CD Lab Practical 5

Name : Chetan Pardhi

Roll No. : 22

Batch : B1

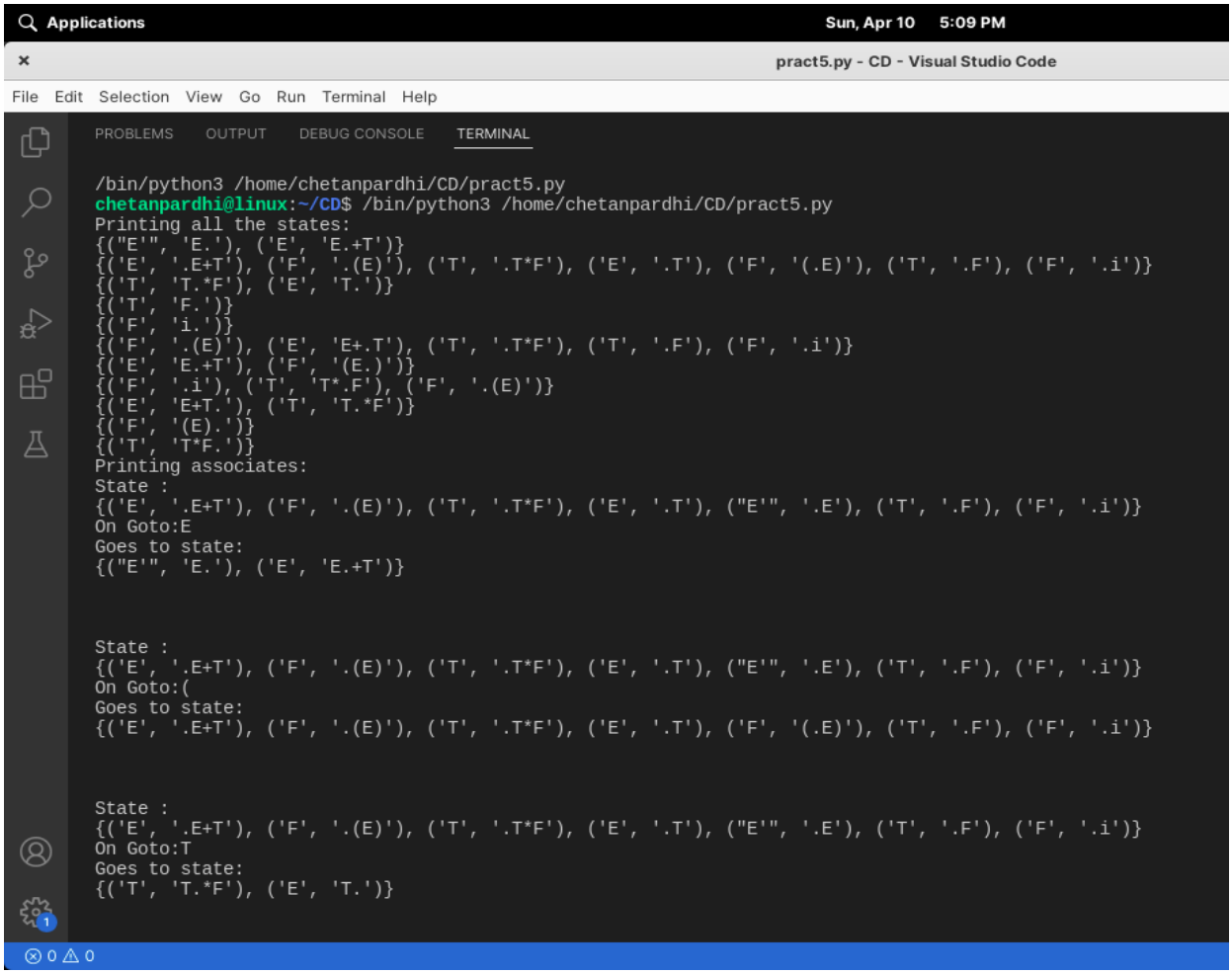
Aim : Construct the SLR parser for the given grammar

