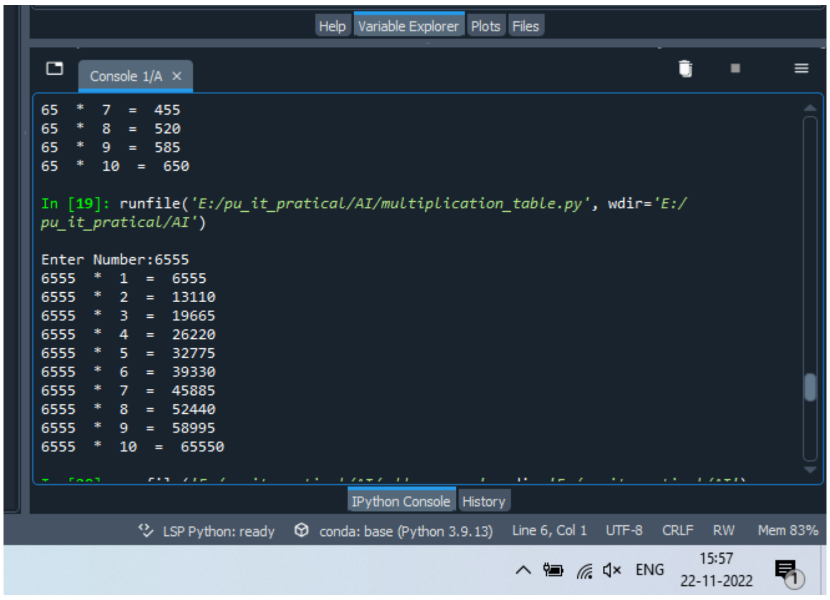
PRACTICAL-2

A)AIM: Write a python program to print the multiplication table for the given number.

Code:

```
x = int(input("Enter Number:"))
a=1
while(a<=10):
      print(x,' * ',a,' = ',a*x)
      a=a+1
```

Output:



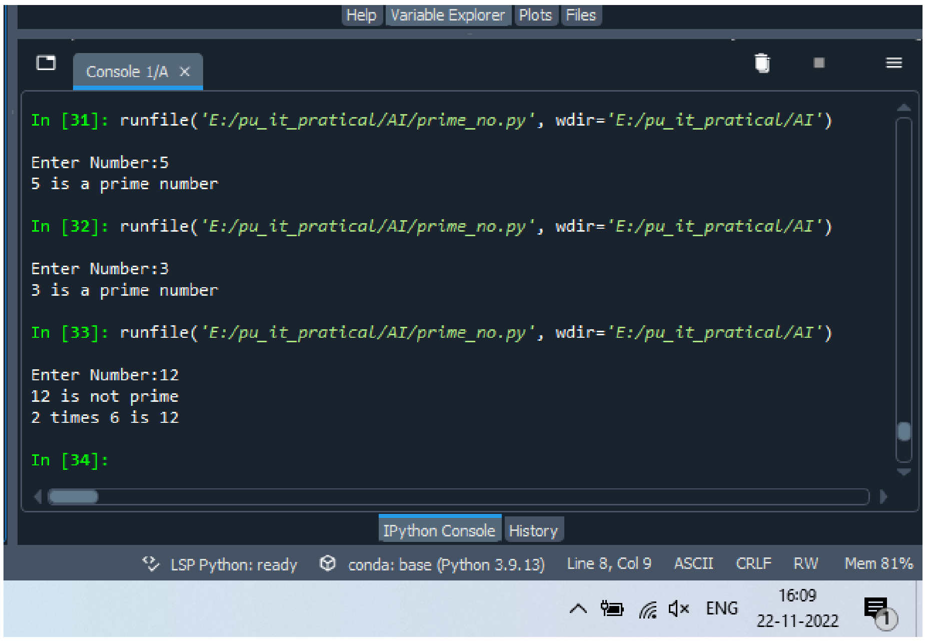
```
65  *  7  =  455
65  *  8  =  520
65  *  9  =  585
65  *  10  =  650

In [19]: runfile('E:/pu_it_pratical/AI/multiplication_table.py', wdir='E:/
pu_it_pratical/AI')

Enter Number:6555
6555  *  1  =  6555
6555  *  2  =  13110
6555  *  3  =  19665
6555  *  4  =  26220
6555  *  5  =  32775
6555  *  6  =  39330
6555  *  7  =  45885
6555  *  8  =  52440
6555  *  9  =  58995
6555  *  10  =  65550
```

B)AIM: Write a python program to check whether the given number is prime or not.

Code:

```
x = int(input("Enter Number:"))
if x > 1:
   for n in range(2, x):
      if (x % n) == 0:
         print(x, "is not prime")
         print(n, "times", x // n, "is", x)
         break
   else:
      print(x, "is a prime number")
else:
      print(x, "is not prime number")
```

Output:



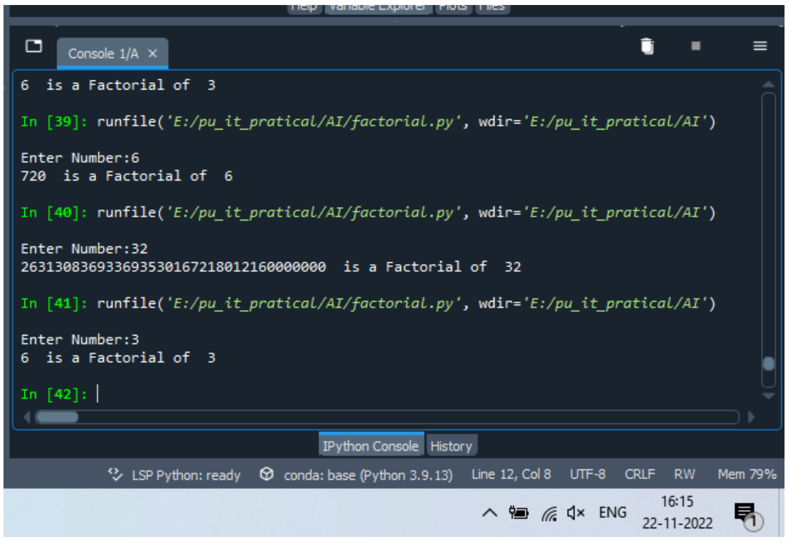
```
In [31]: runfile('E:/pu_it_pratical/AI/prime_no.py', wdir='E:/pu_it_pratical/AI')

Enter Number:5
5 is a prime number

In [32]: runfile('E:/pu_it_pratical/AI/prime_no.py', wdir='E:/pu_it_pratical/AI')

Enter Number:3
3 is a prime number

In [33]: runfile('E:/pu_it_pratical/AI/prime_no.py', wdir='E:/pu_it_pratical/AI')

Enter Number:12
12 is not prime
2 times 6 is 12

In [34]:
```

C) AIM: Write a python program to find factorial of the given number.

Code:

```
a = int(input("Enter Number:"))
p=1
i=a
while(a>1):
  p=p*a
  a=a-1
print(p," is a Factorial of ",i)
```

Output:



```
6  is a Factorial of  3

In [39]: runfile('E:/pu_it_pratical/AI/factorial.py', wdir='E:/pu_it_pratical/AI')

Enter Number:6
720  is a Factorial of  6

In [40]: runfile('E:/pu_it_pratical/AI/factorial.py', wdir='E:/pu_it_pratical/AI')

Enter Number:32
263130836933693530167218012160000000  is a Factorial of  32

In [41]: runfile('E:/pu_it_pratical/AI/factorial.py', wdir='E:/pu_it_pratical/AI')

Enter Number:3
6  is a Factorial of  3

In [42]:
```

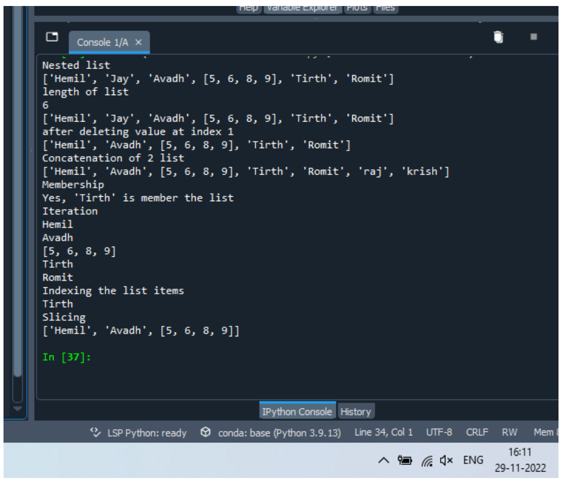
PRACTICAL-4

AIM: A) Write a python program to implement List operations (Nested List, Length, Concatenation, Membership, Iteration, Indexing and Slicing)?

