Installation

1. To install all the relevant python packages, use the command on the command prompt as ‘pip install –r requirements.txt’ after extracting the zip folder
2. To run the file, on a command prompt type ‘python m1.py’

Explanation

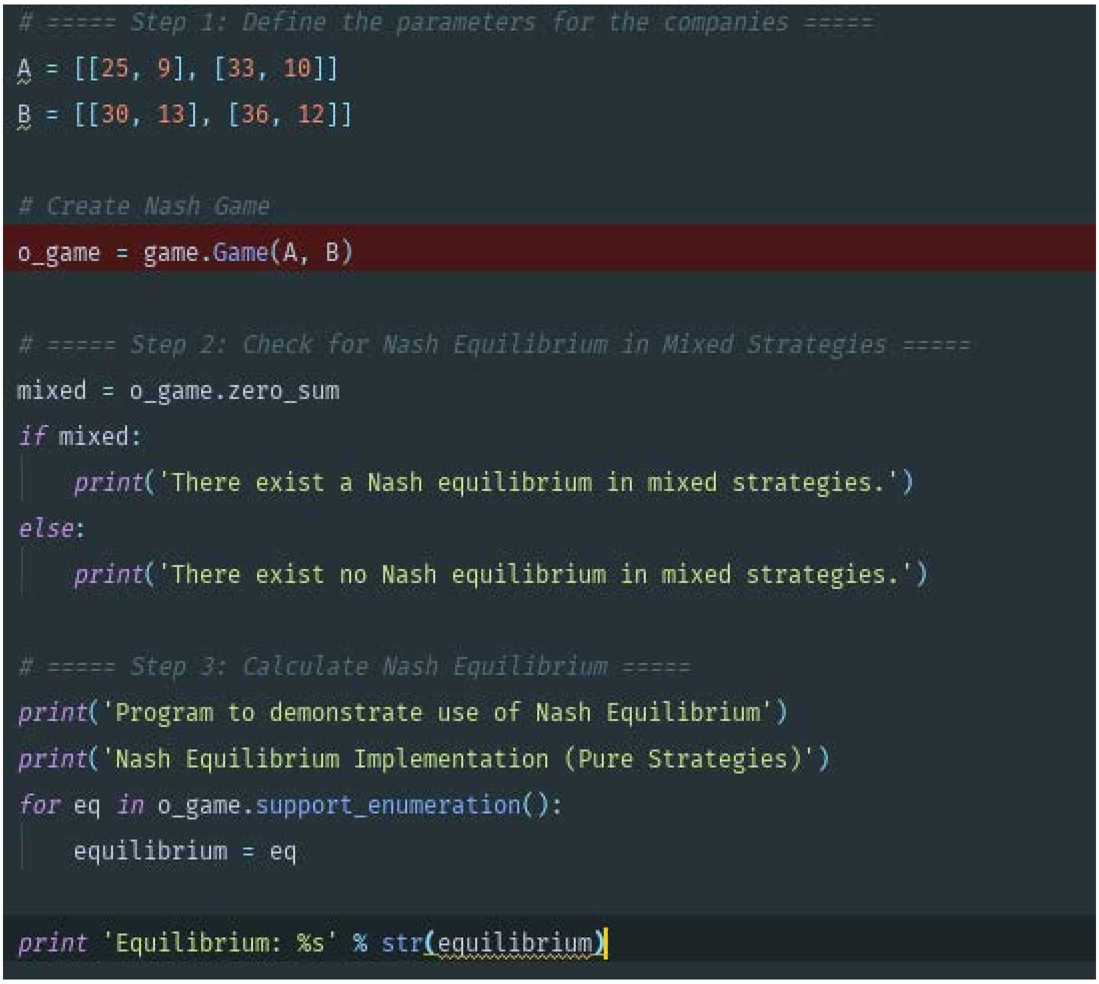
1. **Is there any mixed strategy Nash equilibrium in this game?**
   No, there is no Mixed Strategy Nash Equilibrium in this game, because the zero_sum call results into a False.

   ```
   There exist no Nash equilibrium in mixed strategies.
   ```

2. **If yes, what is the mixed strategy Nash equilibrium (or equilibria)?**
   There is no Mixed Strategy Nash Equilibrium

3. **Also, write a simple Python implementation of Game Theory to illustrate the implementation of Nash Equilibrium:**

   Using the same Matrix as represented in the question, we write a program to compute the Nash Equilibrium as follows:



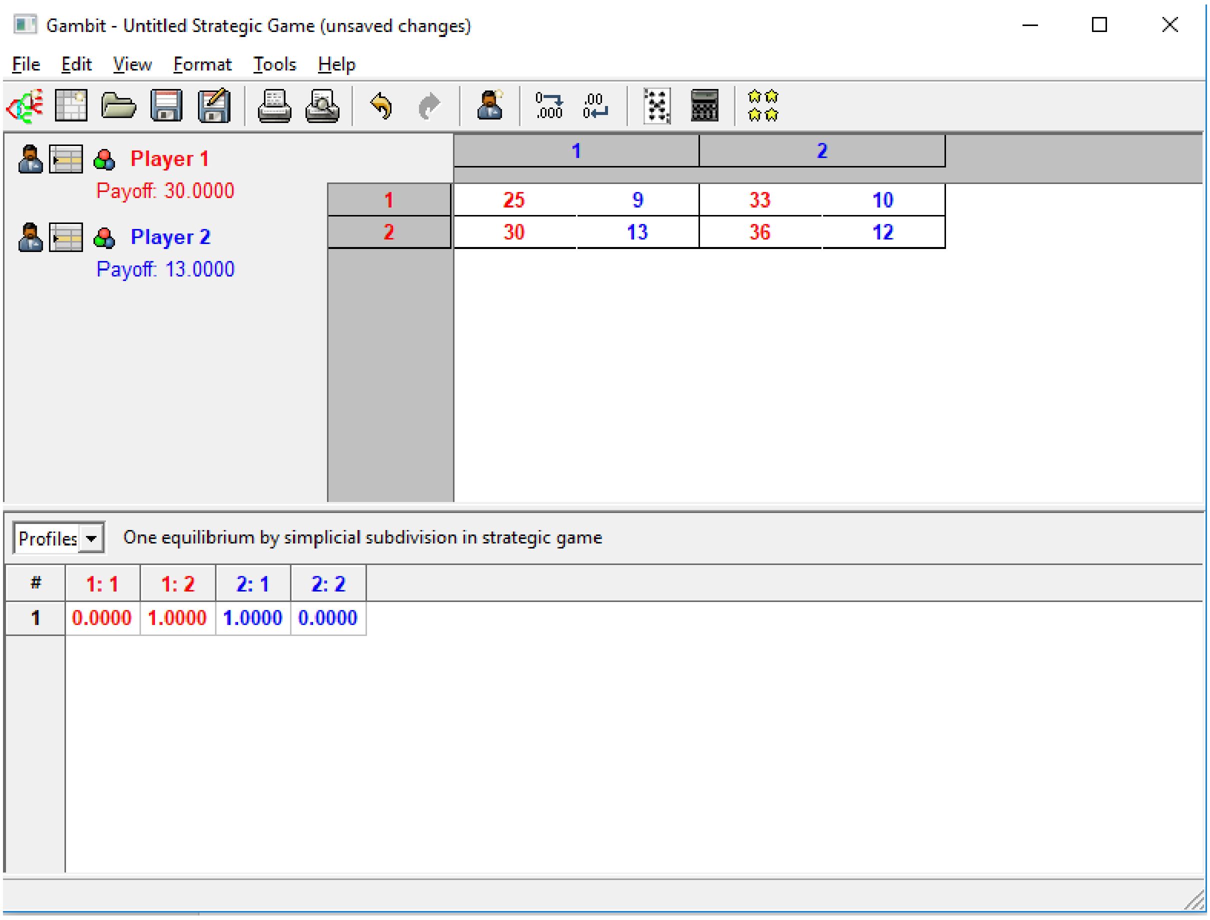
```
# ===== Step 1: Define the parameters for the companies =====
A = [[25, 9], [33, 10]]
B = [[30, 13], [36, 12]]

# Create Nash Game
o_game = game.Game(A, B)

# ===== Step 2: Check for Nash Equilibrium in Mixed Strategies =====
mixed = o_game.zero_sum
if mixed:
    print('There exist a Nash equilibrium in mixed strategies.')
else:
    print('There exist no Nash equilibrium in mixed strategies.')

# ===== Step 3: Calculate Nash Equilibrium =====
print('Program to demonstrate use of Nash Equilibrium')
print('Nash Equilibrium Implementation (Pure Strategies)')
for eq in o_game.support_enumeration():
    equilibrium = eq

print 'Equilibrium: %s' % str(equilibrium)
```

4. **If both firms decide their strategies simultaneously, what is the Nash equilibrium (or equilibria)?**
   From the aforementioned program, the Nash Equilibrium is

```
Equilibrium: (array([ 0.,  1.]), array([ 1.,  0.]))
```

5. **Which Python Packages are best suited for this implementation? Would the Python Libraries on Linear Programming be suitable for this purpose?**
   Packages such as Numpy, Scipy (which is used by the package Nash under the hood) and Nash are best suited for this implementation. Yes, python libraries on Linear Programming are suitable for this purpose

6. **Research the Gambit software (open source) and show how it can be leveraged for the above implementation.**



Player 1
Payoff: 30.0000

Player 2
Payoff: 13.0000

	1		2	
1	25	9	33	10
2	30	13	36	12

One equilibrium by simplicial subdivision in strategic game

#	1: 1	1: 2	2: 1	2: 2
1	0.0000	1.0000	1.0000	0.0000

Gambit software was used to get the same results as the code, that is array(0,1) and array(1,0). One can also look at the payoffs and figure that the best pure strategy Nash Equilibrium is <Down, Left>, that is, <30,13>