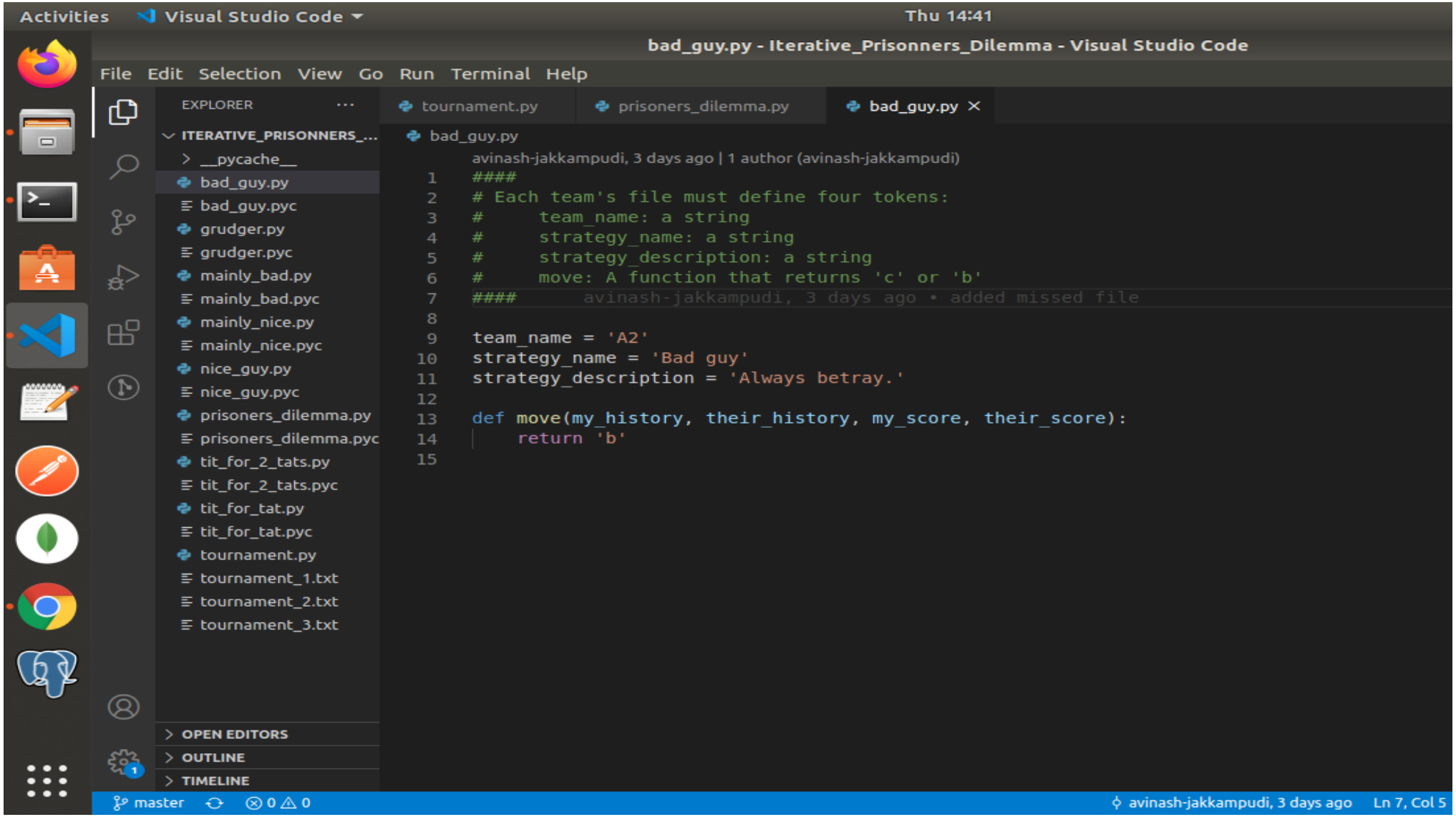
  Ex: Consider Two players P0 and P1

  P0 be the player with random moves.
  P1 be the player with strategy Tit for two Tats.
  P0 = [C, C, D, D, C, C, D]
  P1 = [C, C, C, D, C, C, C]