Code:

```
list_name = ['Hemil','Jay','Avadh',[5,6,8,9],'Tirth','Romit']
print('Nested list')
print(list_name)                                #nested list
print('length of list')
print(len(list_name))                     #printing length of list
print(list_name)                          #printing original list
list_name.pop(1)                          #delete value at index 1
print('after deleting value at index 1')
print(list_name)                            #printing list after pop
list_items = ['raj','krish']
print('Concatenation of 2 list')
print(list_name+list_items)                          #Concatenation of 2 list
print('Membership')
if "Tirth" in list_name:
  print("Yes, 'Tirth' is member the list") # printing the membership of the list
print('Iteration')                              #Iteration
i = 0
while i < len(list_name):
  print(list_name[i])
  i = i + 1
print('Indexing the list items')           #indexing
print(list_name[3])
print('Slicing')                                #Slicing
print(list_name[0:3])
```

Output A:



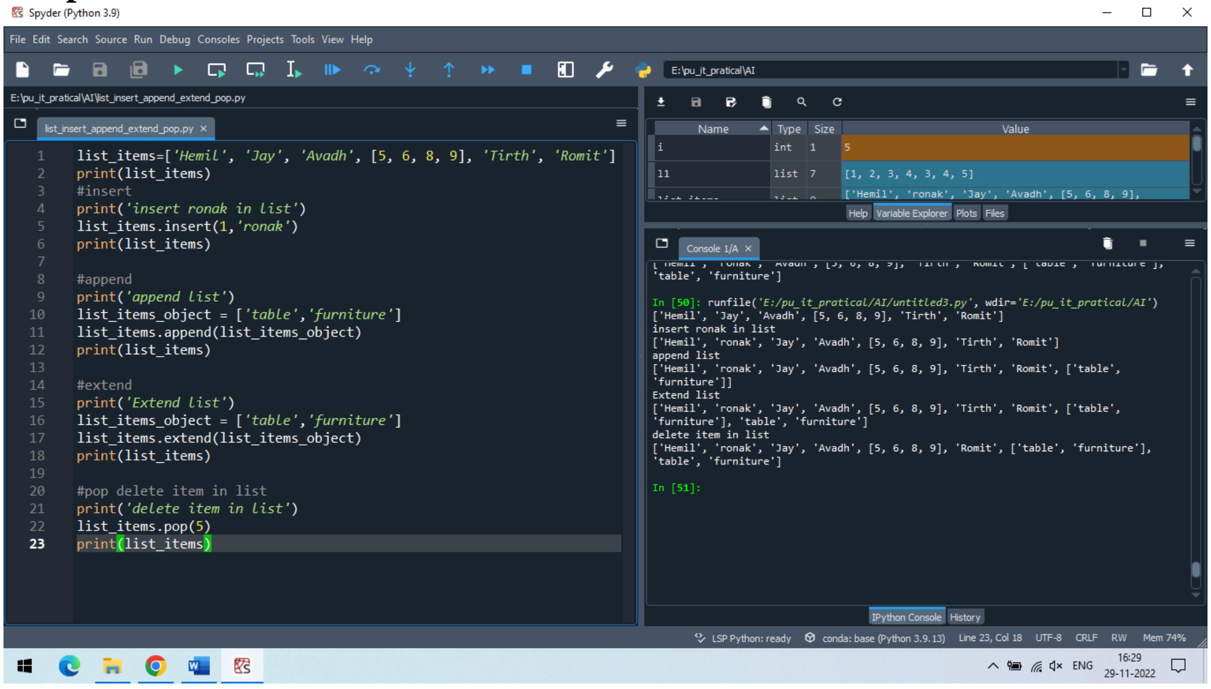
```
Nested list
['Hemil', 'Jay', 'Avadh', [5, 6, 8, 9], 'Tirth', 'Romit']
length of list
6
['Hemil', 'Jay', 'Avadh', [5, 6, 8, 9], 'Tirth', 'Romit']
after deleting value at index 1
['Hemil', 'Avadh', [5, 6, 8, 9], 'Tirth', 'Romit']
Concatenation of 2 list
['Hemil', 'Avadh', [5, 6, 8, 9], 'Tirth', 'Romit', 'raj', 'krish']
Membership
Yes, 'Tirth' is member the list
Iteration
Hemil
Avadh
[5, 6, 8, 9]
Tirth
Romit
Indexing the list items
Tirth
Slicing
['Hemil', 'Avadh', [5, 6, 8, 9]]

In [37]:
```

B) Write a python program to implement List methods (Add, Append, Extend & Delete).

Code:

```
list_items=['Hemil', 'Jay', 'Avadh', [5, 6, 8, 9], 'Tirth', 'Romit']
print(list_items)
#insert
print('insert ronak in list')
list_items.insert(1,'ronak')
print(list_items)
#append
print('append list')
list_items_object = ['table','furniture']
list_items.append(list_items_object)
print(list_items)
#extend
print('Extend list')
list_items_object = ['table','furniture']
list_items.extend(list_items_object)
print(list_items)
#pop delete item in list
print('delete item in list')
list_items.pop(5)
print(list_items)
```

Output B:



E:\pu_it_pratical\AI\list_insert_append_extend_pop.py

```
list_items=['Hemil', 'Jay', 'Avadh', [5, 6, 8, 9], 'Tirth', 'Romit']
print(list_items)
#insert
print('insert ronak in list')
list_items.insert(1,'ronak')
print(list_items)

#append
print('append list')
list_items_object = ['table','furniture']
list_items.append(list_items_object)
print(list_items)

#extend
print('Extend list')
list_items_object = ['table','furniture']
list_items.extend(list_items_object)
print(list_items)

#pop delete item in list
print('delete item in list')
list_items.pop(5)
print(list_items)
```

Name	Type	Size	Value
i	int	1	5
l1	list	7	[1, 2, 3, 4, 3, 4, 5]
[illegible]	[illegible]	[illegible]	['Hemil', 'ronak', 'Jay', 'Avadh', [5, 6, 8, 9],

```
'table', 'furniture']

In [50]: runfile('E:/pu_it_pratical/AI/untitled3.py', wdir='E:/pu_it_pratical/AI')
['Hemil', 'Jay', 'Avadh', [5, 6, 8, 9], 'Tirth', 'Romit']
insert ronak in list
['Hemil', 'ronak', 'Jay', 'Avadh', [5, 6, 8, 9], 'Tirth', 'Romit']
append list
['Hemil', 'ronak', 'Jay', 'Avadh', [5, 6, 8, 9], 'Tirth', 'Romit', ['table',
'furniture']]
Extend list
['Hemil', 'ronak', 'Jay', 'Avadh', [5, 6, 8, 9], 'Tirth', 'Romit', ['table',
'furniture'], 'table', 'furniture']
delete item in list
['Hemil', 'ronak', 'Jay', 'Avadh', [5, 6, 8, 9], 'Romit', ['table', 'furniture'],
'table', 'furniture']

In [51]:
```

Spyder (Python 3.9)
File Edit Search Source Run Debug Consoles Projects Tools View Help
E:\pu_it_pratical\AI
list_insert_append_extend_pop.py
Help Variable Explorer Plots Files
Console 1/A
IPython Console History
LSP Python: ready conda: base (Python 3.9.13) Line 23, Col 18 UTF-8 CRLF RW Mem 74%
ENG 16:29 29-11-2022