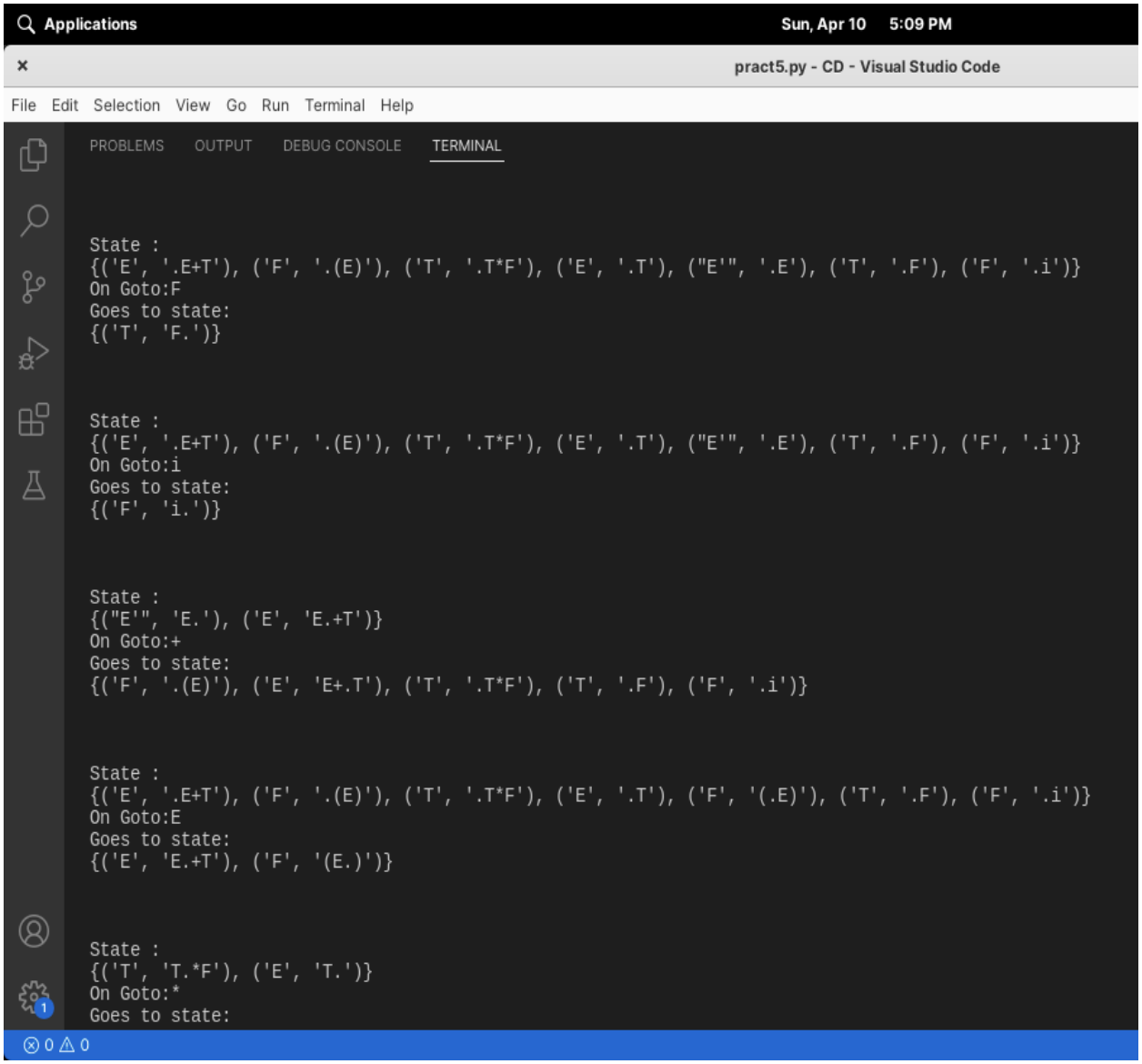
```
grammar=[('E','.E+T'),('E','.T'),('T','.T*F'),('T','.F'),('F','.(E)'),('F'
,'.i')]


def getAugmentedGrammar(grammar):
 return {("E'",'.E')}


def closure(state):
 b=len(state)
 newState=set()
 for i in state:
   newState.add(i)
   if(i[1][-1]=='.'):
     continue
   for j in range(len(i[1])):
```

