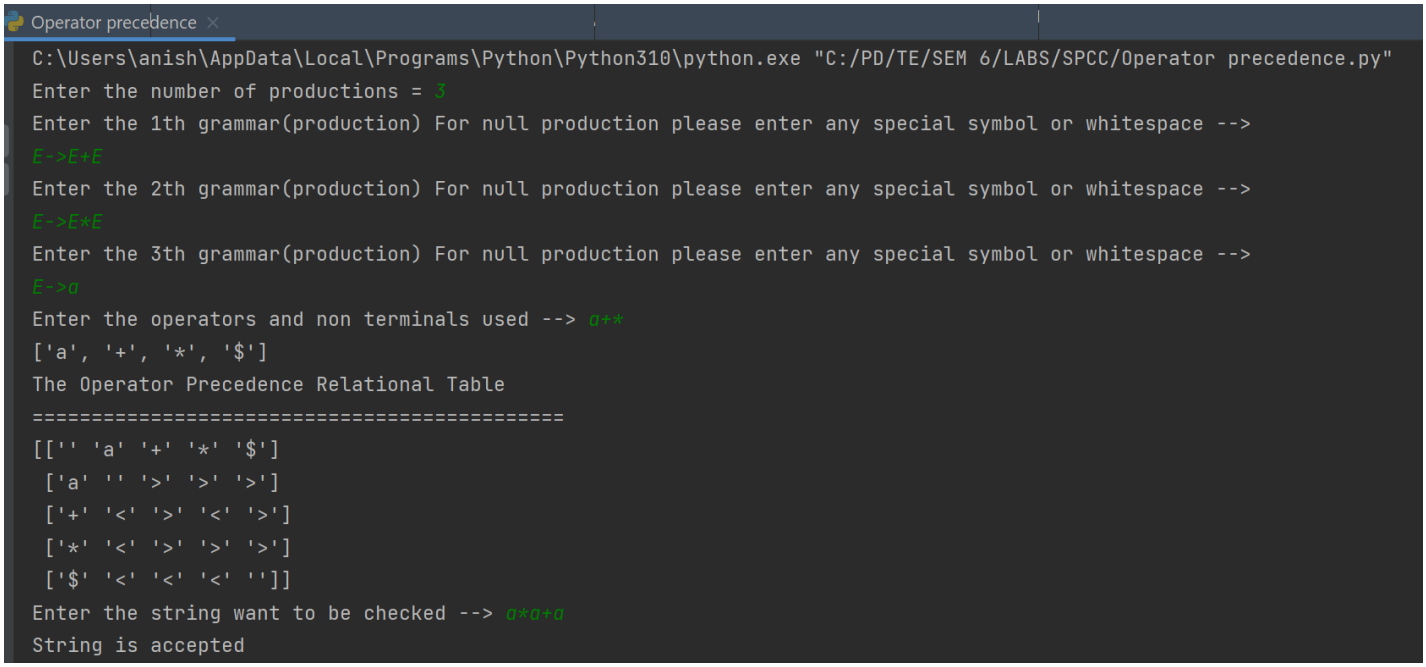
Code:

```
import numpy as np
def stringcheck():
    a = list(input("Enter the operators and non terminals used --> "))
    a.append('$')
    print(a)
    l = list("abcdefghijklmnopqrstuvwxyz")
    o = list('(/*%+-)')
    p = list('(/*%+-)')
    n = np.empty([len(a) + 1, len(a) + 1], dtype=str, order="C")
    for j in range(1, len(a) + 1):
        n[0][j] = a[j - 1]
        n[j][0] = a[j - 1]
    for i in range(1, len(a) + 1):
        for j in range(1, len(a) + 1):
            if ((n[i][0] in l) and (n[0][j] in l)):
                n[i][j] = ""
            elif ((n[i][0] in l)):
                n[i][j] = ">"
            elif ((n[i][0] in o) and (n[0][j] in o)):
                if (o.index(n[i][0]) <= o.index(n[0][j])):
                    n[i][j] = ">"
                else:
                    n[i][j] = "<"
            elif ((n[i][0] in o) and n[0][j] in l):
                n[i][j] = "<"
            elif (n[i][0] == "$" and n[0][j] != "$"):
                n[i][j] = "<"
            elif (n[0][j] == "$" and n[i][0] != "$"):
                n[i][j] = ">"
            else:
                break
    print("The Operator Precedence Relational
Table\n============================================")
    print(n)
    i = list(input("Enter the string want to be checked --> "))
    i.append("$")
    s = [None] * len(i)
    q = 0
    s.insert(q, "$")
    x = [row[0] for row in n]
    y = list(n[0])
    h = 0
    while (s[0] != s[1]):
        if ((i[len(i) - 2] in p)):
            break
        elif ((s[q] in x) and (i[h] in y)):
            if (n[x.index(s[q])][y.index(i[h])] == "<"):
                q += 1
                s.insert(q, i[h])
                h += 1
            elif (n[x.index(s[q])][y.index(i[h])] == ">"):
                s.pop(q)
                q -= 1
            elif ((n[x.index(s[q])][y.index(i[h])] == '') and ((s[q] == "$") and (i[h] == "$"))):
                s[1] = s[0]
        else:
            break
    if (s[0] != s[1]):
        return False
    else:
        return True
```

```
def grammarcheck(i):
    print("Enter the",str(i + 1) + "th grammar(production) For null production please enter any special symbol or
whitespace --> ")
    b = list(input().split("->"))
    f = list("abcdefghijklmnopqrstuvwxyz")