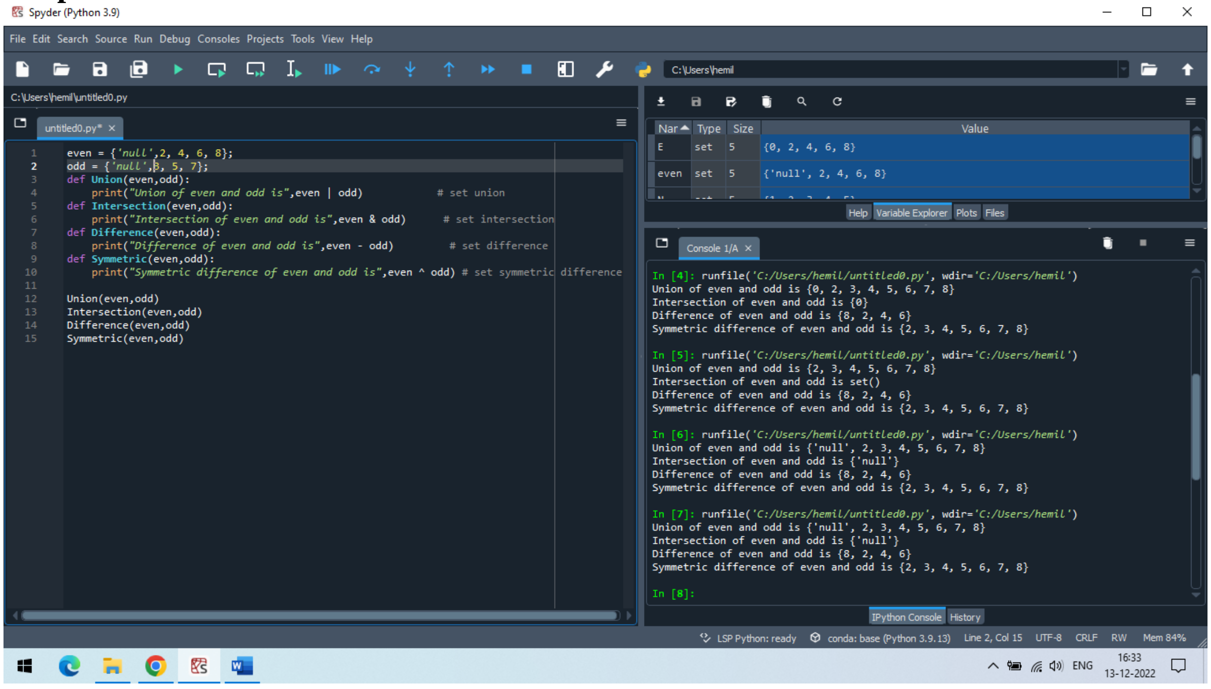
PRACTICAL-5

AIM: a) Write a python program to Illustrate Different Set Operations? b) Write a python program to generate Calendar for the given month and year? c) Write a python program to implement Simple Calculator program?

Code A :

```
even = {'null',2, 4, 6, 8};
odd = {'null',3, 5, 7};
def Union(even,odd):
   print("Union of even and odd is",even | odd)           # set union
def Intersection(even,odd):
   print("Intersection of even and odd is",even & odd)      # set intersection
def Difference(even,odd):
   print("Difference of even and odd is",even - odd)        # set difference
def Symmetric(even,odd):
   print("Symmetric difference of even and odd is",even ^ odd) # set symmetric difference

Union(even,odd)
Intersection(even,odd)
Difference(even,odd)
Symmetric(even,odd)
```

Output:



```
even = {'null',2, 4, 6, 8};
odd = {'null',3, 5, 7};
def Union(even,odd):
    print("Union of even and odd is",even | odd)           # set union
def Intersection(even,odd):
    print("Intersection of even and odd is",even & odd)      # set intersection
def Difference(even,odd):
    print("Difference of even and odd is",even - odd)         # set difference
def Symmetric(even,odd):
    print("Symmetric difference of even and odd is",even ^ odd) # set symmetric difference

Union(even,odd)
Intersection(even,odd)
Difference(even,odd)
Symmetric(even,odd)
```

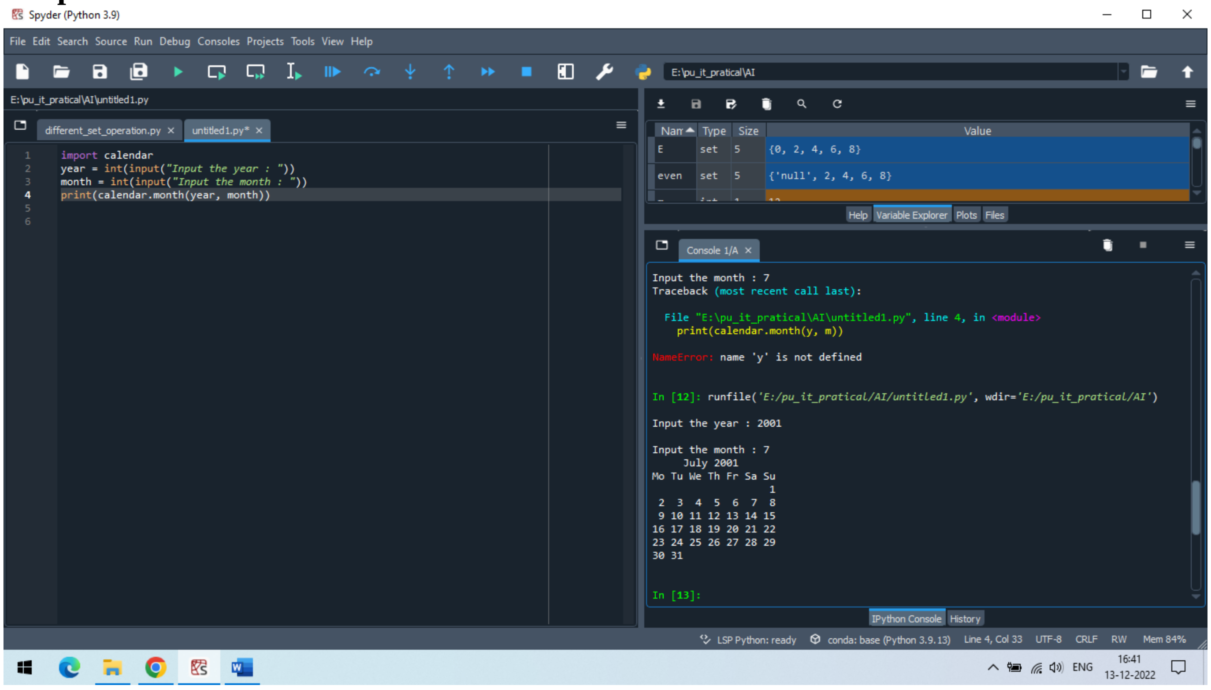
```
In [4]: runfile('C:/Users/hemil/untitled0.py', wdir='C:/Users/hemil')
Union of even and odd is {0, 2, 3, 4, 5, 6, 7, 8}
Intersection of even and odd is {0}
Difference of even and odd is {8, 2, 4, 6}
Symmetric difference of even and odd is {2, 3, 4, 5, 6, 7, 8}

In [5]: runfile('C:/Users/hemil/untitled0.py', wdir='C:/Users/hemil')
Union of even and odd is {2, 3, 4, 5, 6, 7, 8}
Intersection of even and odd is set()
Difference of even and odd is {8, 2, 4, 6}
Symmetric difference of even and odd is {2, 3, 4, 5, 6, 7, 8}

In [6]: runfile('C:/Users/hemil/untitled0.py', wdir='C:/Users/hemil')
Union of even and odd is {'null', 2, 3, 4, 5, 6, 7, 8}
Intersection of even and odd is {'null'}
Difference of even and odd is {8, 2, 4, 6}
Symmetric difference of even and odd is {2, 3, 4, 5, 6, 7, 8}

In [7]: runfile('C:/Users/hemil/untitled0.py', wdir='C:/Users/hemil')
Union of even and odd is {'null', 2, 3, 4, 5, 6, 7, 8}
Intersection of even and odd is {'null'}
Difference of even and odd is {8, 2, 4, 6}
Symmetric difference of even and odd is {2, 3, 4, 5, 6, 7, 8}

In [8]:
```

Name	Type	Size	Value
E	set	5	{0, 2, 4, 6, 8}
even	set	5	{'null', 2, 4, 6, 8}

Code B :

```
import calendar
year = int(input("Input the year : "))
month = int(input("Input the month : "))
print(calendar.month(year, month))
```

