Name: Atharva Paliwal
Roll No. 6B-40

Practical No. 4

Aim: Write a program to perform string validation for LR grammar.

Program:

```
#Driver Code
Non_Terminal,Terminal,Rules,parse_table=list(),list(),dict(),dict()

# Input Non Terminal
print('Enter Non Terminals : ')
Non_Terminal=list(str(i) for i in input().split())
for x in Non_Terminal:
    Rules[x]=[]

# Input Terminal
print('Enter Terminals:')
Terminal=list(str(i) for i in input().split())+['$']

rule_index=[]
# Input Production Rules
print('Enter rules in format A : BC|D and 0 for null : ')
for _ in range(len(Non_Terminal)):
    rule=input().split(':')
    for i in rule[1].split('|'):
        Rules[rule[0].strip()].append(i.strip())
        rule_index.append((rule[0],i))

print(rule_index)

print('Enter Parsing Table state,NT/T:action/Goto (-1 to exit): ')
while(True):
    s=input()
    if s=='-1':
        break
    s=s.split(':')
    s1=s[0].split(',')
    parse_table[(s1[0],s1[1])]=s[1]


for i,j in parse_table.items():
    print(i,' -> ',j)
```

```
stack='00'
string=input('Enter string to validate : ')
buffer=string+'$'
result='0'
flag=True
print('Stack\tBuffer\tAction')
while result!='Accept':
    print(stack,end='\t')
    print(buffer,end='\t')
    try:
        result=parse_table[(stack[-2]+stack[-1],buffer[0])]
    except:
        flag=False
        break
    print(result)
    if result=='Accept':
        print('String Accepted')
    elif result[0]=='S':
        stack=stack+buffer[0]+'0'+result[1]
        buffer=buffer[1:]
    elif result[0]=='R':
        rule=rule_index[int(result[1:])-1]
        l=3*len(rule[1])
        stack=stack[:len(stack)-l]
        stack=stack+rule[0]
        stack=stack+parse_table[(stack[-3]+stack[-2],stack[-1])]
if flag==False:
    print('String is not Accepted.')
```

Input & Output:

```
PS C:\Users\ACER> & C:/Users/ACER/AppData/Local/Programs/Python/Python39/pyth
on.exe c:/Users/ACER/Desktop/Courses/College/Compiler-Design/Lab/Expt4/prac4.
py
Enter Non Terminals :
E T F
Enter Terminals:
+ * ( ) i
Enter rules in format A : BC|D and 0 for null :
E:E+T|T
T:T*F|F
F:(E)|i
[('E', 'E+T'), ('E', 'T'), ('T', 'T*F'), ('T', 'F'), ('F', '(E)'), ('F', 'i')
]
Enter Parsing Table state,NT/T:action/Goto (-1 to exit):
00,i:S5
00,(:S4
00,E:01
00,T:02
00,F:03
01,+:S6
01,$:Accept
02,+:R2
02,*:S7
02,):R2
02,$:R2
03,+:R4
03,*:R4
03,):R4
03,$:R4
04,i:S5
04,(:S4
04,E:08
04,T:02
04,F:03
05,+:R6
05,*:R6
05,):R6
```

```
05,*:R6
05,):R6
05,$:R6
06,i:S5
06,(:S4
06,T:09
06,F:03
07,i:S5
07,(:S4
07,F:10
08,+:S6
08,):S11
09,+:R1
09,*:R7
09,):R1
09,$:R1
10,+:R3
10,*:R3
10,):R3
10,$:R4
11,+:R5
11,*:R5
11,):R5
11,$:R5
-1
('00', 'i')  ->  S5
('00', '(')  ->  S4
('00', 'E')  ->  01
('00', 'T')  ->  02
('00', 'F')  ->  03
('01', '+')  ->  S6
('01', '$')  ->  Accept
('02', '+')  ->  R2
('02', '*')  ->  S7
('02', ')')  ->  R2
('02', '$')  ->  R2
('03', '+')  ->  R4
```

```
('03', '*')  ->  R4
('03', ')')  ->  R4
('03', '$')  ->  R4
('04', 'i')  ->  S5
('04', '(')  ->  S4
('04', 'E')  ->  08
('04', 'T')  ->  02
('04', 'F')  ->  03
('05', '+')  ->  R6
('05', '*')  ->  R6
('05', ')')  ->  R6
('05', '$')  ->  R6
('06', 'i')  ->  S5
('06', '(')  ->  S4
('06', 'T')  ->  09
('06', 'F')  ->  03
('07', 'i')  ->  S5
('07', '(')  ->  S4
('07', 'F')  ->  10
('08', '+')  ->  S6
('08', ')')  ->  S11
('09', '+')  ->  R1
('09', '*')  ->  R7
('09', ')')  ->  R1
('09', '$')  ->  R1
('10', '+')  ->  R3
('10', '*')  ->  R3
('10', ')')  ->  R3
('10', '$')  ->  R4
('11', '+')  ->  R5
('11', '*')  ->  R5
('11', ')')  ->  R5
('11', '$')  ->  R5
Enter string to validate : i*i+i
Stack   Buffer  Action
00      i*i+i$  S5
00i05   *i+i$   R6
```

```
('10', '+')  ->  R3
('10', '*')  ->  R3
('10', ')')  ->  R3
('10', '$')  ->  R4
('11', '+')  ->  R5
('11', '*')  ->  R5
('11', ')')  ->  R5
('11', '$')  ->  R5
Enter string to validate : i*i+i
Stack   Buffer  Action
00      i*i+i$  S5
00i05   *i+i$   R6
00F03   *i+i$   R4
00T02   *i+i$   S7
00T02*07        i+i$    S5
00T02*07i05     +i$     R6
00T02*07F10     +i$     R3
00T02   +i$     R2
00E01   +i$     S6
00E01+06        i$      S5
00E01+06i05     $       R6
00E01+06F03     $       R4
00E01+06T09     $       R1
00E01   $       Accept
String Accepted
PS C:\Users\ACER>
```