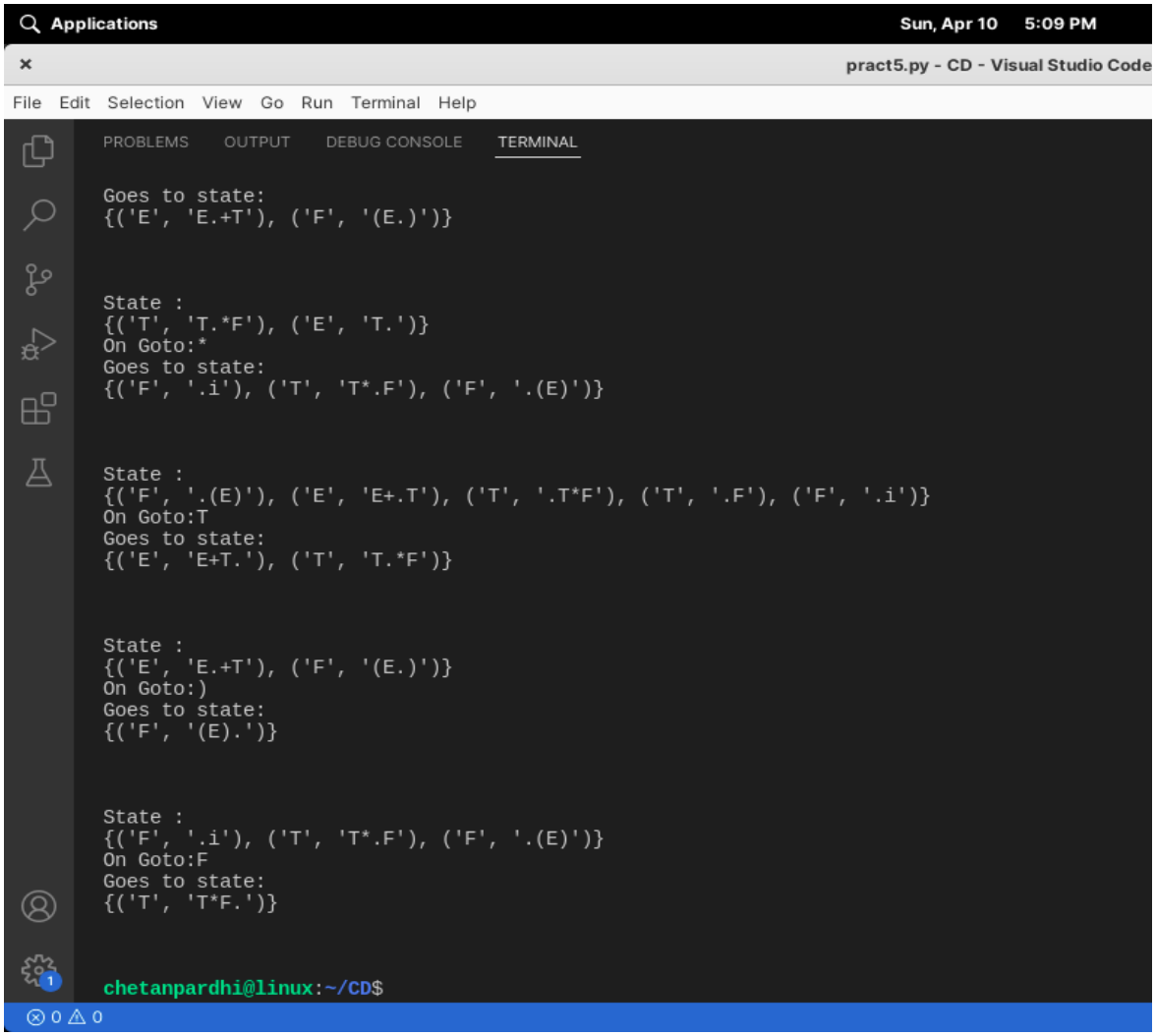
```
    if(i[1][j]=='.' and i[1][j+1].isupper()):
      for k in grammar:
        if(k[0]==i[1][j+1]):
          newState.add(k)
 if b==len(newState):
   return newState
 return closure(newState)


def goto(state,a):
 newState=set()
 for i in state:
   ind=i[1].index('.')
   if(ind<len(i[1])-1 and i[1][ind+1]==a):
     s=i[1].replace('.',"")
     s1=""
     for c in s:
       s1+=c
       if(c==a):
         s1+='.'
     newState.add((i[0],s1))
 cl=closure(newState)
 return cl
queue=[closure(getAugmentedGrammar(grammar))]
allStates=[]
assoStates=[]
while(len(queue)>0):
 c=queue.pop(0)
```

```
 for i in c:
   if(i[1][-1]=='.'):
     continue
   p=i[1][i[1].index('.')+1]
   s1=goto(c,i[1][i[1].index('.')+1])
   if(s1 not in allStates):
     queue.append(s1)
     allStates.append(s1)
     assoStates.append([c,p,s1])
print("Printing all the states:")
for i in allStates:
 print(i)
print("Printing associates:")
for i in assoStates:
 a=i[0]
 b=i[1]
 c=i[2]
 print("State :\n"+str(a)+"\nOn Goto:"+str(b)+"\nGoes to state:\n"+str(c)
)
 print("\n\n")
```

Screen-Shot:


Applications
Sun, Apr 10 5:09 PM
pract5.py - CD - Visual Studio Code
File Edit Selection View Go Run Terminal Help
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
/bin/python3 /home/chetanpardhi/CD/pract5.py
chetanpardhi@linux:~/CD$ /bin/python3 /home/chetanpardhi/CD/pract5.py
Printing all the states:
{("E'", 'E.'), ('E', 'E.+T')}
{('E', '.E+T'), ('F', '.(E)'), ('T', '.T*F'), ('E', '.T'), ('F', '(.E)'), ('T', '.F'), ('F', '.i')}
{('T', 'T.*F'), ('E', 'T.')}
{('T', 'F.')}
{('F', 'i.')}
{('F', '.(E)'), ('E', 'E+.T'), ('T', '.T*F'), ('T', '.F'), ('F', '.i')}
{('E', 'E.+T'), ('F', '(E.)')}
{('F', '.i'), ('T', 'T*.F'), ('F', '.(E)')}
{('E', 'E+T.'), ('T', 'T.*F')}
{('F', '(E).')}
{('T', 'T*F.')}
Printing associates:
State :
{('E', '.E+T'), ('F', '.(E)'), ('T', '.T*F'), ('E', '.T'), ("E'", '.E'), ('T', '.F'), ('F', '.i')}
On Goto:E
Goes to state:
{("E'", 'E.'), ('E', 'E.+T')}
State :
{('E', '.E+T'), ('F', '.(E)'), ('T', '.T*F'), ('E', '.T'), ("E'", '.E'), ('T', '.F'), ('F', '.i')}
On Goto:(
Goes to state:
{('E', '.E+T'), ('F', '.(E)'), ('T', '.T*F'), ('E', '.T'), ('F', '(.E)'), ('T', '.F'), ('F', '.i')}
State :
{('E', '.E+T'), ('F', '.(E)'), ('T', '.T*F'), ('E', '.T'), ("E'", '.E'), ('T', '.F'), ('F', '.i')}
On Goto:T
Goes to state:
{('T', 'T.*F'), ('E', 'T.')}
⊗ 0 ⚠ 0


Applications
Sun, Apr 10 5:09 PM
pract5.py - CD - Visual Studio Code
File Edit Selection View Go Run Terminal Help
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
State :
{('E', '.E+T'), ('F', '.(E)'), ('T', '.T*F'), ('E', '.T'), ("E'", '.E'), ('T', '.F'), ('F', '.i')}
On Goto:F
Goes to state:
{('T', 'F.')}
State :
{('E', '.E+T'), ('F', '.(E)'), ('T', '.T*F'), ('E', '.T'), ("E'", '.E'), ('T', '.F'), ('F', '.i')}
On Goto:i
Goes to state:
{('F', 'i.')}
State :
{("E'", 'E.'), ('E', 'E.+T')}
On Goto:+
Goes to state:
{('F', '.(E)'), ('E', 'E+.T'), ('T', '.T*F'), ('T', '.F'), ('F', '.i')}
State :
{('E', '.E+T'), ('F', '.(E)'), ('T', '.T*F'), ('E', '.T'), ('F', '(.E)'), ('T', '.F'), ('F', '.i')}
On Goto:E
Goes to state:
{('E', 'E.+T'), ('F', '(E.)')}
State :
{('T', 'T.*F'), ('E', 'T.')}
On Goto:*
Goes to state:
0 0


Applications
Sun, Apr 10 5:09 PM
pract5.py - CD - Visual Studio Code
File Edit Selection View Go Run Terminal Help
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
Goes to state:
{('E', 'E.+T'), ('F', '(E.)')}
State :
{('T', 'T.*F'), ('E', 'T.')}
On Goto:*
Goes to state:
{('F', '.i'), ('T', 'T*.F'), ('F', '.(E)')}
State :
{('F', '.(E)'), ('E', 'E+.T'), ('T', '.T*F'), ('T', '.F'), ('F', '.i')}
On Goto:T
Goes to state:
{('E', 'E+T.'), ('T', 'T.*F')}
State :
{('E', 'E.+T'), ('F', '(E.)')}
On Goto:)
Goes to state:
{('F', '(E).')}
State :
{('F', '.i'), ('T', 'T*.F'), ('F', '.(E)')}
On Goto:F
Goes to state:
{('T', 'T*F.')}
chetanpardhi@linux:~/CD$
0 0