Output:


Spyder (Python 3.9)
File Edit Search Source Run Debug Consoles Projects Tools View Help
E:\pu_it_pratical\AI
E:\pu_it_pratical\AI\untitled1.py
different_set_operation.py
untitled1.py*
Help Variable Explorer Plots Files
Console 1/A
IPython Console History

```
import calendar
year = int(input("Input the year : "))
month = int(input("Input the month : "))
print(calendar.month(year, month))
```

Nam	Type	Size	Value
E	set	5	{0, 2, 4, 6, 8}
even	set	5	{'null', 2, 4, 6, 8}

```
Input the month : 7
Traceback (most recent call last):

  File "E:\pu_it_pratical\AI\untitled1.py", line 4, in <module>
    print(calendar.month(y, m))

NameError: name 'y' is not defined


In [12]: runfile('E:/pu_it_pratical/AI/untitled1.py', wdir='E:/pu_it_pratical/AI')

Input the year : 2001

Input the month : 7
     July 2001
Mo Tu We Th Fr Sa Su
                   1
 2  3  4  5  6  7  8
 9 10 11 12 13 14 15
16 17 18 19 20 21 22
23 24 25 26 27 28 29
30 31


In [13]:
```

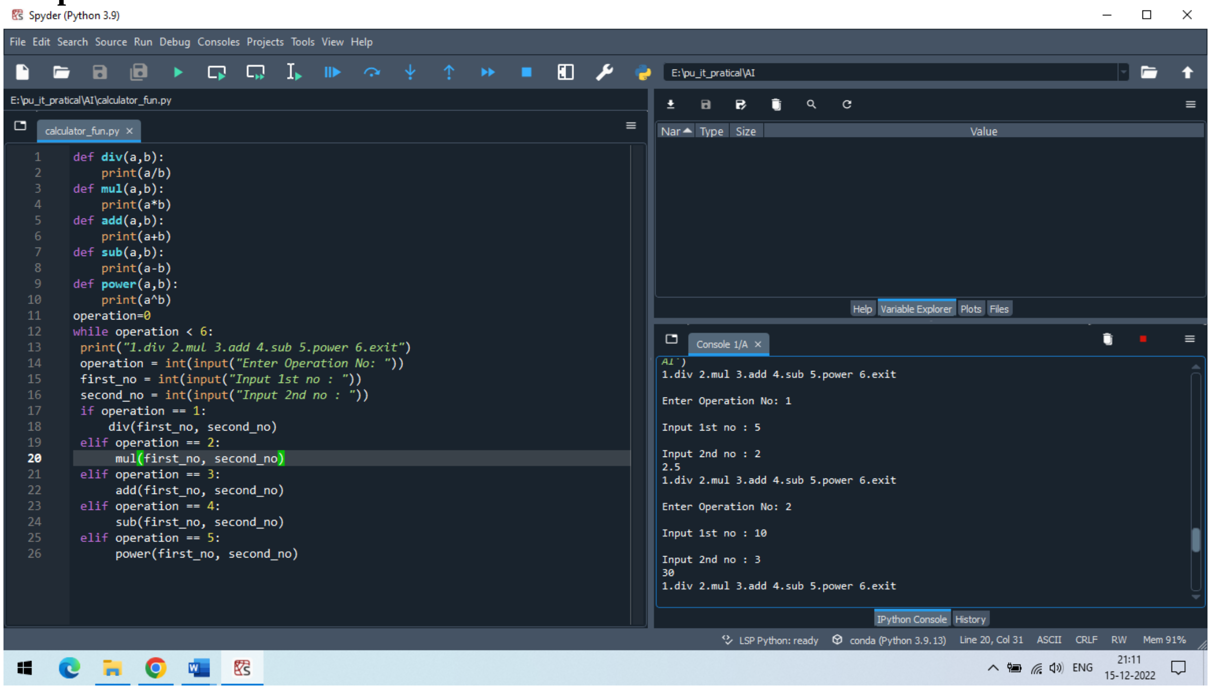
LSP Python: ready conda: base (Python 3.9.13) Line 4, Col 33 UTF-8 CRLF RW Mem 84%
16:41 13-12-2022

Code C :

```
def div(a,b):
    print(a/b)
def mul(a,b):
    print(a*b)
def add(a,b):
    print(a+b)
def sub(a,b):
    print(a-b)
def power(a,b):
    print(a^b)
operation=0
while operation < 6:
 print("1.div 2.mul 3.add 4.sub 5.power 6.exit")
 operation = int(input("Enter Operation No: "))
 first_no = int(input("Input 1st no : "))
 second_no = int(input("Input 2nd no : "))
 if operation == 1:
     div(first_no, second_no)
 elif operation == 2:
      mul(first_no, second_no)
```

```
elif operation == 3:
    add(first_no, second_no)
elif operation == 4:
    sub(first_no, second_no)
elif operation == 5:
    power(first_no, second_no)
```

Output:



Spyder (Python 3.9)
File Edit Search Source Run Debug Consoles Projects Tools View Help
E:\pu_it_pratical\AI
E:\pu_it_pratical\AI\calculator_fun.py
calculator_fun.py
Name Type Size Value
Help Variable Explorer Plots Files
Console 1/A
IPython Console History
LSP Python: ready conda (Python 3.9.13) Line 20, Col 31 ASCII CRLF RW Mem 91%
ENG 21:11 15-12-2022

```
def div(a,b):
    print(a/b)
def mul(a,b):
    print(a*b)
def add(a,b):
    print(a+b)
def sub(a,b):
    print(a-b)
def power(a,b):
    print(a^b)
operation=0
while operation < 6:
 print("1.div 2.mul 3.add 4.sub 5.power 6.exit")
 operation = int(input("Enter Operation No: "))
 first_no = int(input("Input 1st no : "))
 second_no = int(input("Input 2nd no : "))
 if operation == 1:
     div(first_no, second_no)
 elif operation == 2:
      mul(first_no, second_no)
 elif operation == 3:
      add(first_no, second_no)
 elif operation == 4:
      sub(first_no, second_no)
 elif operation == 5:
      power(first_no, second_no)
```

```
AI')
1.div 2.mul 3.add 4.sub 5.power 6.exit

Enter Operation No: 1

Input 1st no : 5

Input 2nd no : 2
2.5
1.div 2.mul 3.add 4.sub 5.power 6.exit

Enter Operation No: 2

Input 1st no : 10

Input 2nd no : 3
30
1.div 2.mul 3.add 4.sub 5.power 6.exit
```

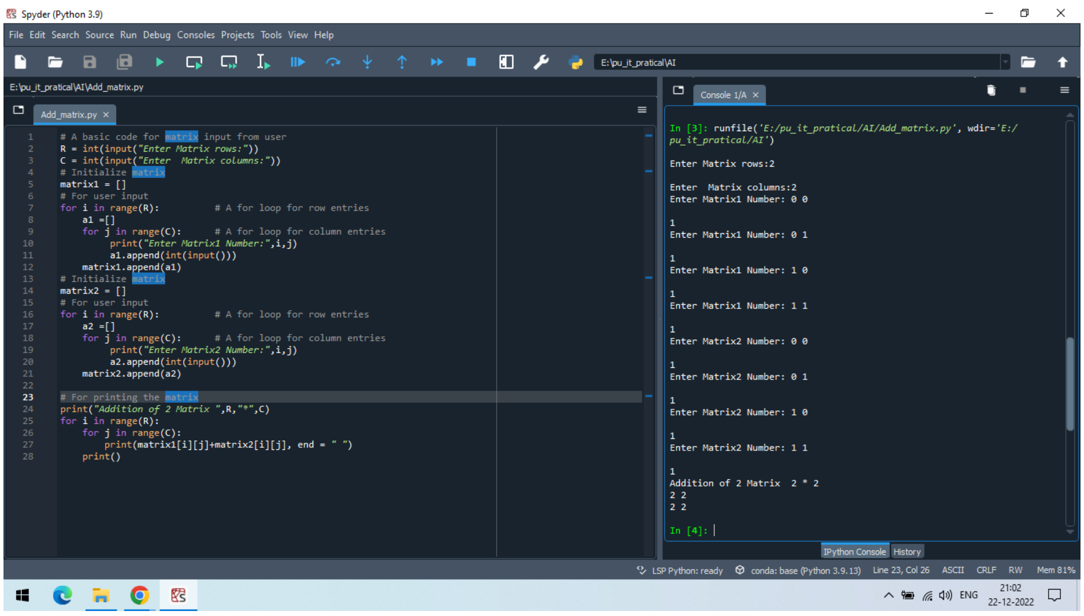
PRACTICAL-6

AIM: a)Write a python program to Add Two Matrices.
b) Write a python program to Transpose a Matrix.