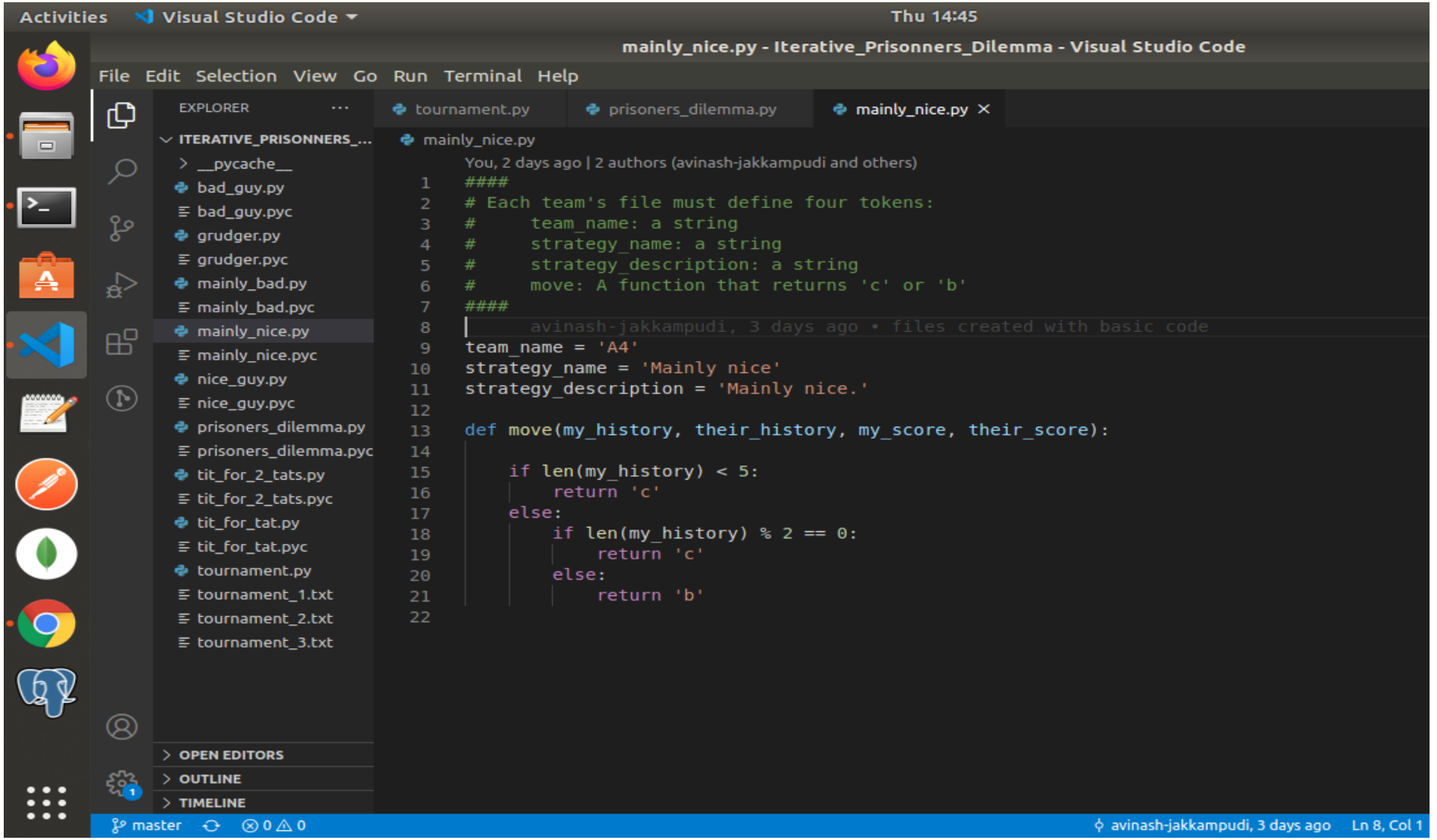
Codes for all Strategies



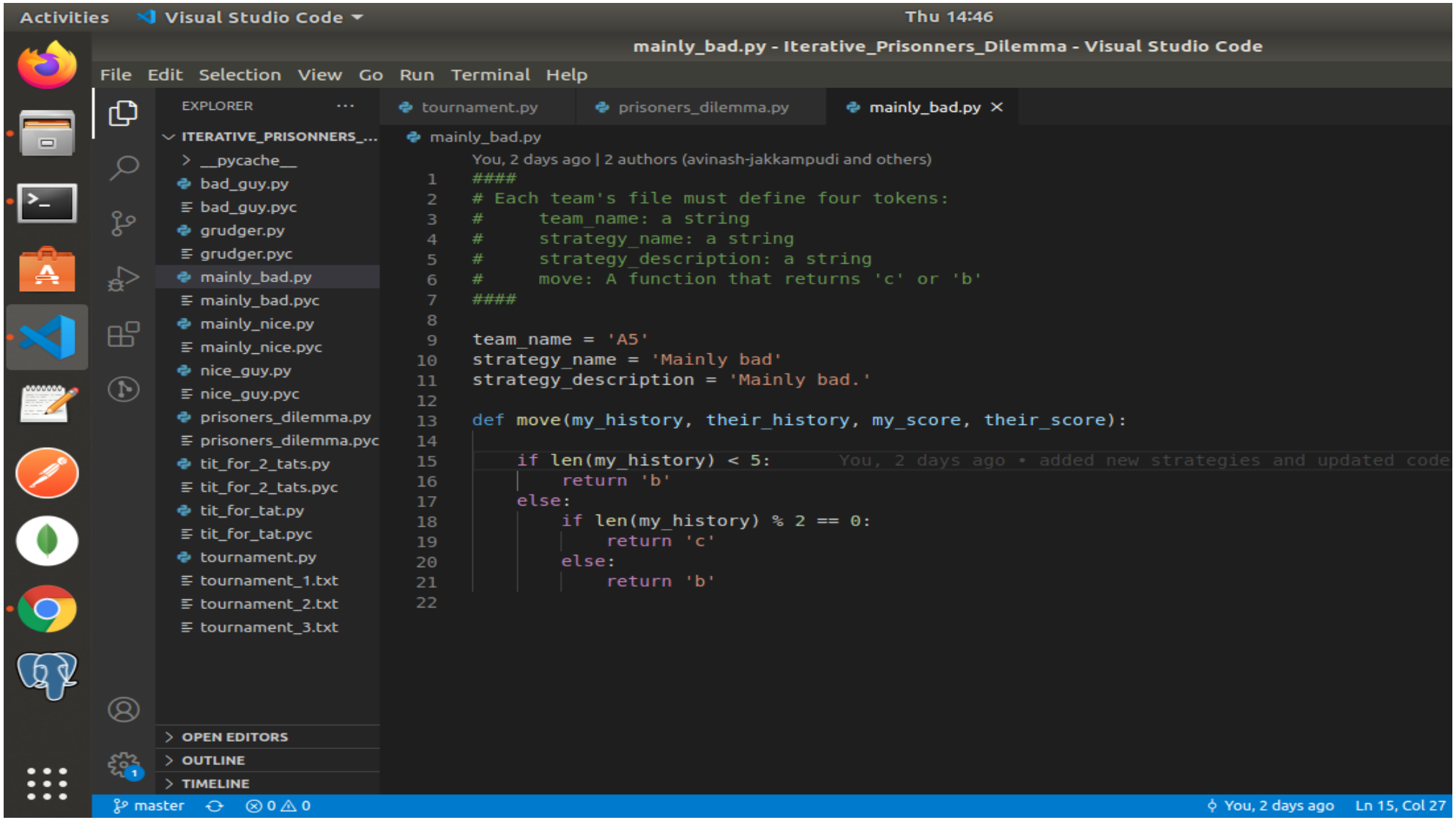
Nice Guy


Activities
Visual Studio Code
Thu 14:41
bad_guy.py - Iterative_Prisonners_Dilemma - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER
ITERATIVE_PRISONNERS_...
__pycache__
bad_guy.py
bad_guy.pyc
grudger.py
grudger.pyc
mainly_bad.py
mainly_bad.pyc
mainly_nice.py
mainly_nice.pyc
nice_guy.py
nice_guy.pyc
prisoners_dilemma.py
prisoners_dilemma.pyc
tit_for_2_tats.py
tit_for_2_tats.pyc
tit_for_tat.py
tit_for_tat.pyc
tournament.py
tournament_1.txt
tournament_2.txt
tournament_3.txt
OPEN EDITORS
OUTLINE
TIMELINE
tournament.py
prisoners_dilemma.py
bad_guy.py
bad_guy.py
avinash-jakkampudi, 3 days ago | 1 author (avinash-jakkampudi)
1 ####
2 # Each team's file must define four tokens:
3 # team_name: a string
4 # strategy_name: a string
5 # strategy_description: a string
6 # move: A function that returns 'c' or 'b'
7 #### avinash-jakkampudi, 3 days ago • added missed file
8
9 team_name = 'A2'
10 strategy_name = 'Bad guy'
11 strategy_description = 'Always betray.'
12
13 def move(my_history, their_history, my_score, their_score):
14 return 'b'
15
master
0 0
avinash-jakkampudi, 3 days ago
Ln 7, Col 5

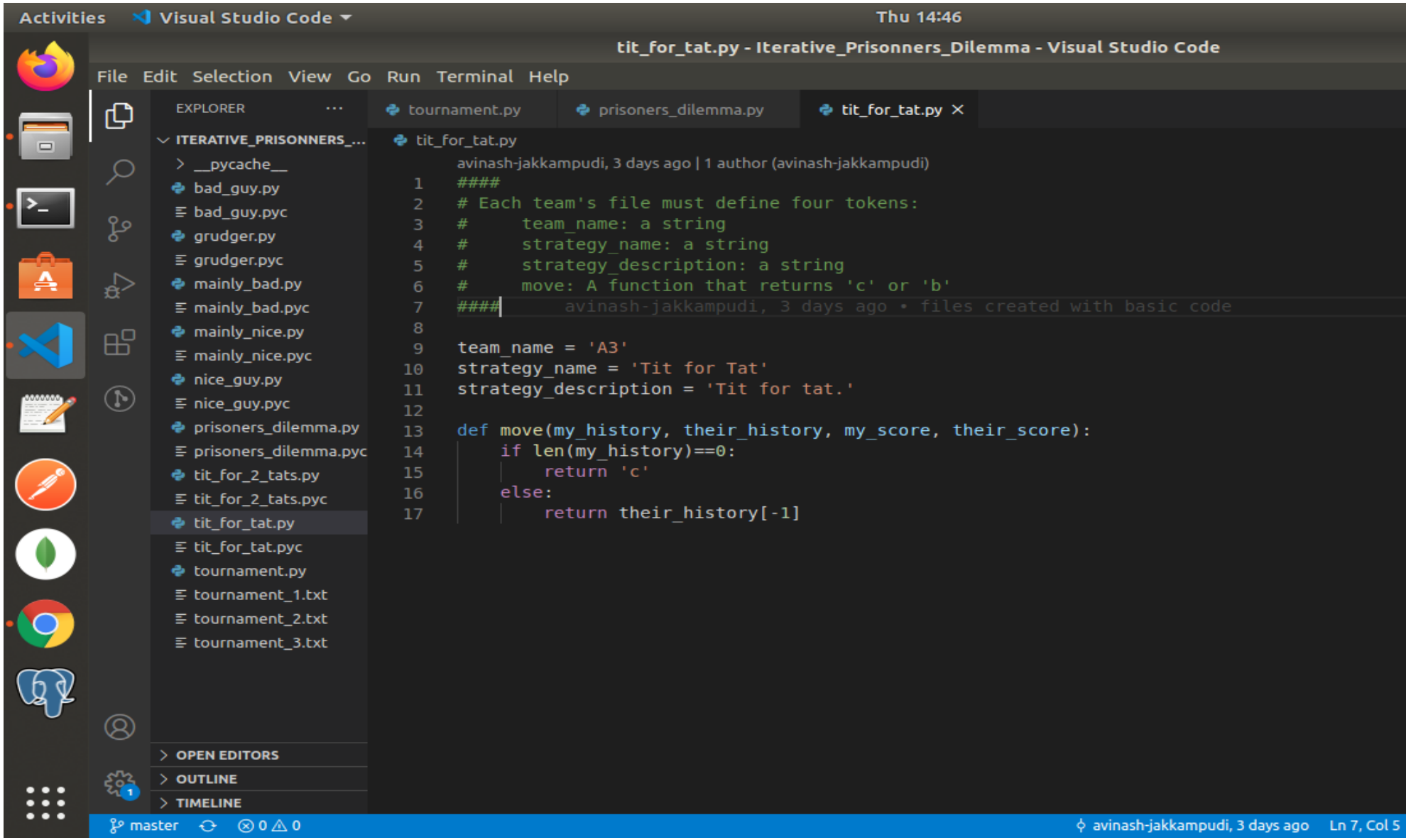
Bad Guy


Activities
Visual Studio Code
Thu 14:45
mainly_nice.py - Iterative_Prisonners_Dilemma - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER
ITERATIVE_PRISONNERS_...
__pycache__
bad_guy.py
bad_guy.pyc
grudger.py
grudger.pyc
mainly_bad.py
mainly_bad.pyc
mainly_nice.py
mainly_nice.pyc
nice_guy.py
nice_guy.pyc
prisoners_dilemma.py
prisoners_dilemma.pyc
tit_for_2_tats.py
tit_for_2_tats.pyc
tit_for_tat.py
tit_for_tat.pyc
tournament.py
tournament_1.txt
tournament_2.txt
tournament_3.txt
OPEN EDITORS
OUTLINE
TIMELINE
tournament.py
prisoners_dilemma.py
mainly_nice.py
mainly_nice.py
You, 2 days ago | 2 authors (avinash-jakkampudi and others)
1 ####
2 # Each team's file must define four tokens:
3 # team_name: a string
4 # strategy_name: a string
5 # strategy_description: a string
6 # move: A function that returns 'c' or 'b'
7 ####
8 avinash-jakkampudi, 3 days ago • files created with basic code
9 team_name = 'A4'
10 strategy_name = 'Mainly nice'
11 strategy_description = 'Mainly nice.'
12
13 def move(my_history, their_history, my_score, their_score):
14
15 if len(my_history) < 5:
16 return 'c'
17 else:
18 if len(my_history) % 2 == 0:
19 return 'c'
20 else:
21 return 'b'
22
master
0 0
avinash-jakkampudi, 3 days ago
Ln 8, Col 1

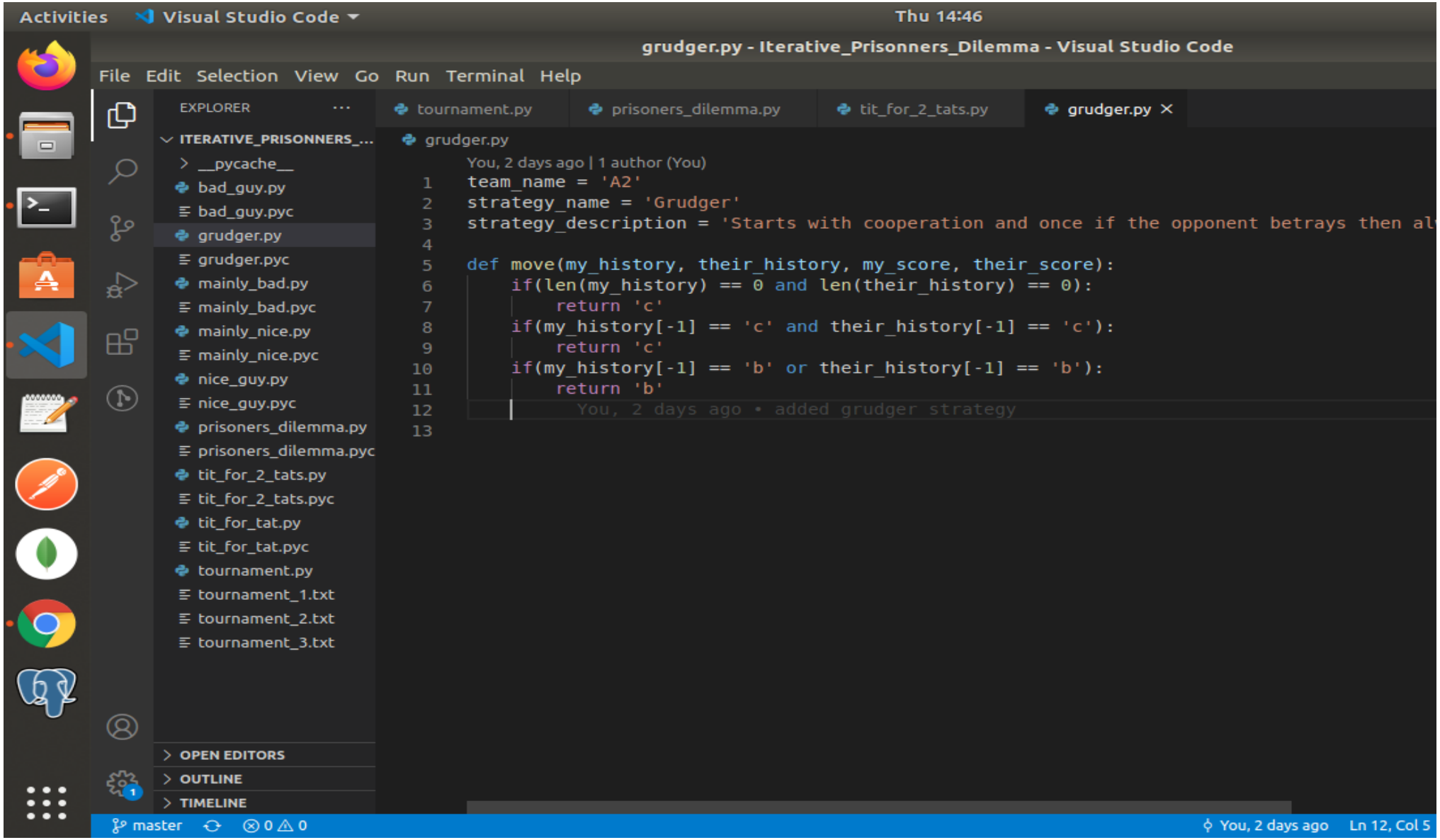
Mainly Nice


Activities
Visual Studio Code
Thu 14:46
mainly_bad.py - Iterative_Prisonners_Dilemma - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER
ITERATIVE_PRISONNERS_...
__pycache__
bad_guy.py
bad_guy.pyc
grudger.py
grudger.pyc
mainly_bad.py
mainly_bad.pyc
mainly_nice.py
mainly_nice.pyc
nice_guy.py
nice_guy.pyc
prisoners_dilemma.py
prisoners_dilemma.pyc
tit_for_2_tats.py
tit_for_2_tats.pyc
tit_for_tat.py
tit_for_tat.pyc
tournament.py
tournament_1.txt
tournament_2.txt
tournament_3.txt
OPEN EDITORS
OUTLINE
TIMELINE
tournament.py
prisoners_dilemma.py
mainly_bad.py
mainly_bad.py
You, 2 days ago | 2 authors (avinash-jakkampudi and others)
1 ####
2 # Each team's file must define four tokens:
3 # team_name: a string
4 # strategy_name: a string
5 # strategy_description: a string
6 # move: A function that returns 'c' or 'b'
7 ####
8
9 team_name = 'A5'
10 strategy_name = 'Mainly bad'
11 strategy_description = 'Mainly bad.'
12
13 def move(my_history, their_history, my_score, their_score):
14
15 if len(my_history) < 5: You, 2 days ago • added new strategies and updated code
16 return 'b'
17 else:
18 if len(my_history) % 2 == 0:
19 return 'c'
20 else:
21 return 'b'
22
master
0 0
You, 2 days ago
Ln 15, Col 27

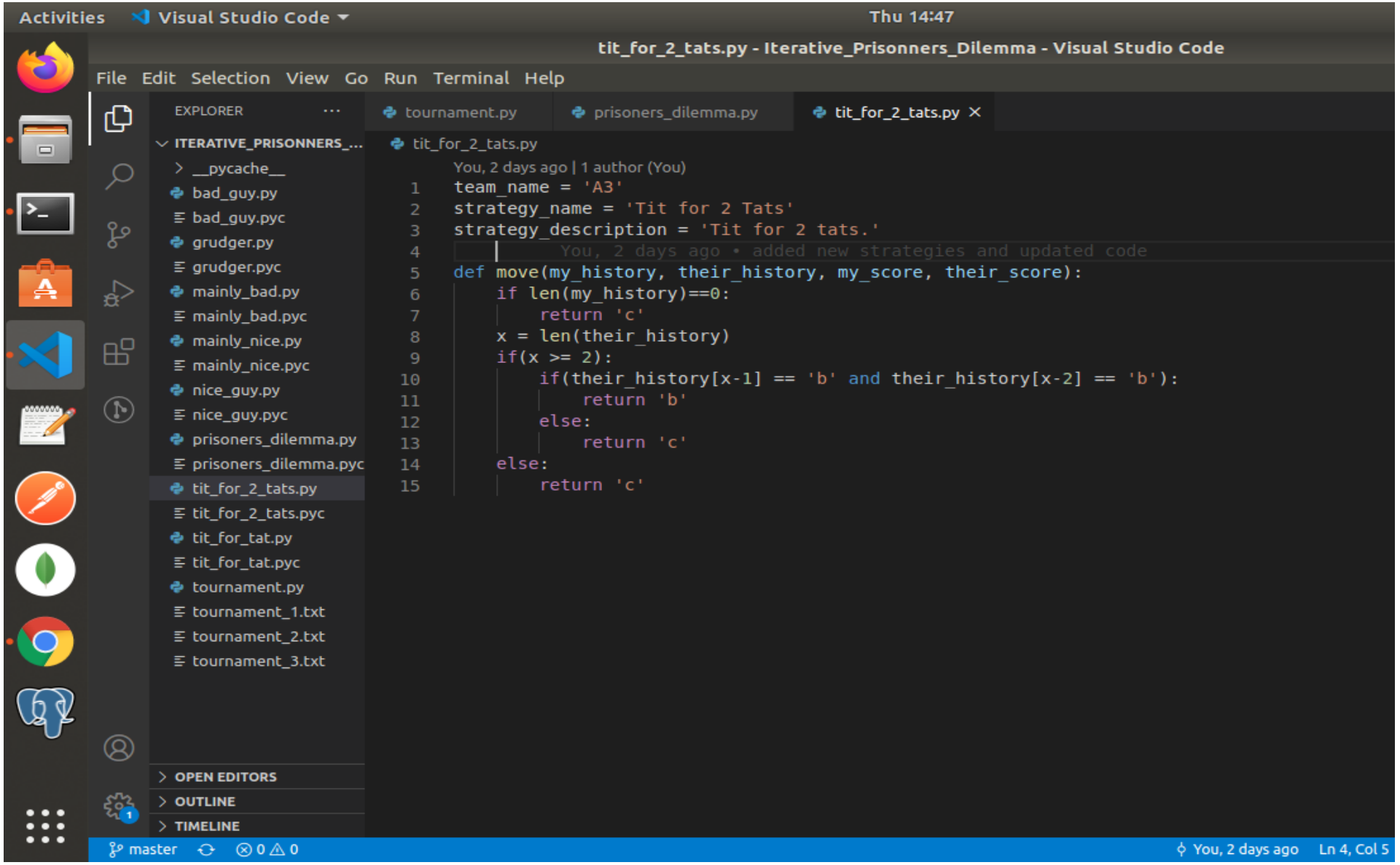
Mainly Bad


Activities
Visual Studio Code
Thu 14:46
tit_for_tat.py - Iterative_Prisonners_Dilemma - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER
ITERATIVE_PRISONNERS_...
__pycache__
bad_guy.py
bad_guy.pyc
grudger.py
grudger.pyc
mainly_bad.py
mainly_bad.pyc
mainly_nice.py
mainly_nice.pyc
nice_guy.py
nice_guy.pyc
prisoners_dilemma.py
prisoners_dilemma.pyc
tit_for_2_tats.py
tit_for_2_tats.pyc
tit_for_tat.py
tit_for_tat.pyc
tournament.py
tournament_1.txt
tournament_2.txt
tournament_3.txt
OPEN EDITORS
OUTLINE
TIMELINE
tournament.py
prisoners_dilemma.py
tit_for_tat.py
tit_for_tat.py
avinash-jakkampudi, 3 days ago | 1 author (avinash-jakkampudi)
1 ####
2 # Each team's file must define four tokens:
3 # team_name: a string
4 # strategy_name: a string
5 # strategy_description: a string
6 # move: A function that returns 'c' or 'b'
7 #### avinash-jakkampudi, 3 days ago • files created with basic code
8
9 team_name = 'A3'
10 strategy_name = 'Tit for Tat'
11 strategy_description = 'Tit for tat.'
12
13 def move(my_history, their_history, my_score, their_score):
14 if len(my_history)==0:
15 return 'c'
16 else:
17 return their_history[-1]
master
0 0
avinash-jakkampudi, 3 days ago
Ln 7, Col 5

Tit for Tat


Activities
Visual Studio Code
Thu 14:46
grudger.py - Iterative_Prisonners_Dilemma - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER
ITERATIVE_PRISONNERS_...
__pycache__
bad_guy.py
bad_guy.pyc
grudger.py
grudger.pyc
mainly_bad.py
mainly_bad.pyc
mainly_nice.py
mainly_nice.pyc
nice_guy.py
nice_guy.pyc
prisoners_dilemma.py
prisoners_dilemma.pyc
tit_for_2_tats.py
tit_for_2_tats.pyc
tit_for_tat.py
tit_for_tat.pyc
tournament.py
tournament_1.txt
tournament_2.txt
tournament_3.txt
OPEN EDITORS
OUTLINE
TIMELINE
tournament.py
prisoners_dilemma.py
tit_for_2_tats.py
grudger.py
grudger.py
You, 2 days ago | 1 author (You)
1 team_name = 'A2'
2 strategy_name = 'Grudger'
3 strategy_description = 'Starts with cooperation and once if the opponent betrays then al
4
5 def move(my_history, their_history, my_score, their_score):
6 if(len(my_history) == 0 and len(their_history) == 0):
7 return 'c'
8 if(my_history[-1] == 'c' and their_history[-1] == 'c'):
9 return 'c'
10 if(my_history[-1] == 'b' or their_history[-1] == 'b'):
11 return 'b'
12 You, 2 days ago • added grudger strategy
13
master
0 0
You, 2 days ago
Ln 12, Col 5

Grudger



Tit for Two Tats

Tasks Implemented

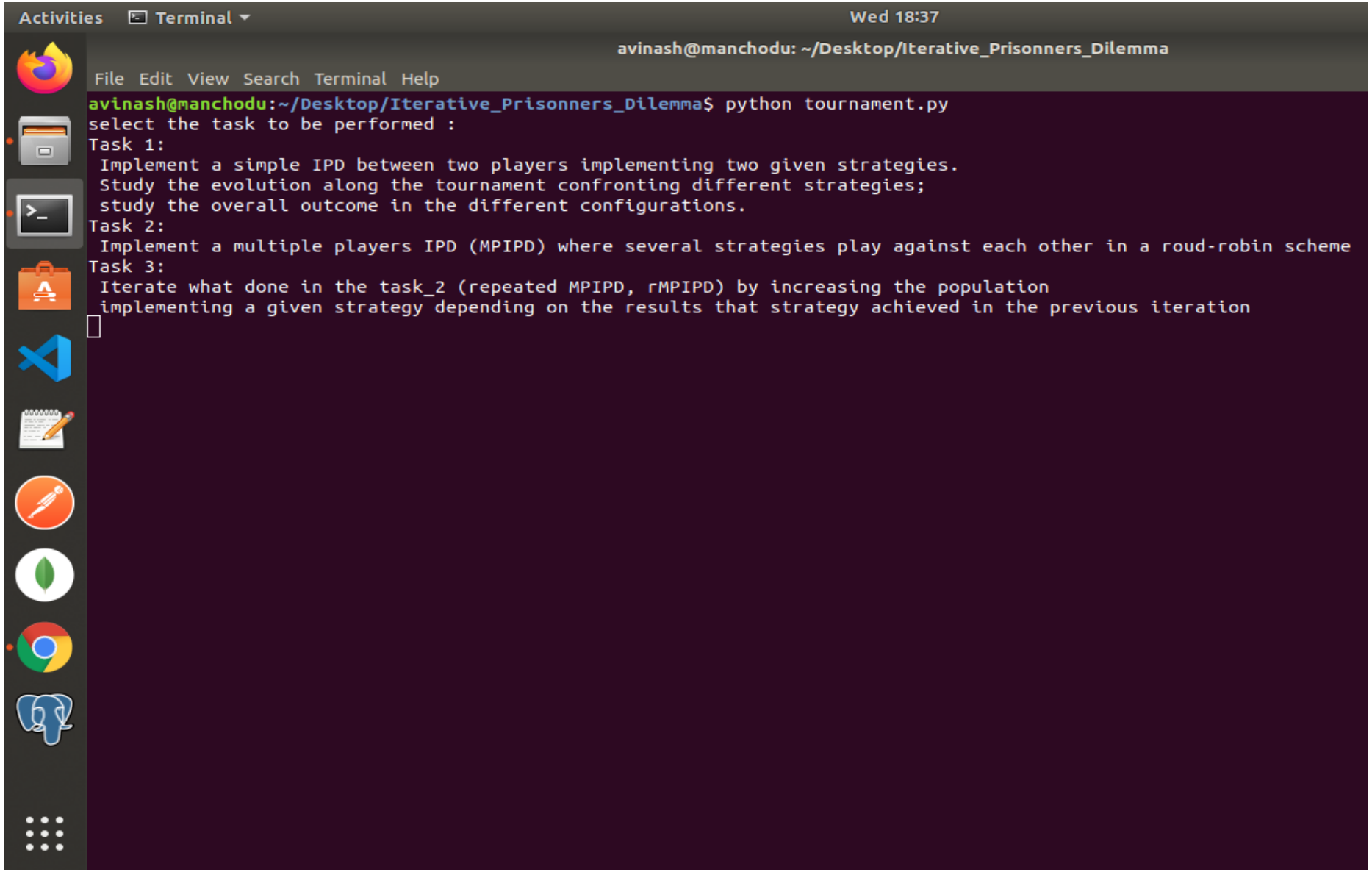
1. Implement a simple IPD between two players implementing two given strategies. Study the evolution along the tournament confronting different strategies; study the overall outcome in the different configurations.

C:\Users\user\Desktop\HLP_Project\Iterative_Prisonners_Dilemma\tournament.py

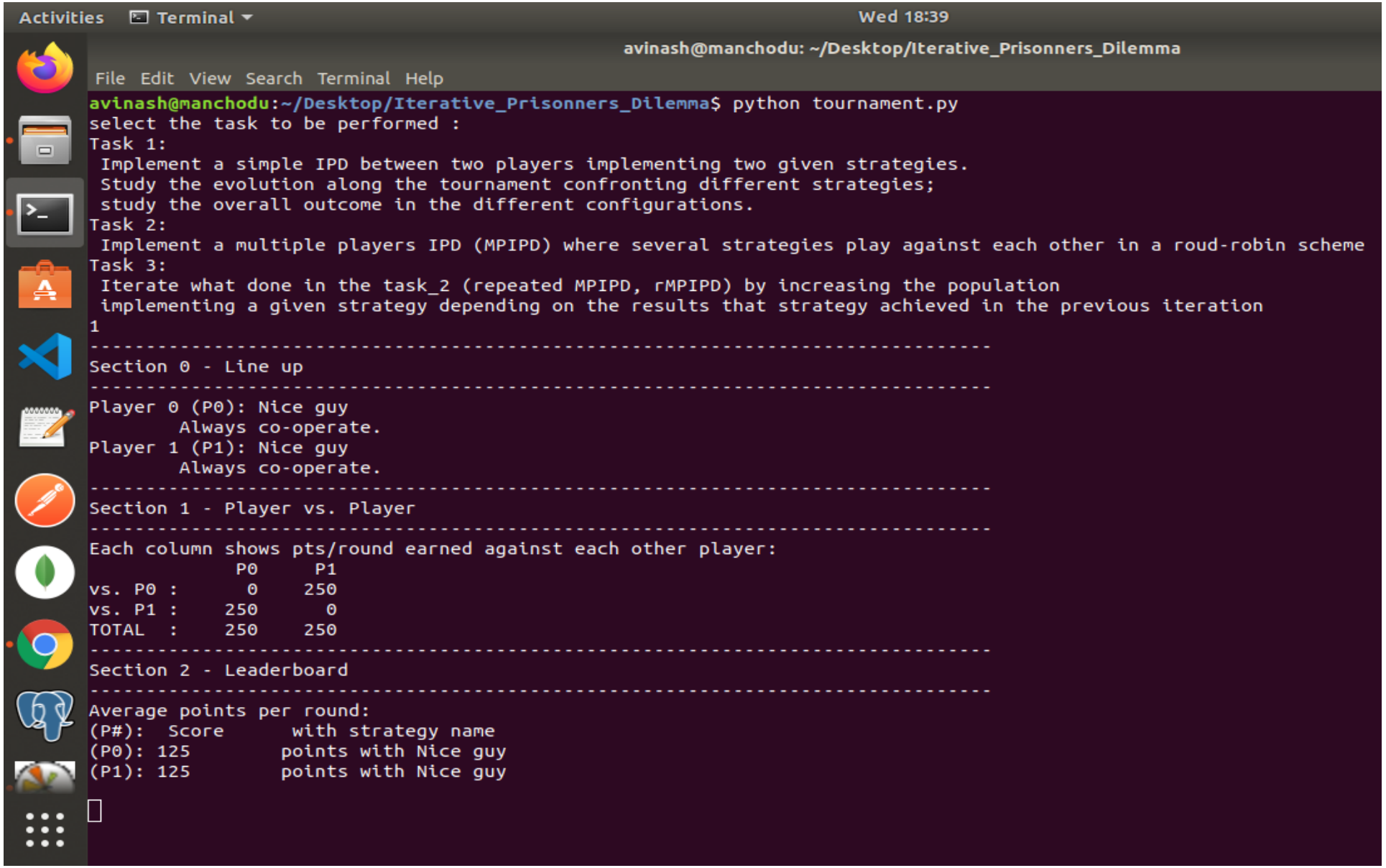
tournament.py

```
25
26  def task_one():
27      number_of_players = 2
28      players_strategies = [0]*number_of_players
29      strategies_length = len(strategies)
30      all_section = ''
31      i = 0
32      while(i < strategies_length):
33          players_strategies[0] = strategies[i]
34          j = i
35          while(j < strategies_length):
36              players_strategies[1] = strategies[j]
37              modules = players_strategies
38              scores, moves = tournament(modules)
39              all_section = create_reports(modules, scores,
40              j = j+1
41          i = i+1
42      post_to_file(all_section)
43
```

Task 1



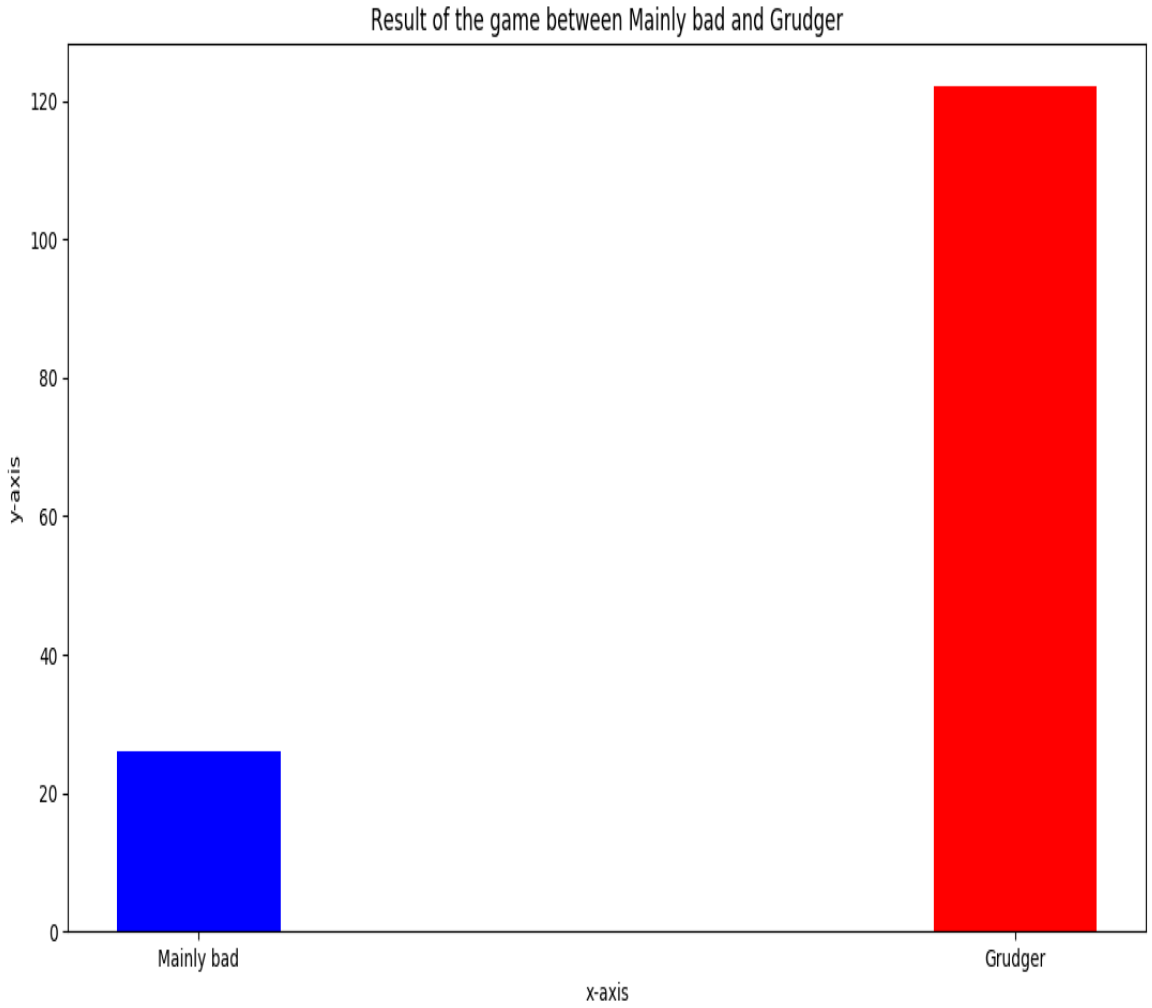
Select the Task to be performed



Selected Task 1

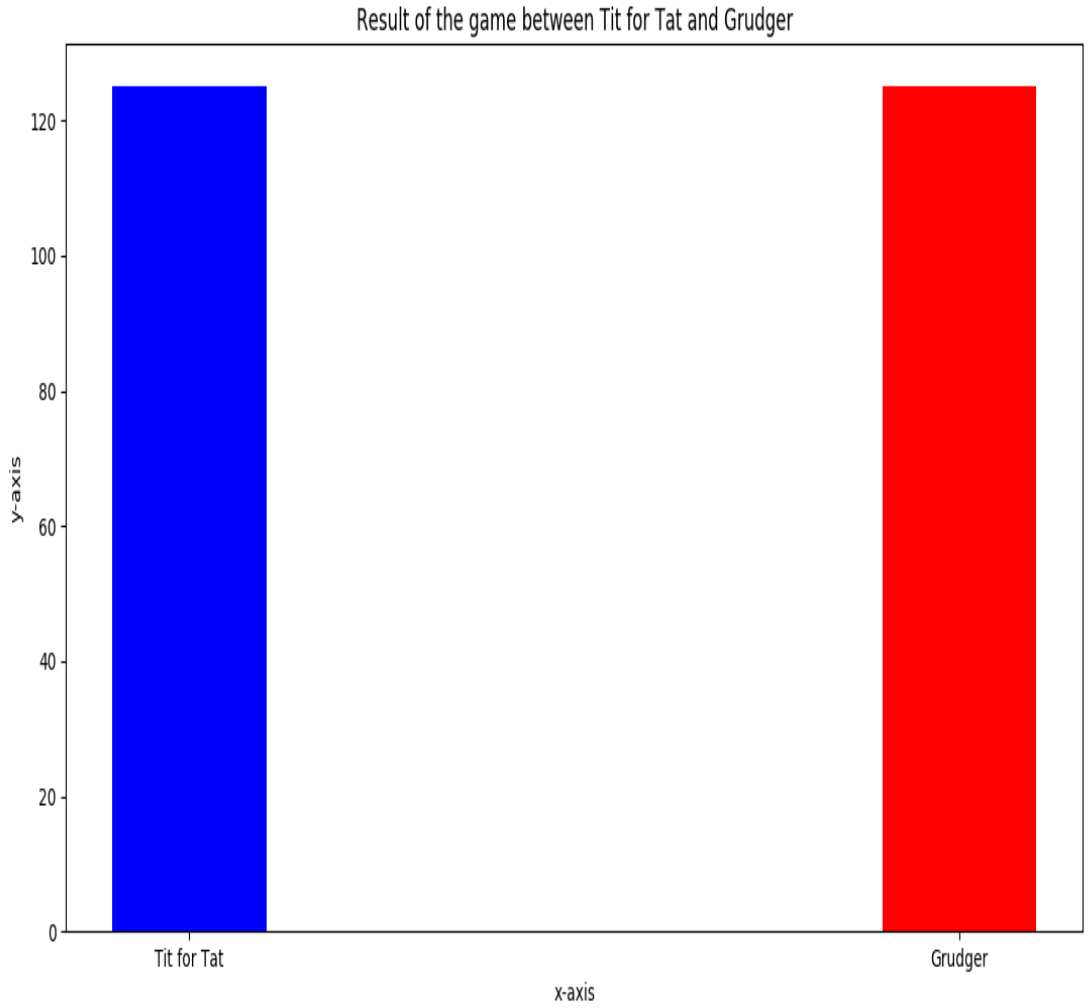
atplotlib ▾ Wed 18:41 ties Matplotlib ▾ Wed 18:41

