Compiler Design(18CSC304J)

Experiment 7

Computation of First

Harsh Goel
RA1811003010185

Aim: To study and implement Computation of First.

Language: Python

Procedure:

1. Create a file or select the file for performing the operations on.
2. Start any python IDE and type the necessary code.
3. Run the code and perform the operations required.
4. Note the output and document it.

Algorithm:

1. If x is a terminal, then FIRST(x) = { 'x' }
2. If x-> Є, is a production rule, then add Є to FIRST(x).
3. If X->Y1 Y2 Y3….Yn is a production,
   a) FIRST(X) = FIRST(Y1)
   b) If FIRST(Y1) contains Є then FIRST(X) = { FIRST(Y1) – Є } U { FIRST(Y2) }
   c) If FIRST (Yi) contains Є for all i = 1 to n, then add Є to FIRST(X).

Code Snippet:

```
import sys

sys.setrecursionlimit(60)


def first(string):
    first_ = set()
    if string in non_terminals:
        alternatives = productions_dict[string]

        for alternative in alternatives:
            first_2 = first(alternative)
            first_ = first_ | first_2

    elif string in terminals:
```

```
        first_ = {string}

    elif string == '' or string == 'ə':
        first_ = {'ə'}

    else:
        first_2 = first(string[0])
        if 'ə' in first_2:
            i = 1
            while 'ə' in first_2:
                # print("inside while")

                first_ = first_ | (first_2 - {'ə'})
                # print('string[i:]=', string[i:])
                if string[i:] in terminals:
                    first_ = first_ | {string[i:]}
                    break
                elif string[i:] == '':
                    first_ = first_ | {'ə'}
                    break
                first_2 = first(string[i:])
                first_ = first_ | first_2 - {'ə'}
                i += 1
        else:
            first_ = first_ | first_2
    return first_


no_of_terminals = int(input("Enter no. of terminals: "))

terminals = []

print("Enter the terminals :")
for _ in range(no_of_terminals):
    terminals.append(input())

no_of_non_terminals = int(input("Enter no. of non terminals: "))

non_terminals = []

print("Enter the non terminals :")
for _ in range(no_of_non_terminals):
    non_terminals.append(input())

starting_symbol = input("Enter the starting symbol: ")

no_of_productions = int(input("Enter no of productions: "))

productions = []
```

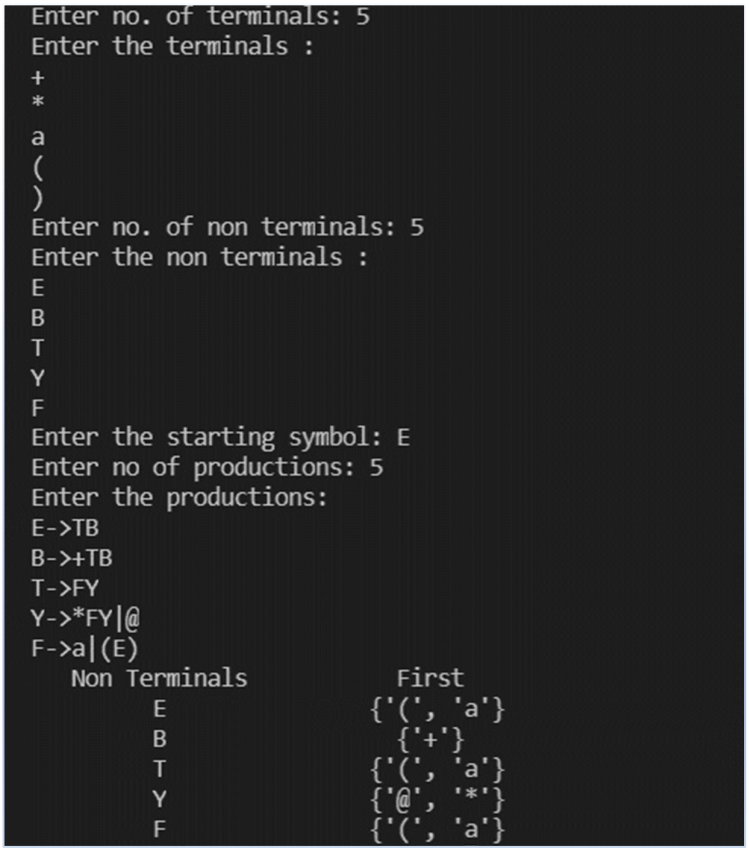
```
print("Enter the productions:")
for _ in range(no_of_productions):
    productions.append(input())

productions_dict = {}

for nT in non_terminals:
    productions_dict[nT] = []

for production in productions:
    nonterm_to_prod = production.split("->")
    alternatives = nonterm_to_prod[1].split("|")
    for alternative in alternatives:
        productions_dict[nonterm_to_prod[0]].append(alternative)
FIRST = {}
for non_terminal in non_terminals:
    FIRST[non_terminal] = set()
for non_terminal in non_terminals:
    FIRST[non_terminal] = FIRST[non_terminal] | first(non_terminal)
print("{: ^20}{: ^20}".format('Non Terminals', 'First'))
for non_terminal in non_terminals:
    print("{: ^20}{: ^20}".format(non_terminal, str(FIRST[non_termina
l]), ))
```

Output Screenshots:



Result:

The code was successfully implemented in Python and output was recorded.