Code a)

```
R = int(input("Enter Matrix rows:"))
C = int(input("Enter  Matrix columns:"))
matrix1 = []                              # Initialize matrix
# For user input
for i in range(R):          # A for loop for row entries
    a1 =[]
    for j in range(C):      # A for loop for column entries
        print("Enter Matrix1 Number:",i,j)
        a1.append(int(input()))
    matrix1.append(a1)
matrix2 = []                     # Initialize matrix
# For user input
for i in range(R):          # A for loop for row entries
    a2 =[]
    for j in range(C):      # A for loop for column entries
        print("Enter Matrix2 Number:",i,j)
        a2.append(int(input()))
    matrix2.append(a2)
print("Addition of 2 Matrix ",R,"*",C)          # For printing the matrix
for i in range(R):
    for j in range(C):
        print(matrix1[i][j]+matrix2[i][j], end = " ")
    print()
```

Output:



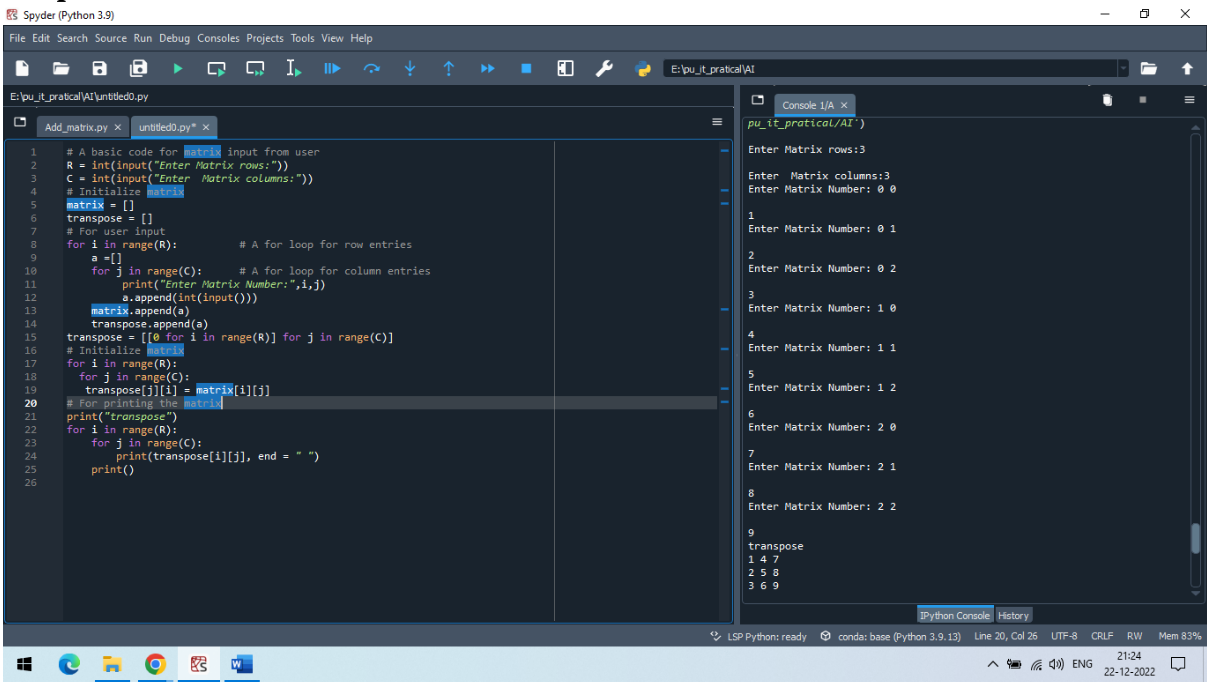
```
# A basic code for matrix input from user
R = int(input("Enter Matrix rows:"))
C = int(input("Enter  Matrix columns:"))
# Initialize matrix
matrix1 = []
# For user input
for i in range(R):          # A for loop for row entries
    a1 =[]
    for j in range(C):      # A for loop for column entries
        print("Enter Matrix1 Number:",i,j)
        a1.append(int(input()))
    matrix1.append(a1)
# Initialize matrix
matrix2 = []
# For user input
for i in range(R):          # A for loop for row entries
    a2 =[]
    for j in range(C):      # A for loop for column entries
        print("Enter Matrix2 Number:",i,j)
        a2.append(int(input()))
    matrix2.append(a2)

# For printing the matrix
print("Addition of 2 Matrix ",R,"*",C)
for i in range(R):
    for j in range(C):
        print(matrix1[i][j]+matrix2[i][j], end = " ")
    print()
```

```
In [3]: runfile('E:/pu_it_pratical/AI/Add_matrix.py', wdir='E:/
pu_it_pratical/AI')

Enter Matrix rows:2

Enter  Matrix columns:2
Enter Matrix1 Number: 0 0

1
Enter Matrix1 Number: 0 1

1
Enter Matrix1 Number: 1 0

1
Enter Matrix1 Number: 1 1

1
Enter Matrix2 Number: 0 0

1
Enter Matrix2 Number: 0 1

1
Enter Matrix2 Number: 1 0

1
Enter Matrix2 Number: 1 1

1
Addition of 2 Matrix  2 * 2
2 2
2 2

In [4]:
```

Code b)

```
R = int(input("Enter Matrix rows:"))
C = int(input("Enter  Matrix columns:"))
matrix = []                              # Initialize matrix
transpose = []
# For user input
for i in range(R):          # A for loop for row entries
    a =[]
    for j in range(C):      # A for loop for column entries
         print("Enter Matrix Number:",i,j)
         a.append(int(input()))
    matrix.append(a)
    transpose.append(a)
transpose = [[0 for i in range(R)] for j in range(C)]
# Initialize matrix
for i in range(R):
  for j in range(C):
   transpose[j][i] = matrix[i][j]
# For printing the matrix
print("transpose")
for i in range(R):
    for j in range(C):
        print(transpose[i][j], end = " ")
    print()
```

Output:



```
# A basic code for matrix input from user
R = int(input("Enter Matrix rows:"))
C = int(input("Enter  Matrix columns:"))
# Initialize matrix
matrix = []
transpose = []
# For user input
for i in range(R):          # A for loop for row entries
    a =[]
    for j in range(C):      # A for loop for column entries
         print("Enter Matrix Number:",i,j)
         a.append(int(input()))
    matrix.append(a)
    transpose.append(a)
transpose = [[0 for i in range(R)] for j in range(C)]
# Initialize matrix
for i in range(R):
  for j in range(C):
   transpose[j][i] = matrix[i][j]
# For printing the matrix
print("transpose")
for i in range(R):
    for j in range(C):
        print(transpose[i][j], end = " ")
    print()
```

```
pu_it_pratical/AI')

Enter Matrix rows:3

Enter  Matrix columns:3
Enter Matrix Number: 0 0

1
Enter Matrix Number: 0 1

2
Enter Matrix Number: 0 2

3
Enter Matrix Number: 1 0

4
Enter Matrix Number: 1 1

5
Enter Matrix Number: 1 2

6
Enter Matrix Number: 2 0

7
Enter Matrix Number: 2 1

8
Enter Matrix Number: 2 2

9
transpose
1 4 7
2 5 8
3 6 9
```