    if (b[0] == " " or b[0] == "" or b[0] in f or len(b) == 1):
        return False
    else:
        b.pop(0)
        b = list(b[0])
        s = list("ABCDEFGHIJKLMNOPQRSTUVWXYZ")
        o = list("(abcdefghijklmnopqrstuvwxyz^/*+-|)")
    sp = ['!', '@', '#', '$', '?', '~', '`', ',', ';', ':', '"', '=', '_', '&', "'", "", " "]
    for i in range(0, len(b), 2):
        if (b[i] == " "):
            g = False
        elif (b[i] in sp):
            g = False
            break
        elif (b[len(b) - 1] in o and ((b[0] == "(" and b[len(b) - 1] == ")") or (b.count("(") == b.count(")")))):
            g = True
        elif (b[i] in f):
            g = True
        elif (b[len(b) - 1] in o):
            g = False
        elif ((i == len(b) - 1) and (b[i] in s)):
            g = True
        elif ((i == len(b) - 1) and (b[i] not in s) and (b[i] in o) and b[i - 1] in o):
            g = True
        elif ((b[i] in s) and (b[i + 1] in o)):
            g = True
        elif ((b[i] in s) and (b[i + 1] in s)):
            g = False
            break
        else:
            g = False
            break
        if (g == True):
            return True
        else:
            return False


c = int(input("Enter the number of productions = "))
for i in range(c):
    if (grammarcheck(i)):
        t = True
    else:
        t = False
        break
if t:
    if (stringcheck()):
        print("String is accepted")
    else:
        print("String is not accepted")
else:
    print("Grammar is not accepted ")
```

Output:


Operator precedence
C:\Users\anish\AppData\Local\Programs\Python\Python310\python.exe "C:/PD/TE/SEM 6/LABS/SPCC/Operator precedence.py"
Enter the number of productions = 3
Enter the 1th grammar(production) For null production please enter any special symbol or whitespace -->
E->E+E
Enter the 2th grammar(production) For null production please enter any special symbol or whitespace -->
E->E*E
Enter the 3th grammar(production) For null production please enter any special symbol or whitespace -->
E->a
Enter the operators and non terminals used --> a+*
['a', '+', '*', '$']
The Operator Precedence Relational Table
==========
[['' 'a' '+' '*' '$']
['a' '' '>' '>' '>']
['+' '<' '>' '<' '>']
['*' '<' '>' '>' '>']
['$' '<' '<' '<' '']]
Enter the string want to be checked --> a*a+a
String is accepted