Figure 1



x= y=77.255

Figure 1



x= y=


Activities
Text Editor
Wed 18:43
Open
tournament_1.txt
~/Desktop/Iterative_Prisonners_Dilemma
Save
Section 0 - Line up
Player 0 (P0): Grudger
Starts with cooperation and once if the opponent betrays then always grudger betrays irrespective of the opponent move.
Player 1 (P1): Grudger
Starts with cooperation and once if the opponent betrays then always grudger betrays irrespective of the opponent move.
Section 1 - Player vs. Player
Each column shows pts/round earned against each other player:
P0 P1
vs. P0 : 0 250
vs. P1 : 250 0
TOTAL : 250 250
Section 2 - Leaderboard
Average points per round:
(P#): Score with strategy name
(P0): 125 points with Grudger
(P1): 125 points with Grudger
Section 0 - Line up
Player 0 (P0): Grudger
Starts with cooperation and once if the opponent betrays then always grudger betrays irrespective of the opponent move.
Player 1 (P1): Tit for 2 Tats
Tit for 2 tats.
Section 1 - Player vs. Player
Each column shows pts/round earned against each other player:
P0 P1
Plain Text
Tab Width: 8
Ln 9, Col 30
INS

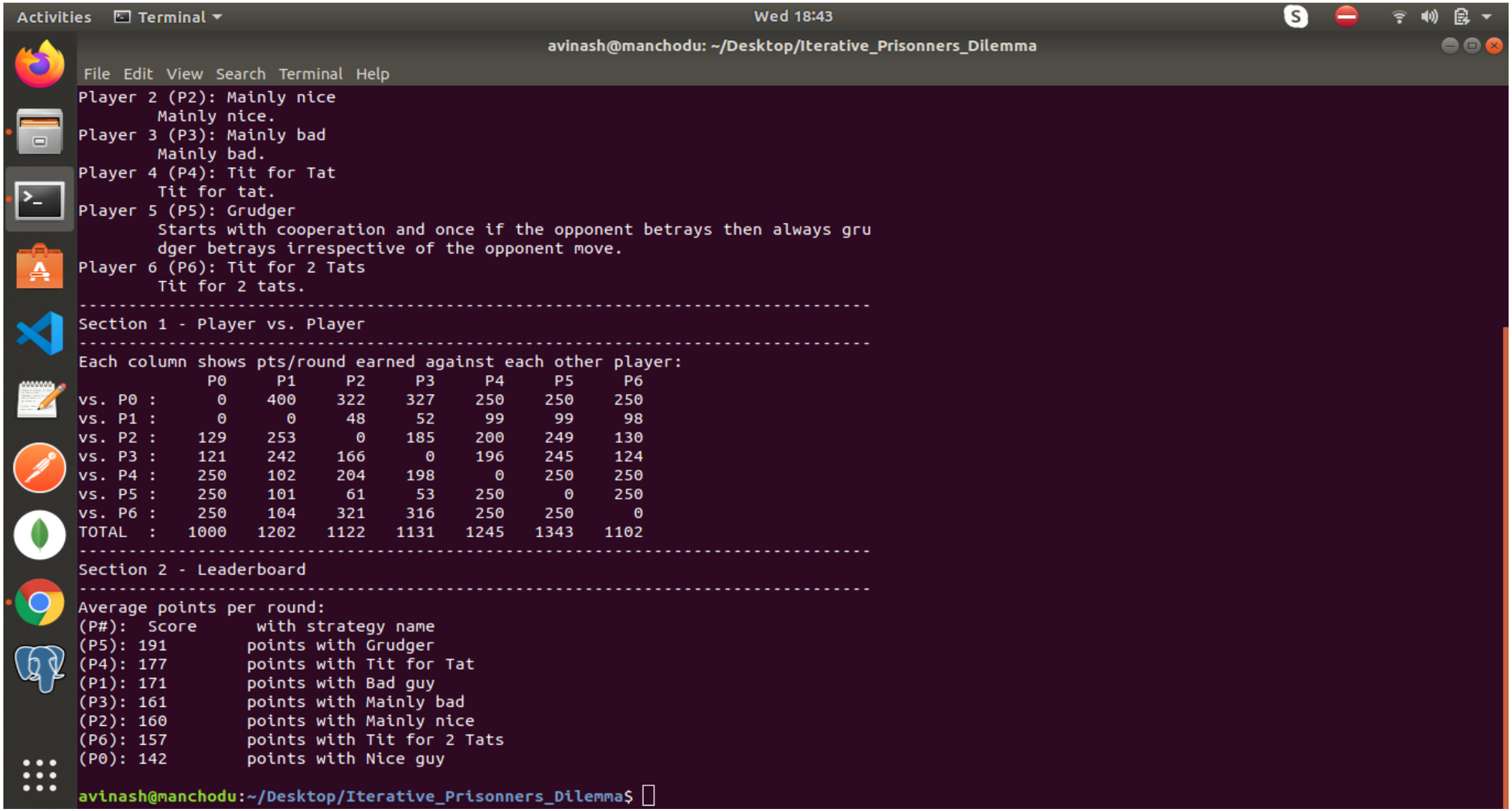
2. Implement a multiple players IPD (MPIPD) where several strategies play against each other in a round-robin scheme

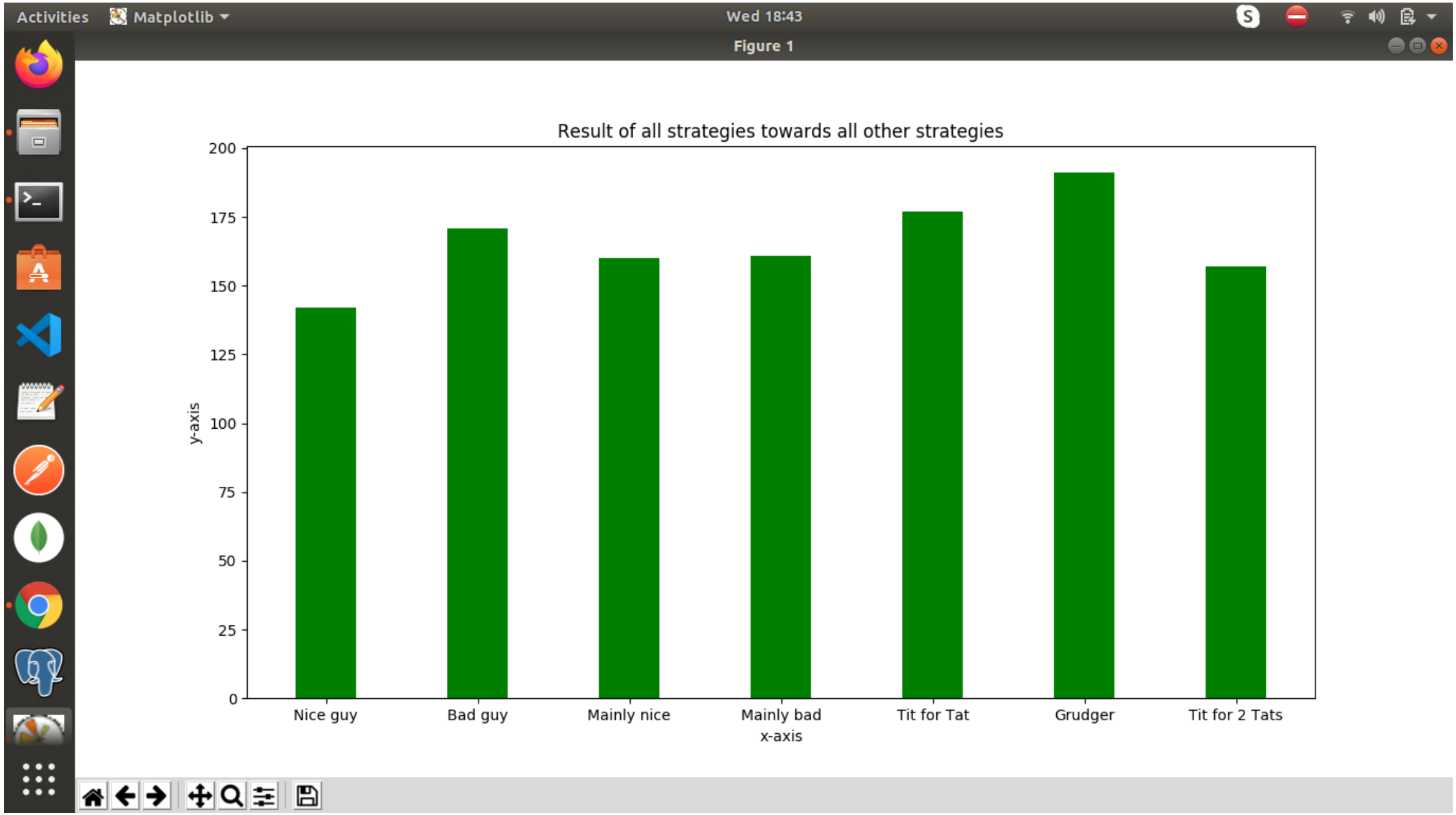
C:\Users\user\Desktop\HLP_Project\Iterative_Prisonners_Dilemma\tournament.py

tournament.py

```
43
44  def task_two():
45      all_section = ''
46      modules = strategies
47      scores, moves = tournament(modules)
48      plot_graph(scores,modules)
49      all_section = create_reports(modules, scores, moves,all_section)
50      post_to_file(all_section)
51
```

Task 2


Activities
Terminal
Wed 18:43
avinash@manchodu: ~/Desktop/Iterative_Prisonners_Dilemma
File Edit View Search Terminal Help
Player 2 (P2): Mainly nice
Mainly nice.
Player 3 (P3): Mainly bad
Mainly bad.
Player 4 (P4): Tit for Tat
Tit for tat.
Player 5 (P5): Grudger
Starts with cooperation and once if the opponent betrays then always gru
dger betrays irrespective of the opponent move.
Player 6 (P6): Tit for 2 Tats
Tit for 2 tats.
Section 1 - Player vs. Player
Each column shows pts/round earned against each other player:
P0 P1 P2 P3 P4 P5 P6
vs. P0 : 0 400 322 327 250 250 250
vs. P1 : 0 0 48 52 99 99 98
vs. P2 : 129 253 0 185 200 249 130
vs. P3 : 121 242 166 0 196 245 124
vs. P4 : 250 102 204 198 0 250 250
vs. P5 : 250 101 61 53 250 0 250
vs. P6 : 250 104 321 316 250 250 0
TOTAL : 1000 1202 1122 1131 1245 1343 1102
Section 2 - Leaderboard
Average points per round:
(P#): Score with strategy name
(P5): 191 points with Grudger
(P4): 177 points with Tit for Tat
(P1): 171 points with Bad guy
(P3): 161 points with Mainly bad
(P2): 160 points with Mainly nice
(P6): 157 points with Tit for 2 Tats
(P0): 142 points with Nice guy
avinash@manchodu:~/Desktop/Iterative_Prisonners_Dilemma$


Activities
Matplotlib
Wed 18:43
Figure 1
Result of all strategies towards all other strategies
y-axis
200
175
150
125
100
75
50
25
0
Nice guy
Bad guy
Mainly nice
Mainly bad
Tit for Tat
Grudger
Tit for 2 Tats
x-axis

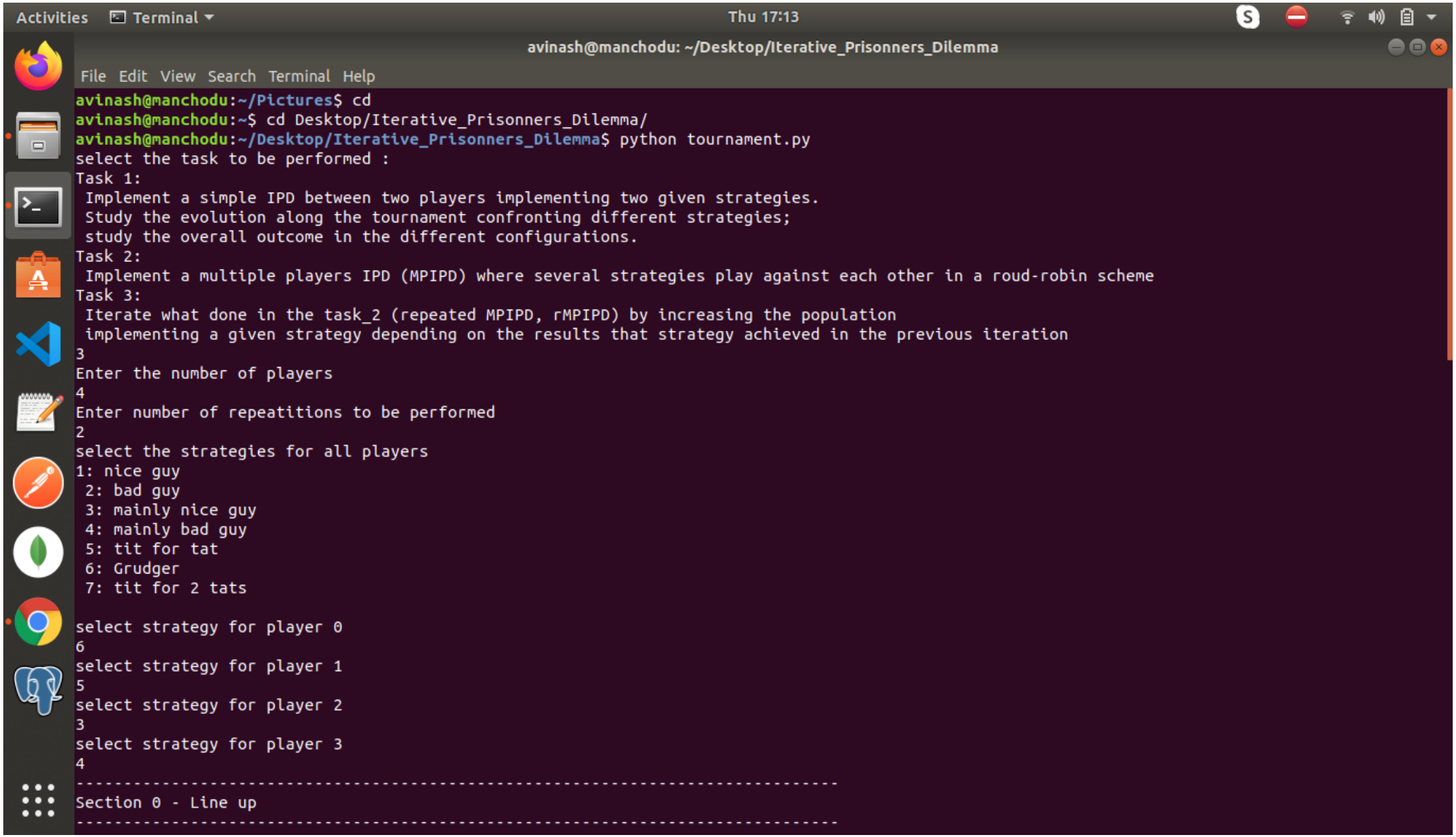
3. Iterate what done in the previous task (repeated MPIPD, rMPIPD) by increasing the population implementing a given strategy depending on the results that strategy achieved in the previous iteration

C:\Users\user\Desktop\HLP_Project\Iterative_Prisonners_Dilemma\tournament.py

tournament.py

```
52  def task_three():
53      all_section = ''
54      number_of_players = int(input('Enter the number of players \n'))
55      if(number_of_players > 1):
56          number_of_repeatitions = int(input('Enter number of repeatitions to be performed \n'))
57          if(number_of_repeatitions < 1):
58              print("Please enter valid number of repeatitions to be performed which should be > 0")
59              number_of_repeatitions = int(input('Enter number of repititions to be performed \n'))
60              if(number_of_repeatitions < 1):
61                  print("Invalid enter of repeatitions please start the tournament again \n")
62                  return ""
63          print("select the strategies for all players")
64          print("1: nice guy \n 2: bad guy \n 3: mainly nice guy \n 4: mainly bad guy \n 5: tit for tat \n 6: Gru
65          players_strategies = [0]*number_of_players
66          for each_player in range(number_of_players):
67              print("select strategy for player "+str(each_player))
68              strategy_number = int(input())
69              players_strategies[each_player] = strategies[strategy_number-1]
70          current_repeatition = 0
71          while(current_repeatition < number_of_repeatitions):
72              modules = players_strategies
73              scores, moves = tournament(modules)
74              all_section = create_reports(modules, scores, moves,all_section)
75              if current_repeatition < number_of_repeatitions-1:
76                  print("Enter 0: to change the strategies of the players \n 1: to continue with existing strateg
77                  change_strategy = int(input())
78                  if(change_strategy == 0):
79                      print("1: nice guy \n 2: bad guy \n 3: mainly nice guy \n 4: mainly bad guy \n 5: tit for t
80                      players_strategies = [0]*number_of_players
81                      for each_player in range(number_of_players):
82                          print("select strategy for player "+str(each_player))
83                          strategy_number = int(input())
84                          players_strategies[each_player] = strategies[strategy_number-1]
85              current_repeatition = current_repeatition + 1
86      else:
87          print("Tournament is not possible with "+str(number_of_players)+" players")
```

Task 3


Activities
Terminal
Thu 17:13
avinash@manchodu: ~/Desktop/Iterative_Prisonners_Dilemma
File Edit View Search Terminal Help
avinash@manchodu:~/Pictures$ cd
avinash@manchodu:~$ cd Desktop/Iterative_Prisonners_Dilemma/
avinash@manchodu:~/Desktop/Iterative_Prisonners_Dilemma$ python tournament.py
select the task to be performed :
Task 1:
Implement a simple IPD between two players implementing two given strategies.
Study the evolution along the tournament confronting different strategies;
study the overall outcome in the different configurations.
Task 2:
Implement a multiple players IPD (MPIPD) where several strategies play against each other in a roud-robin scheme
Task 3:
Iterate what done in the task_2 (repeated MPIPD, rMPIPD) by increasing the population
implementing a given strategy depending on the results that strategy achieved in the previous iteration
3
Enter the number of players
4
Enter number of repeatitions to be performed
2
select the strategies for all players
1: nice guy
2: bad guy
3: mainly nice guy
4: mainly bad guy
5: tit for tat
6: Grudger
7: tit for 2 tats
select strategy for player 0
6
select strategy for player 1
5
select strategy for player 2
3
select strategy for player 3
4
Section 0 - Line up

Activities Terminal Wed 18:50

avinash@manchodu: ~/Desktop/Iterative_Prisonners_Dilemma

File Edit View Search Terminal Help

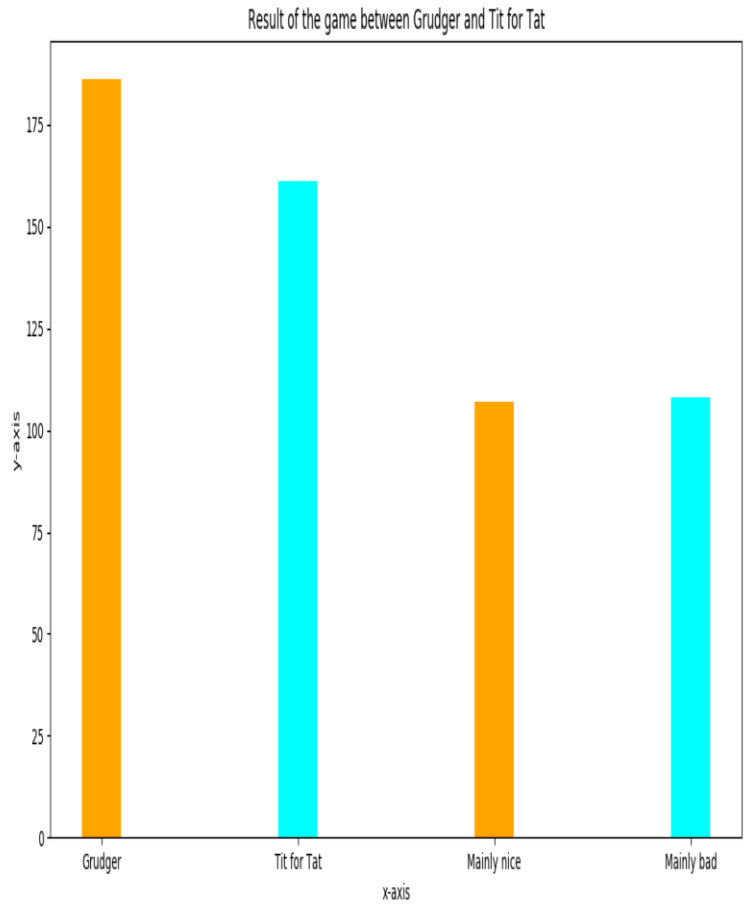
```
Enter 0: to change the strategies of the players
 1: to continue with existing strategies

1
--------------------------------------------------------------------------------
Section 0 - Line up
--------------------------------------------------------------------------------
Player 0 (P0): Grudger
       Starts with cooperation and once if the opponent betrays then always gru
       dger betrays irrespective of the opponent move.
Player 1 (P1): Tit for Tat
       Tit for tat.
Player 2 (P2): Mainly nice
       Mainly nice.
Player 3 (P3): Mainly bad
       Mainly bad.
--------------------------------------------------------------------------------
Section 1 - Player vs. Player
--------------------------------------------------------------------------------
Each column shows pts/round earned against each other player:
             P0     P1     P2     P3
vs. P0 :      0    250     59     54
vs. P1 :    250      0    204    195
vs. P2 :    248    200      0    183
vs. P3 :    242    195    167      0
TOTAL  :    740    645    430    432
--------------------------------------------------------------------------------
Section 2 - Leaderboard
--------------------------------------------------------------------------------
Average points per round:
(P#):  Score      with strategy name
(P0): 185        points with Grudger
(P1): 161        points with Tit for Tat
(P3): 108        points with Mainly bad
(P2): 107        points with Mainly nice

Enter 0: to change the strategies of the players
 1: to continue with existing strategies
```

Activities Matplotlib Wed 18:48

Figure 1



x= y=127.868

Crucial Code for the Tournament

sers\user\Desktop\HLP_Project\Iterative_Prisonners_Dilemma\tournament.py

tournament.py | prisoners_dilemma.py

```
105
106 def tournament(modules):
107     for module in modules:
108             importlib.reload(module)
109             for required_variable in ['strategy_name', 'strategy_description']:
110                 if not hasattr(module, required_variable):
111                     setattr(module, required_variable, 'missing assignment')
112
113     scores, moves = prisoners_dilemma.main_play(modules)
114     return scores, moves
115
```

C:\Users\user\Desktop\HLP_Project\Iterative_Prisonners_Dilemma\prisoners_dilemma.py

tournament.py | prisoners_dilemma.py

```python
import random

def main_play(modules):
    scores, moves = play_tournament(modules)
    # section0, section1, section2, section3 = make_reports(modules, scores, moves)
    # print(section0+section1+section2)
    # post_to_file(section0+section1+section2 )
    return scores, moves

def play_tournament(modules):

    zeros_list = [0]*len(modules)
    scores = [zeros_list[:] for module in modules]
    moves = [zeros_list[:] for module in modules]
    for first_team_index in range(len(modules)):
        for second_team_index in range(first_team_index):
            player1 = modules[first_team_index]
            player2 = modules[second_team_index]
            score1, score2, moves1, moves2 = play_iterative_rounds(player1, player2)
            scores[first_team_index][second_team_index] = score1/len(moves1)
            moves[first_team_index][second_team_index] = moves1
            scores[second_team_index][first_team_index] = score2/len(moves2)
            moves[second_team_index][first_team_index] = moves2

        scores[first_team_index][first_team_index] = 0
        moves[first_team_index][first_team_index] = ''
    return scores, moves

```

C:\Users\user\Desktop\HLP_Project\Iterative_Prisonners_Dilemma\prisoners_dilemma.py

tournament.py | prisoners_dilemma.py

```
30  def play_iterative_rounds(player1, player2):
31
32      number_of_rounds = random.randint(100, 200)
33      moves1 = ''
34      moves2 = ''
35      score1 = 0
36      score2 = 0
37      for round in range(number_of_rounds):
38          score1, score2, moves1, moves2 = play_round(player1, player2, score1, score2, moves1, moves2)
39      return (score1, score2, moves1, moves2)
40
```

C:\Users\user\Desktop\HLP_Project\Iterative_Prisonners_Dilemma\prisoners_dilemma.py

tournament.py | prisoners_dilemma.py

```
41  def play_round(player1, player2, score1, score2, moves1, moves2):
42
43      RELEASE = 250
44      TREAT = 400
45      SEVERE_PUNISHMENT = 0
46      PUNISHMENT = 100
47
48      # Keep T > R > P > S
49      # Keep 2R > T + S
50
51      ERROR = -250
52
53      action1 = player1.move(moves1, moves2, score1, score2)
54      action2 = player2.move(moves2, moves1, score2, score1)
55      if (type(action1) != str) or (len(action1) != 1):
56          action1=' '
57      if (type(action2) != str) or (len(action2) != 1):
58          action2=' '
59
60      actions = action1 + action2
61      if actions == 'cc':
62          score1 += RELEASE
63          score2 += RELEASE
64      elif actions == 'cb':
65          score1 += SEVERE_PUNISHMENT
66          score2 += TREAT
67      elif actions == 'bc':
68          score1 += TREAT
69          score2 += SEVERE_PUNISHMENT
70      elif actions == 'bb':
71          score1 += PUNISHMENT
72          score2 += PUNISHMENT
73      else:
74          score1 += ERROR
75          score2 += ERROR
76
```

Outputs

The outputs of all tasks in the tournament will be saved as text files with task numbers.

Ex: If first task is executed then the output of the task will be saved as Tournament_1.txt

Conclusion

- Prisoner’s dilemma is still a current research area with nearly 15000 papers during the past two years (Source: Google Scholar). New strategies are developed, and old ones are reused in new areas, but there is always a problem of possibility to misjudge opponent which will bring worse results in the end.
- Individuals with more information will have advantage in most of situations so the strategies that learn about the opponents and adjust their own behaviour will certainly have an increasingly important role in the future.