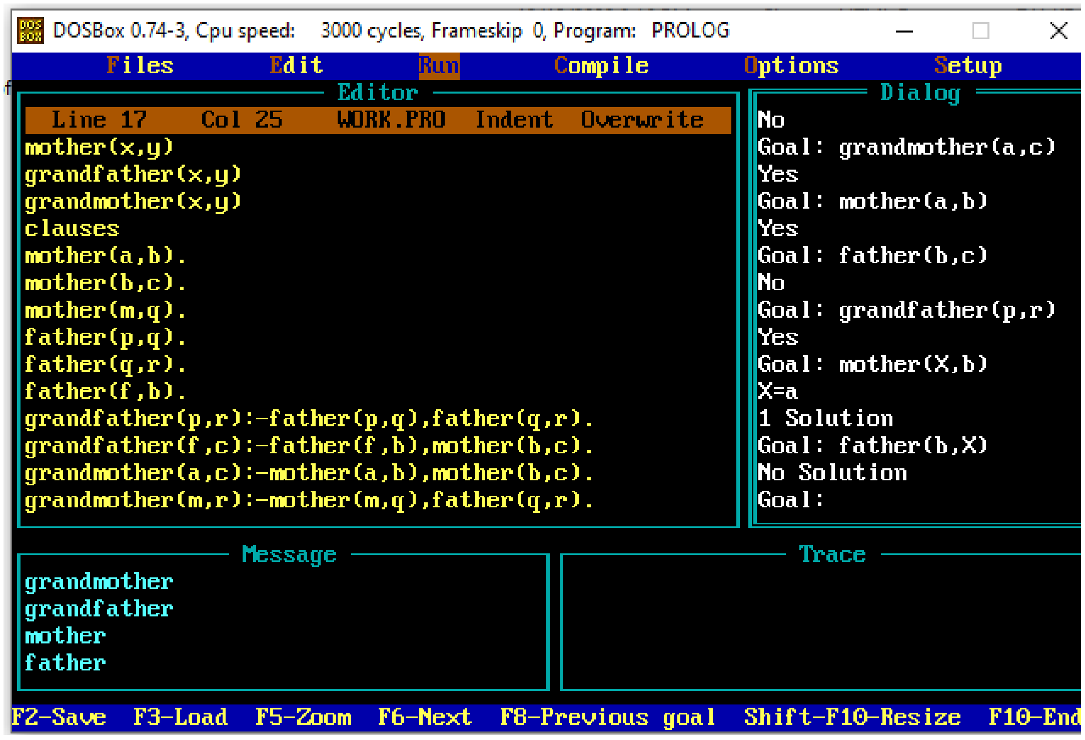
PRACTICAL-1

AIM: Write a program in prolog to implement simple facts and Queries.

Code:

```
domains
x,y = symbol
predicates
father(x,y)
mother(x,y)
grandfather(x,y)
grandmother(x,y)
clauses
mother(a,b).
mother(b,c).
mother(m,q).
father(p,q).
father(q,r).
father(f,b).
grandfather(p,r):-father(p,q),father(q,r).
grandfather(f,c):-father(f,b),mother(b,c).
grandmother(a,c):-mother(a,b),mother(b,c).
grandmother(m,r):-mother(m,q), father(q,r).
```

Output:


DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: PROLOG
Files
Edit
Run
Compile
Options
Setup
Editor
Dialog
Message
Trace
F2-Save F3-Load F5-Zoom F6-Next F8-Previous goal Shift-F10-Resize F10-End

```
Line 17    Col 25    WORK.PRO    Indent    Overwrite
mother(x,y)
grandfather(x,y)
grandmother(x,y)
clauses
mother(a,b).
mother(b,c).
mother(m,q).
father(p,q).
father(q,r).
father(f,b).
grandfather(p,r):-father(p,q),father(q,r).
grandfather(f,c):-father(f,b),mother(b,c).
grandmother(a,c):-mother(a,b),mother(b,c).
grandmother(m,r):-mother(m,q),father(q,r).
```

```
No
Goal: grandmother(a,c)
Yes
Goal: mother(a,b)
Yes
Goal: father(b,c)
No
Goal: grandfather(p,r)
Yes
Goal: mother(X,b)
X=a
1 Solution
Goal: father(b,X)
No Solution
Goal:
```

```
grandmother
grandfather
mother
father
```

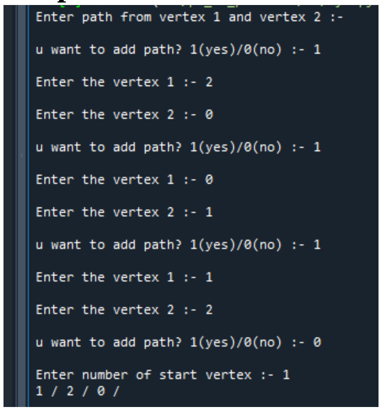
PRACTICAL-7

AIM: Write a python program to implement Breadth First Search Traversal?

Code:

```
from collections import defaultdict
class Graph_bfs:
  def __init__(self):
     self.graph_dict = defaultdict(list)
  def edge(self, From, To):
    self.graph_dict[From].append(To)
  def bfs(self, start):
     visited_node = [False] * (len(self.graph_dict))
     queue1 = []
     queue1.append(start)
     visited_node[start] = True
     while queue1:
       start = queue1.pop(0)
       print(start, end=" / ")
       for i in self.graph_dict[start]:
          if visited_node[i] == False:
             queue1.append(i)
             visited_node[i] = True
b1 = Graph_bfs()
print('Enter path from vertex 1 and vertex 2 :- ')
while(1):
  new = int(input('u want to add path? 1(yes)/0(no) :- '))
  k = bool(new)
  if(k == False):
    break
  key = int(input('Enter the vertex 1 :- '))
  value = int(input('Enter the vertex 2 :- '))
  b1.edge(key, value)
n = int(input('Enter number of start vertex :- '))
b1.bfs(n)
```

Output:



```
Enter path from vertex 1 and vertex 2 :-

u want to add path? 1(yes)/0(no) :- 1

Enter the vertex 1 :- 2

Enter the vertex 2 :- 0

u want to add path? 1(yes)/0(no) :- 1

Enter the vertex 1 :- 0

Enter the vertex 2 :- 1

u want to add path? 1(yes)/0(no) :- 1

Enter the vertex 1 :- 1

Enter the vertex 2 :- 2

u want to add path? 1(yes)/0(no) :- 0

Enter number of start vertex :- 1
1 / 2 / 0 /
```

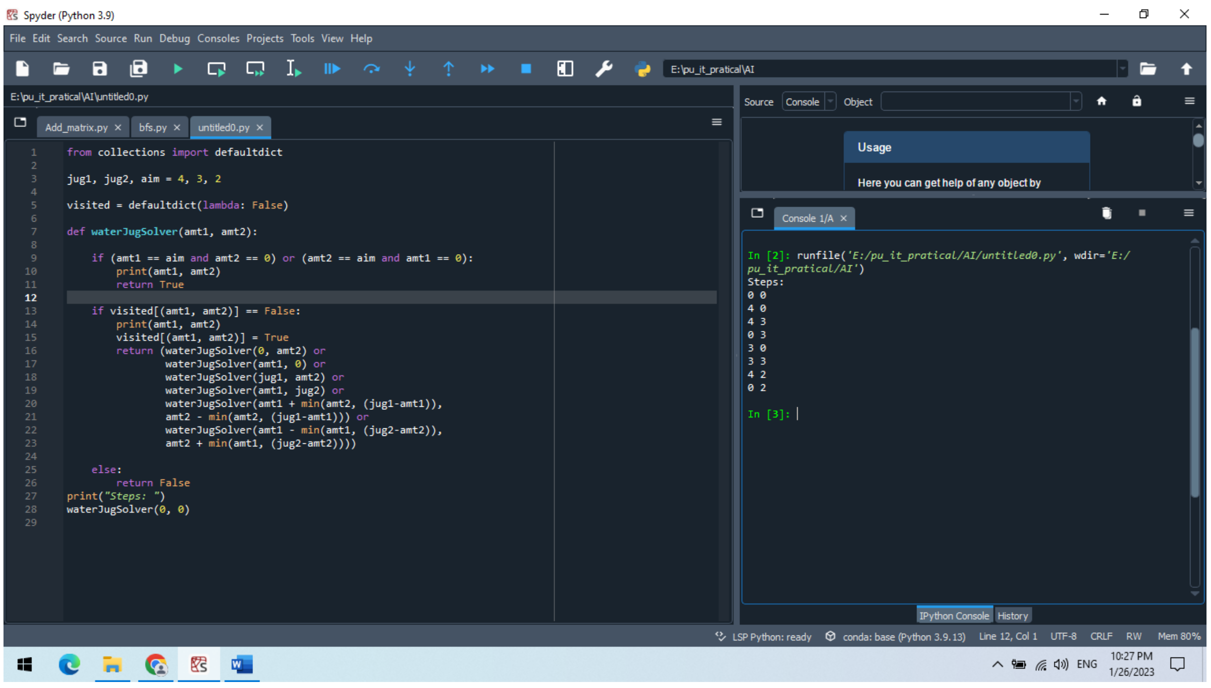
PRACTICAL-8

AIM: Write a python program to implement Water Jug Problem?

Code:

```
from collections import defaultdict
jug1, jug2, aim = 4, 3, 2
visited = defaultdict(lambda: False)
def waterJugSolver(amt1, amt2):
	if (amt1 == aim and amt2 == 0) or (amt2 == aim and amt1 == 0):
		print(amt1, amt2)
		return True
	if visited[(amt1, amt2)] == False:
		print(amt1, amt2)
		visited[(amt1, amt2)] = True
		return (waterJugSolver(0, amt2) or
				waterJugSolver(amt1, 0) or
				waterJugSolver(jug1, amt2) or
				waterJugSolver(amt1, jug2) or
				waterJugSolver(amt1 + min(amt2, (jug1-amt1)),
				amt2 - min(amt2, (jug1-amt1))) or
				waterJugSolver(amt1 - min(amt1, (jug2-amt2)),
				amt2 + min(amt1, (jug2-amt2))))
	else:
		return False
print("Steps: ")
waterJugSolver(0, 0)
```

Output:



```
In [2]: runfile('E:/pu_it_pratical/AI/untitled0.py', wdir='E:/
pu_it_pratical/AI')
Steps:
0 0
4 0
4 3
0 3
3 0
3 3
4 2
0 2

In [3]: 
```

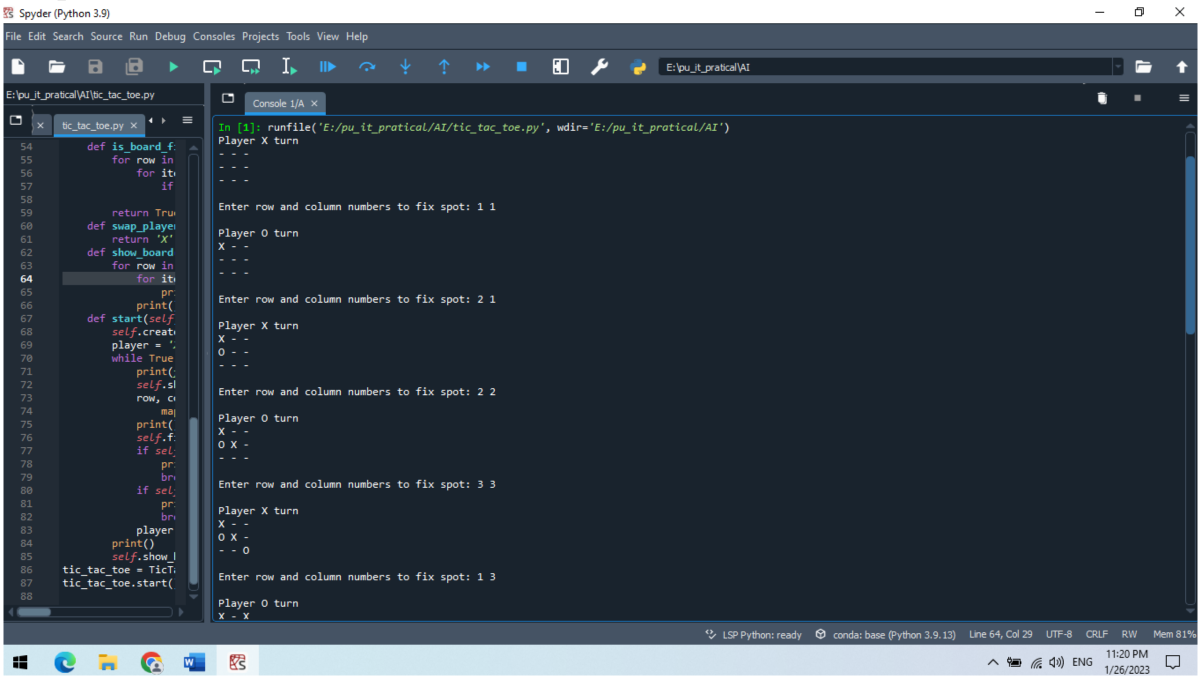
PRACTICAL-9

AIM: Write a program to implement Tic-Tac-Toe game using python.

Code:

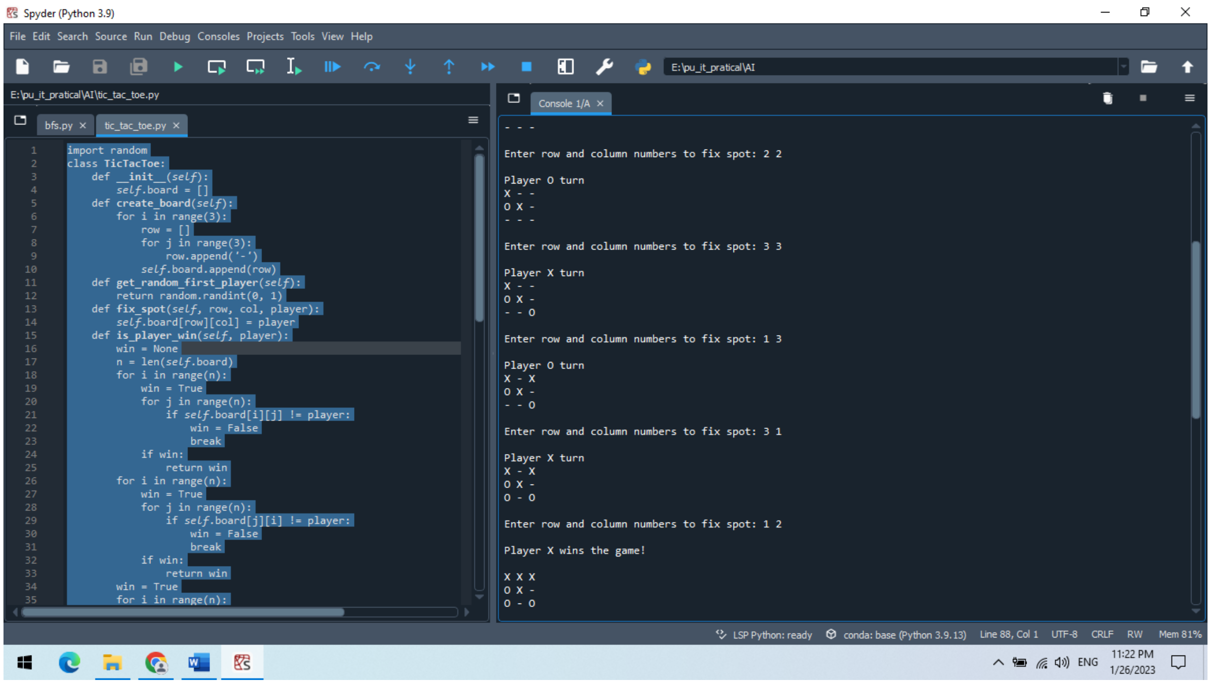
```
import random
class TicTacToe:
    def __init__(self):
        self.board = []
    def create_board(self):
        for i in range(3):
            row = []
            for j in range(3):
                row.append('-')
            self.board.append(row)
    def get_random_first_player(self):
        return random.randint(0, 1)
    def fix_spot(self, row, col, player):
        self.board[row][col] = player
    def is_player_win(self, player):
        win = None
        n = len(self.board)
        for i in range(n):
            win = True
            for j in range(n):
                if self.board[i][j] != player:
                    win = False
                    break
            if win:
                return win
        for i in range(n):
            win = True
            for j in range(n):
                if self.board[j][i] != player:
                    win = False
                    break
            if win:
                return win
        win = True
        for i in range(n):
            if self.board[i][i] != player:
                win = False
                break
        if win:
            return win
        win = True
        for i in range(n):
            if self.board[i][n - 1 - i] != player:
                win = False
                break
        if win:
            return win
        return False
        for row in self.board:
            for item in row:
                if item == '-':
                    return False
        return True
    def is_board_filled(self):
        for row in self.board:
            for item in row:
                if item == '-':
                    return False
        return True
    def swap_player_turn(self, player):
        return 'X' if player == 'O' else 'O'
    def show_board(self):
        for row in self.board:
            for item in row:
                print(item, end=" ")
            print()
    def start(self):
        self.create_board()
        player = 'X' if self.get_random_first_player() == 1 else 'O'
        while True:
            print(f"Player {player} turn")
            self.show_board()
            row, col = list(
                map(int, input("Enter row and column numbers to fix spot: ").split()))
            print()
            self.fix_spot(row - 1, col - 1, player)
            if self.is_player_win(player):
                print(f"Player {player} wins the game!")
                break
            if self.is_board_filled():
                print("Match Draw!")
                break
            player = self.swap_player_turn(player)
        print()
        self.show_board()
tic_tac_toe = TicTacToe()
tic_tac_toe.start()
```

Output:



```
    def is_board_f
        for row in
            for it
                if

        return Tru
    def swap_playe
        return 'X'
    def show_board
        for row in
            for it
                pr
            print(
    def start(self
        self.creat
        player = '
        while True
            print(
            self.sl
            row, c
                ma
            print(
            self.f
            if sel
                pr
                br
            if sel
                pr
                br
            player
        print()
        self.show_
tic_tac_toe = TicT
tic_tac_toe.start(
```

```
In [1]: runfile('E:/pu_it_pratical/AI/tic_tac_toe.py', wdir='E:/pu_it_pratical/AI')
Player X turn
- - -
- - -
- - -

Enter row and column numbers to fix spot: 1 1

Player O turn
X - -
- - -
- - -

Enter row and column numbers to fix spot: 2 1

Player X turn
X - -
O - -
- - -

Enter row and column numbers to fix spot: 2 2

Player O turn
X - -
O X -
- - -

Enter row and column numbers to fix spot: 3 3

Player X turn
X - -
O X -
- - O

Enter row and column numbers to fix spot: 1 3

Player O turn
X - X
```



```
import random
class TicTacToe:
    def __init__(self):
        self.board = []
    def create_board(self):
        for i in range(3):
            row = []
            for j in range(3):
                row.append('-')
            self.board.append(row)
    def get_random_first_player(self):
        return random.randint(0, 1)
    def fix_spot(self, row, col, player):
        self.board[row][col] = player
    def is_player_win(self, player):
        win = None
        n = len(self.board)
        for i in range(n):
            win = True
            for j in range(n):
                if self.board[i][j] != player:
                    win = False
                    break
            if win:
                return win
        for i in range(n):
            win = True
            for j in range(n):
                if self.board[j][i] != player:
                    win = False
                    break
            if win:
                return win
        win = True
        for i in range(n):
```

```
- - -

Enter row and column numbers to fix spot: 2 2

Player O turn
X - -
O X -
- - -

Enter row and column numbers to fix spot: 3 3

Player X turn
X - -
O X -
- - O

Enter row and column numbers to fix spot: 1 3

Player O turn
X - X
O X -
- - O

Enter row and column numbers to fix spot: 3 1

Player X turn
X - X
O X -
O - O

Enter row and column numbers to fix spot: 1 2

Player X wins the game!

X X X
O X -
O - O
```

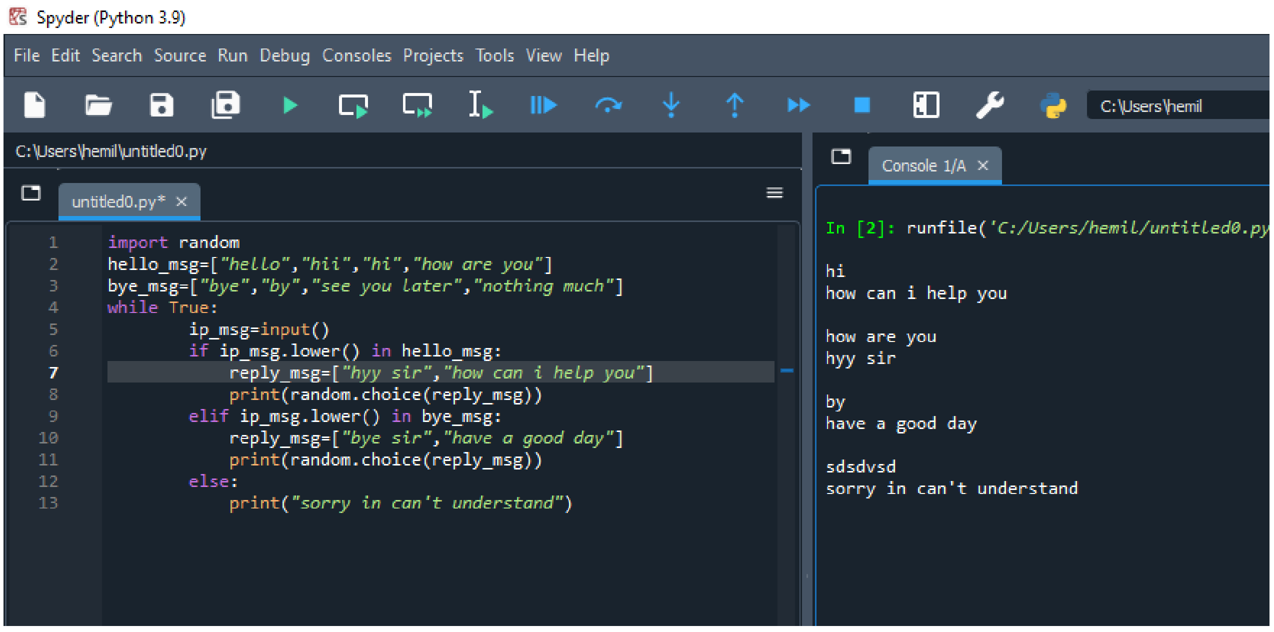
PRACTICAL-3

AIM: Write a python program to implement simple Chatbot.

Code a:

```
import random
hello_msg=["hello","hii","hi","how are you"]
bye_msg=["bye","by","see you later","nothing much"]
while True:
        ip_msg=input()
        if ip_msg.lower() in hello_msg:
            reply_msg=["hyy sir","how can i help you"]
            print(random.choice(reply_msg))
        elif ip_msg.lower() in bye_msg:
            reply_msg=["bye sir","have a good day"]
            print(random.choice(reply_msg))
        else:
            print("sorry in can't understand")from collections import defaultdict
```

Output:


Spyder (Python 3.9)
File Edit Search Source Run Debug Consoles Projects Tools View Help
C:\Users\hemil
C:\Users\hemil\untitled0.py
untitled0.py*
Console 1/A

```
import random
hello_msg=["hello","hii","hi","how are you"]
bye_msg=["bye","by","see you later","nothing much"]
while True:
        ip_msg=input()
        if ip_msg.lower() in hello_msg:
            reply_msg=["hyy sir","how can i help you"]
            print(random.choice(reply_msg))
        elif ip_msg.lower() in bye_msg:
            reply_msg=["bye sir","have a good day"]
            print(random.choice(reply_msg))
        else:
            print("sorry in can't understand")
```

```
In [2]: runfile('C:/Users/hemil/untitled0.py

hi
how can i help you

how are you
hyy sir

by
have a good day

sdsdvsd
sorry in can't understand
```

PRACTICAL-10

AIM: a)Write a python program to remove stop words for a given passage from a text file using NLTK?
b)Write a python program to implement stemming for a given sentence using NLTK?
c)Write a python program to POS (Parts of Speech) tagging for the give sentence using NLTK?

Code a:

```
from collections import defaultdict
```

Output:

Code b:

```
from collections import defaultdict
```

Output:

Code c:

```
from collections import defaultdict
```

Output:

PRACTICAL-11

AIM:a) Write a python program to implement Lemmatization using NLTK?
B)Write a python program to for Text Classification for the give sentence using NLTK?

Code a:

```
from collections import defaultdict
```

Output:

Code b:

```
from collections import defaultdict
```

Output: