Testing

To run&build:

The program requires python3.

Go to source directory and execute python3 main.py

This file presents testing scenarios with outputs. The file is divided into 3 sections:

- CFG
- Input string
- Parsing

Each section contains tests cases corresponding to the section name.

Tests can be run only one by one.

Context-free grammar testing.

The program should check if provided grammar is valid before proceeding.

In order to run tests please run the program and select the file from a path ./tests/CFG_tests/

1. **Incorrect usages.**

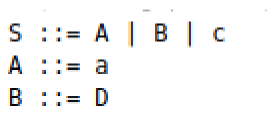
- test1.txt

Production rule contains non-terminal symbol which is not defined.

Expected output:

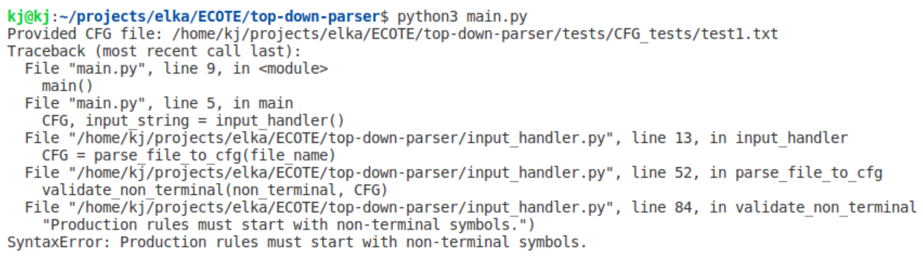
The program should raise an error with the message "Invalid CFG".

CFG:



```
S ::= A | B | c
A ::= a
B ::= D
```

Output:



```
kj@kj:~/projects/elka/ECOTE/top-down-parser$ python3 main.py
Provided CFG file: /home/kj/projects/elka/ECOTE/top-down-parser/tests/CFG_tests/test1.txt
Traceback (most recent call last):
  File "main.py", line 9, in <module>
    main()
  File "main.py", line 5, in main
    CFG, input_string = input_handler()
  File "/home/kj/projects/elka/ECOTE/top-down-parser/input_handler.py", line 13, in input_handler
    CFG = parse_file_to_cfg(file_name)
  File "/home/kj/projects/elka/ECOTE/top-down-parser/input_handler.py", line 52, in parse_file_to_cfg
    validate_non_terminal(non_terminal, CFG)
  File "/home/kj/projects/elka/ECOTE/top-down-parser/input_handler.py", line 84, in validate_non_terminal
    "Production rules must start with non-terminal symbols.")
SyntaxError: Production rules must start with non-terminal symbols.
```

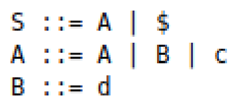
- test2.txt

Left recursion.

Expected output:

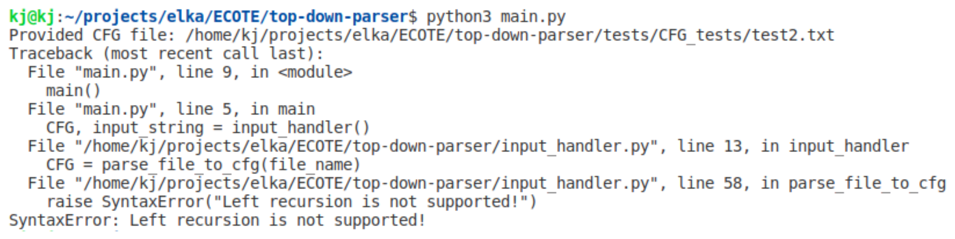
The program should raise an error with the message "*Left recursion is not supported!*".

CFG:



```
S ::= A | $
A ::= A | B | c
B ::= d
```

Output:



```
kj@kj:~/projects/elka/ECOTE/top-down-parser$ python3 main.py
Provided CFG file: /home/kj/projects/elka/ECOTE/top-down-parser/tests/CFG_tests/test2.txt
Traceback (most recent call last):
  File "main.py", line 9, in <module>
    main()
  File "main.py", line 5, in main
    CFG, input_string = input_handler()
  File "/home/kj/projects/elka/ECOTE/top-down-parser/input_handler.py", line 13, in input_handler
    CFG = parse_file_to_cfg(file_name)
  File "/home/kj/projects/elka/ECOTE/top-down-parser/input_handler.py", line 58, in parse_file_to_cfg
    raise SyntaxError("Left recursion is not supported!")
SyntaxError: Left recursion is not supported!
```

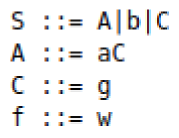
- test3.txt

Production rules defined by a terminal symbol.

Expected output:

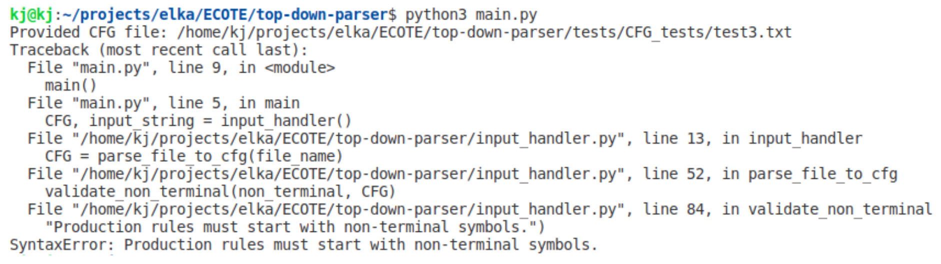
The program should raise an error with the message "*Production rules must start with non-terminal symbols.*".

CFG:



```
S ::= A|b|C
A ::= aC
C ::= g
f ::= w
```

Output:



```
kj@kj:~/projects/elka/ECOTE/top-down-parser$ python3 main.py
Provided CFG file: /home/kj/projects/elka/ECOTE/top-down-parser/tests/CFG_tests/test3.txt
Traceback (most recent call last):
  File "main.py", line 9, in <module>
    main()
  File "main.py", line 5, in main
    CFG, input_string = input_handler()
  File "/home/kj/projects/elka/ECOTE/top-down-parser/input_handler.py", line 13, in input_handler
    CFG = parse_file_to_cfg(file_name)
  File "/home/kj/projects/elka/ECOTE/top-down-parser/input_handler.py", line 52, in parse_file_to_cfg
    validate_non_terminal(non_terminal, CFG)
  File "/home/kj/projects/elka/ECOTE/top-down-parser/input_handler.py", line 84, in validate_non_terminal
    "Production rules must start with non-terminal symbols.")
SyntaxError: Production rules must start with non-terminal symbols.
```

- test4.txt

Duplicated non-terminal symbols.

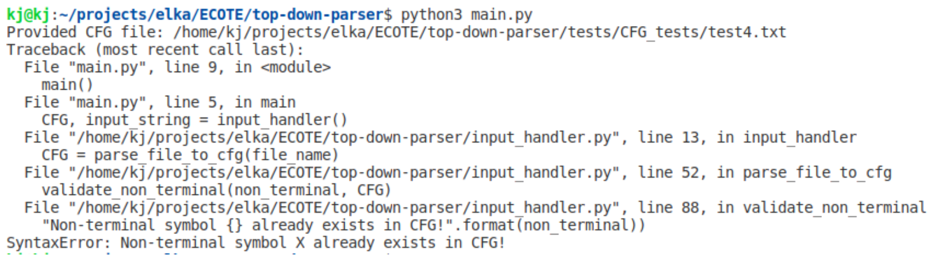
Expected output:

The program should raise an error with the message "*Non-terminal symbol {} already exists in CFG!*".

CFG:

```
S ::= X|Y
X ::= x
X ::= y
```

Output:



```
kj@kj:~/projects/elka/ECOTE/top-down-parser$ python3 main.py
Provided CFG file: /home/kj/projects/elka/ECOTE/top-down-parser/tests/CFG_tests/test4.txt
Traceback (most recent call last):
  File "main.py", line 9, in <module>
    main()
  File "main.py", line 5, in main
    CFG, input_string = input_handler()
  File "/home/kj/projects/elka/ECOTE/top-down-parser/input_handler.py", line 13, in input_handler
    CFG = parse_file_to_cfg(file_name)
  File "/home/kj/projects/elka/ECOTE/top-down-parser/input_handler.py", line 52, in parse_file_to_cfg
    validate_non_terminal(non_terminal, CFG)
  File "/home/kj/projects/elka/ECOTE/top-down-parser/input_handler.py", line 88, in validate_non_terminal
    "Non-terminal symbol {} already exists in CFG!".format(non_terminal))
SyntaxError: Non-terminal symbol X already exists in CFG!
```

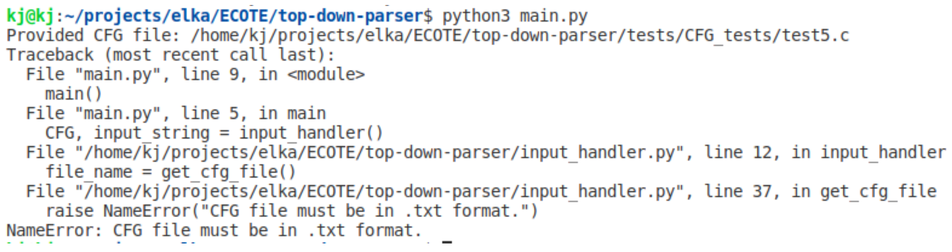
- test5.c

CFG provided in a file with different extension than .txt

Expected output:

The program should raise an error with the message "*CFG file must be in .txt format.*".

Output:



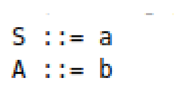
```
kj@kj:~/projects/elka/ECOTE/top-down-parser$ python3 main.py
Provided CFG file: /home/kj/projects/elka/ECOTE/top-down-parser/tests/CFG_tests/test5.c
Traceback (most recent call last):
  File "main.py", line 9, in <module>
    main()
  File "main.py", line 5, in main
    CFG, input_string = input_handler()
  File "/home/kj/projects/elka/ECOTE/top-down-parser/input_handler.py", line 12, in input_handler
    file_name = get_cfg_file()
  File "/home/kj/projects/elka/ECOTE/top-down-parser/input_handler.py", line 37, in get_cfg_file
    raise NameError("CFG file must be in .txt format.")
NameError: CFG file must be in .txt format.
```

2. Correct usages.

The program should properly parse the provided CFG and then ask for the input string. No other messages should be displayed.

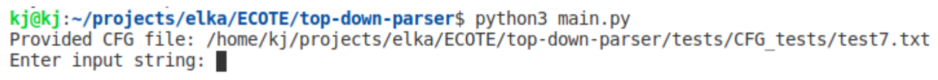
- test6.txt

CFG:



```
S ::= a
A ::= b
```

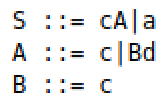
Output:



```
kj@kj:~/projects/elka/ECOTE/top-down-parser$ python3 main.py
Provided CFG file: /home/kj/projects/elka/ECOTE/top-down-parser/tests/CFG_tests/test7.txt
Enter input string: 
```

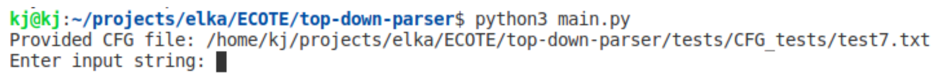
- test7.txt

CFG:



```
S ::= cA|a
A ::= c|Bd
B ::= c
```

Output:



```
kj@kj:~/projects/elka/ECOTE/top-down-parser$ python3 main.py
Provided CFG file: /home/kj/projects/elka/ECOTE/top-down-parser/tests/CFG_tests/test7.txt
Enter input string: 
```

Input string testing

In order to run test please run the program and select any of the files from the path: ./tests/correct_CFG/

1. **Incorrect usages.**

- Test 1

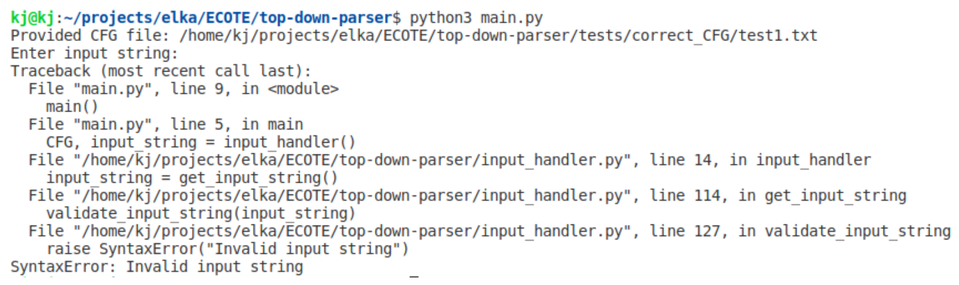
Empty string as an input.

Do not type anything in the command line, press enter.

Expected output:

The program should raise an error with the message "*Invalid input string*".

Output:



```
kj@kj:~/projects/elka/ECOTE/top-down-parser$ python3 main.py
Provided CFG file: /home/kj/projects/elka/ECOTE/top-down-parser/tests/correct_CFG/test1.txt
Enter input string:
Traceback (most recent call last):
  File "main.py", line 9, in <module>
    main()
  File "main.py", line 5, in main
    CFG, input_string = input_handler()
  File "/home/kj/projects/elka/ECOTE/top-down-parser/input_handler.py", line 14, in input_handler
    input_string = get_input_string()
  File "/home/kj/projects/elka/ECOTE/top-down-parser/input_handler.py", line 114, in get_input_string
    validate_input_string(input_string)
  File "/home/kj/projects/elka/ECOTE/top-down-parser/input_handler.py", line 127, in validate_input_string
    raise SyntaxError("Invalid input string")
SyntaxError: Invalid input string
```

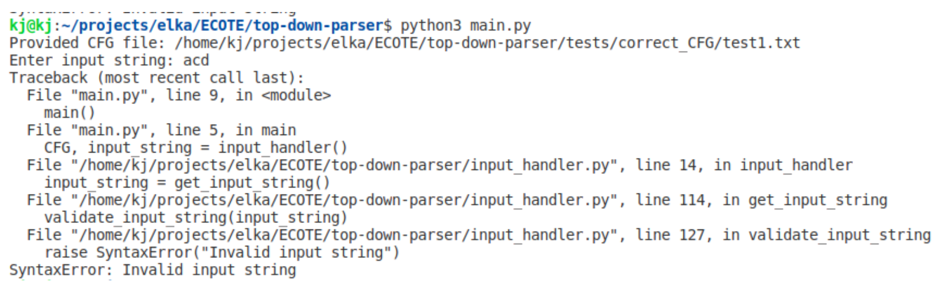
- Test 2

Input string not ending with '$' symbol.

Expected output:

The program should raise an error with the message "*Invalid input string*".

Output:



```
kj@kj:~/projects/elka/ECOTE/top-down-parser$ python3 main.py
Provided CFG file: /home/kj/projects/elka/ECOTE/top-down-parser/tests/correct_CFG/test1.txt
Enter input string: acd
Traceback (most recent call last):
  File "main.py", line 9, in <module>
    main()
  File "main.py", line 5, in main
    CFG, input_string = input_handler()
  File "/home/kj/projects/elka/ECOTE/top-down-parser/input_handler.py", line 14, in input_handler
    input_string = get_input_string()
  File "/home/kj/projects/elka/ECOTE/top-down-parser/input_handler.py", line 114, in get_input_string
    validate_input_string(input_string)
  File "/home/kj/projects/elka/ECOTE/top-down-parser/input_handler.py", line 127, in validate_input_string
    raise SyntaxError("Invalid input string")
SyntaxError: Invalid input string
```

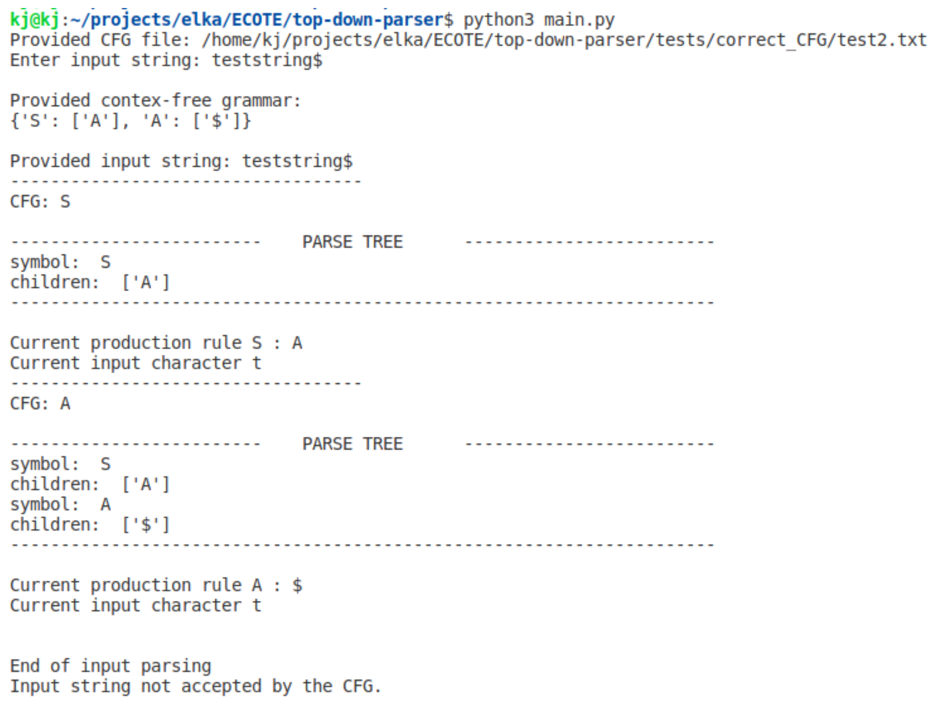
2. **Correct usages.**

- Test 3

Input any string ending with ‘$’ symbol.

Expected output:

The program should parse the provided input, print parse trees and the message about string being accepted or not.



```
kj@kj:~/projects/elka/ECOTE/top-down-parser$ python3 main.py
Provided CFG file: /home/kj/projects/elka/ECOTE/top-down-parser/tests/correct_CFG/test2.txt
Enter input string: teststring$

Provided contex-free grammar:
{'S': ['A'], 'A': ['$']}

Provided input string: teststring$
-----------------------------------
CFG: S

------------------------    PARSE TREE     ------------------------
symbol:  S
children:  ['A']
---------------------------------------------------------------------

Current production rule S : A
Current input character t
-----------------------------------
CFG: A

------------------------    PARSE TREE     ------------------------
symbol:  S
children:  ['A']
symbol:  A
children:  ['$']
---------------------------------------------------------------------

Current production rule A : $
Current input character t


End of input parsing
Input string not accepted by the CFG.
```

Parser testing

1. String is accepted.

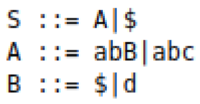
Expected output: At each stage we should see parsing tree and when the parsing is finished – final parsing tree and the message: Input string accepted by the CFG

Please run the program and select the CFG: ./tests/correct_CFG/

- test1.txt

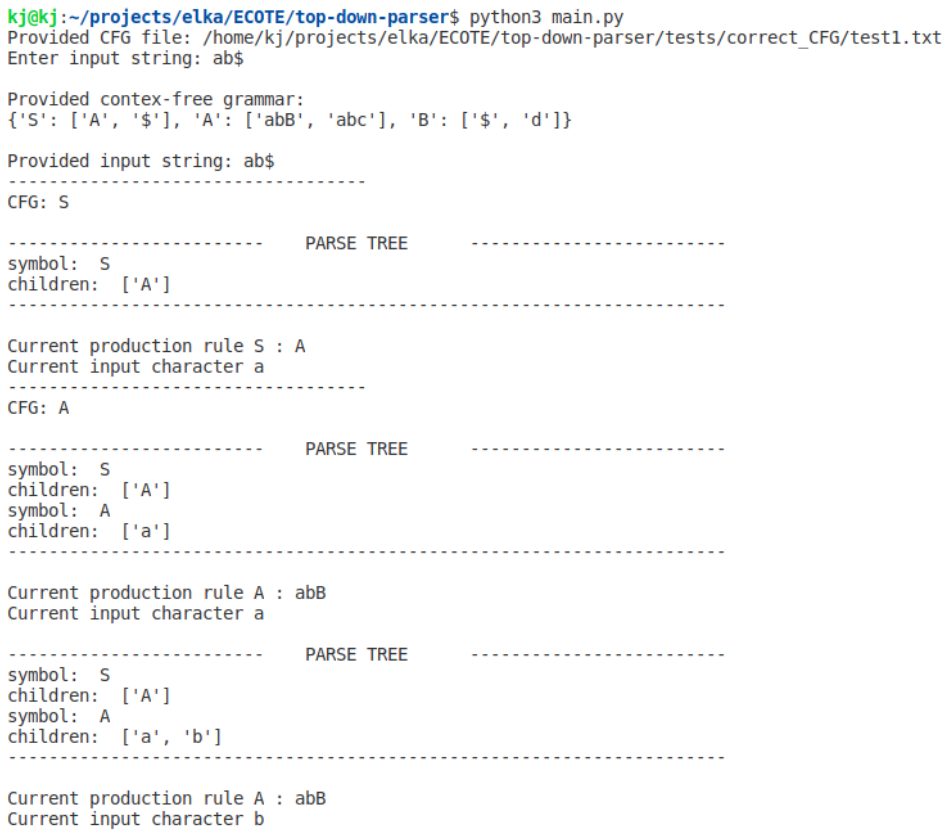
Input string: **ab$**

CFG:

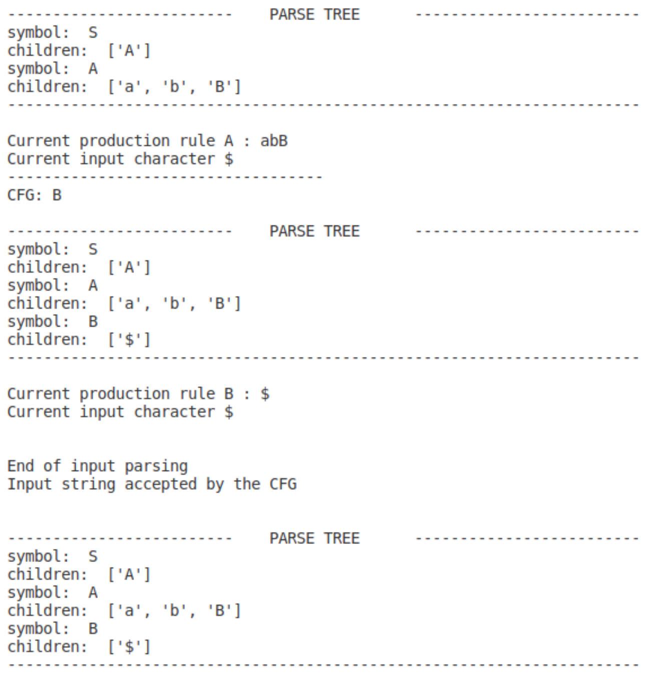


```
S ::= A|$
A ::= abB|abc
B ::= $|d
```

Output:



```
kj@kj:~/projects/elka/ECOTE/top-down-parser$ python3 main.py
Provided CFG file: /home/kj/projects/elka/ECOTE/top-down-parser/tests/correct_CFG/test1.txt
Enter input string: ab$

Provided contex-free grammar:
{'S': ['A', '$'], 'A': ['abB', 'abc'], 'B': ['$', 'd']}

Provided input string: ab$
-----------------------------------
CFG: S

------------------------    PARSE TREE     ------------------------
symbol:  S
children:  ['A']
----------------------------------------------------------------------

Current production rule S : A
Current input character a
-----------------------------------
CFG: A

------------------------    PARSE TREE     ------------------------
symbol:  S
children:  ['A']
symbol:  A
children:  ['a']
----------------------------------------------------------------------

Current production rule A : abB
Current input character a

------------------------    PARSE TREE     ------------------------
symbol:  S
children:  ['A']
symbol:  A
children:  ['a', 'b']
----------------------------------------------------------------------

Current production rule A : abB
Current input character b
```



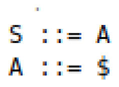
```
------------------------    PARSE TREE     ------------------------
symbol:  S
children:  ['A']
symbol:  A
children:  ['a', 'b', 'B']
--------------------------------------------------------------------

Current production rule A : abB
Current input character $
----------------------------------
CFG: B

------------------------    PARSE TREE     ------------------------
symbol:  S
children:  ['A']
symbol:  A
children:  ['a', 'b', 'B']
symbol:  B
children:  ['$']
--------------------------------------------------------------------

Current production rule B : $
Current input character $


End of input parsing
Input string accepted by the CFG


------------------------    PARSE TREE     ------------------------
symbol:  S
children:  ['A']
symbol:  A
children:  ['a', 'b', 'B']
symbol:  B
children:  ['$']
--------------------------------------------------------------------
```

- test2.txt

Empty symbol testing

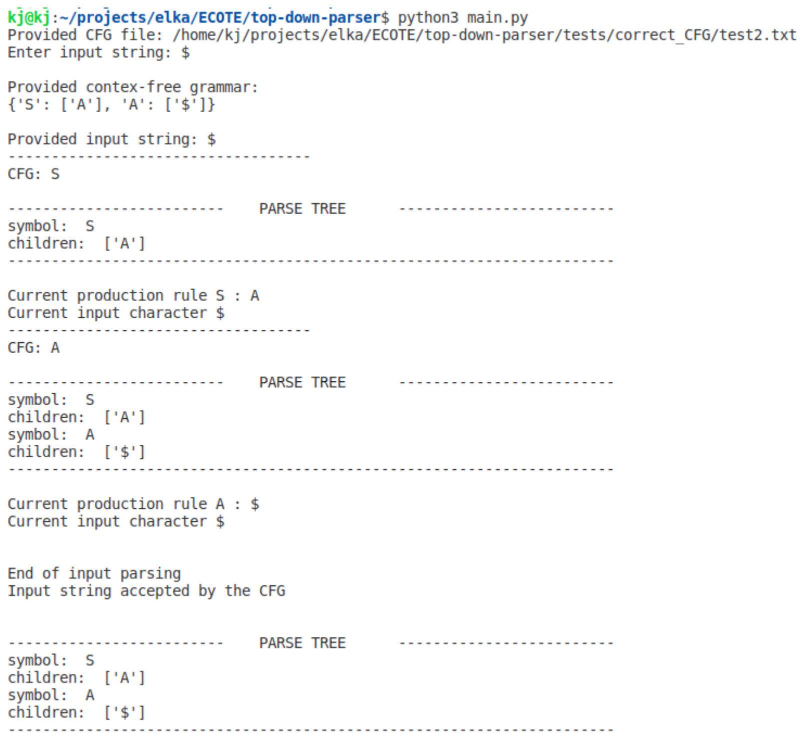
Input string: **$**

CFG:



```
S ::= A
A ::= $
```

Output:



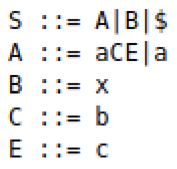
```
kj@kj:~/projects/elka/ECOTE/top-down-parser$ python3 main.py
Provided CFG file: /home/kj/projects/elka/ECOTE/top-down-parser/tests/correct_CFG/test2.txt
Enter input string: $

Provided contex-free grammar:
{'S': ['A'], 'A': ['$']}

Provided input string: $
----------------------------------
CFG: S

------------------------    PARSE TREE     ------------------------
symbol:  S
children:  ['A']
--------------------------------------------------------------------

Current production rule S : A
Current input character $
----------------------------------
CFG: A

------------------------    PARSE TREE     ------------------------
symbol:  S
children:  ['A']
symbol:  A
children:  ['$']
--------------------------------------------------------------------

Current production rule A : $
Current input character $


End of input parsing
Input string accepted by the CFG


------------------------    PARSE TREE     ------------------------
symbol:  S
children:  ['A']
symbol:  A
children:  ['$']
--------------------------------------------------------------------
```

- test3.txt

Two non-terminal symbols in one production rule.

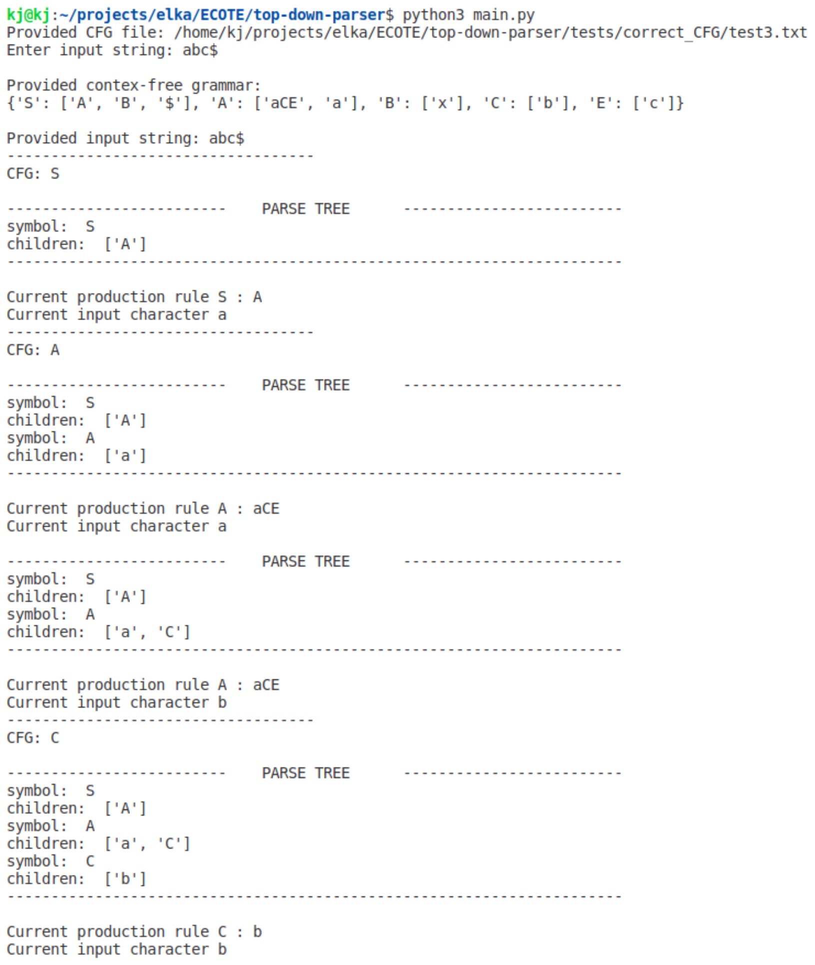
Input string: **abc$**

CFG:

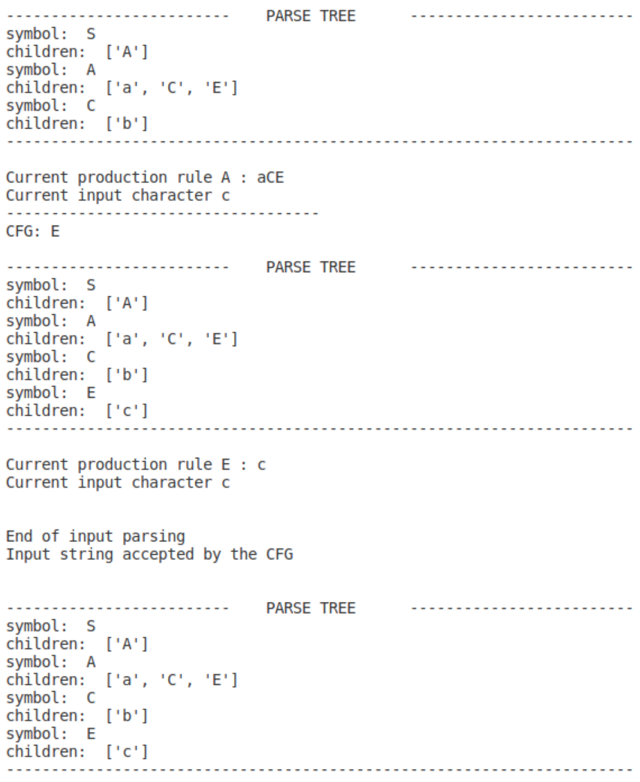


```
S ::= A|B|$
A ::= aCE|a
B ::= x
C ::= b
E ::= c
```

Output:

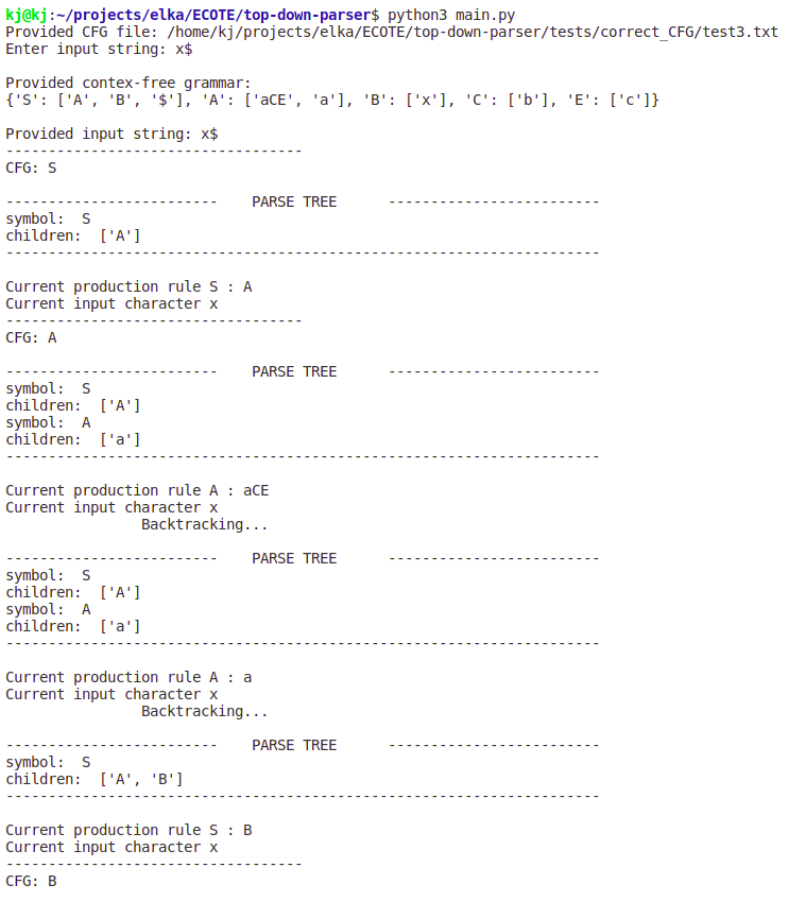


```
kj@kj:~/projects/elka/ECOTE/top-down-parser$ python3 main.py
Provided CFG file: /home/kj/projects/elka/ECOTE/top-down-parser/tests/correct_CFG/test3.txt
Enter input string: abc$

Provided contex-free grammar:
{'S': ['A', 'B', '$'], 'A': ['aCE', 'a'], 'B': ['x'], 'C': ['b'], 'E': ['c']}

Provided input string: abc$
-----------------------------------
CFG: S

-------------------------     PARSE TREE      -------------------------
symbol:  S
children:  ['A']
----------------------------------------------------------------------

Current production rule S : A
Current input character a
-----------------------------------
CFG: A

-------------------------     PARSE TREE      -------------------------
symbol:  S
children:  ['A']
symbol:  A
children:  ['a']
----------------------------------------------------------------------

Current production rule A : aCE
Current input character a

-------------------------     PARSE TREE      -------------------------
symbol:  S
children:  ['A']
symbol:  A
children:  ['a', 'C']
----------------------------------------------------------------------

Current production rule A : aCE
Current input character b
-----------------------------------
CFG: C

-------------------------     PARSE TREE      -------------------------
symbol:  S
children:  ['A']
symbol:  A
children:  ['a', 'C']
symbol:  C
children:  ['b']
----------------------------------------------------------------------

Current production rule C : b
Current input character b
```

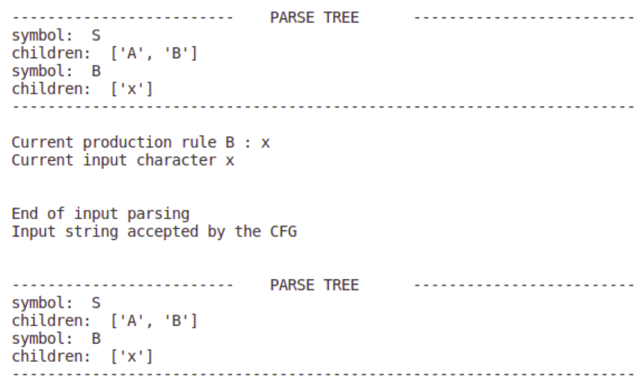


```
-------------------------     PARSE TREE      -------------------------
symbol:  S
children:  ['A']
symbol:  A
children:  ['a', 'C', 'E']
symbol:  C
children:  ['b']
----------------------------------------------------------------------

Current production rule A : aCE
Current input character c
-----------------------------------
CFG: E

-------------------------     PARSE TREE      -------------------------
symbol:  S
children:  ['A']
symbol:  A
children:  ['a', 'C', 'E']
symbol:  C
children:  ['b']
symbol:  E
children:  ['c']
----------------------------------------------------------------------

Current production rule E : c
Current input character c


End of input parsing
Input string accepted by the CFG


-------------------------     PARSE TREE      -------------------------
symbol:  S
children:  ['A']
symbol:  A
children:  ['a', 'C', 'E']
symbol:  C
children:  ['b']
symbol:  E
children:  ['c']
----------------------------------------------------------------------
```

Input string: **x$**



```
kj@kj:~/projects/elka/ECOTE/top-down-parser$ python3 main.py
Provided CFG file: /home/kj/projects/elka/ECOTE/top-down-parser/tests/correct_CFG/test3.txt
Enter input string: x$

Provided contex-free grammar:
{'S': ['A', 'B', '$'], 'A': ['aCE', 'a'], 'B': ['x'], 'C': ['b'], 'E': ['c']}

Provided input string: x$
-----------------------------------
CFG: S

-------------------------     PARSE TREE      -------------------------
symbol:  S
children:  ['A']
----------------------------------------------------------------------

Current production rule S : A
Current input character x
-----------------------------------
CFG: A

-------------------------     PARSE TREE      -------------------------
symbol:  S
children:  ['A']
symbol:  A
children:  ['a']
----------------------------------------------------------------------

Current production rule A : aCE
Current input character x
               Backtracking...

-------------------------     PARSE TREE      -------------------------
symbol:  S
children:  ['A']
symbol:  A
children:  ['a']
----------------------------------------------------------------------

Current production rule A : a
Current input character x
               Backtracking...

-------------------------     PARSE TREE      -------------------------
symbol:  S
children:  ['A', 'B']
----------------------------------------------------------------------

Current production rule S : B
Current input character x
-----------------------------------
CFG: B
```

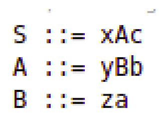


```
-------------------------     PARSE TREE      -------------------------
symbol:  S
children:  ['A', 'B']
symbol:  B
children:  ['x']
-----------------------------------------------------------------------

Current production rule B : x
Current input character x


End of input parsing
Input string accepted by the CFG


-------------------------     PARSE TREE      -------------------------
symbol:  S
children:  ['A', 'B']
symbol:  B
children:  ['x']
-----------------------------------------------------------------------
```

- test4.txt

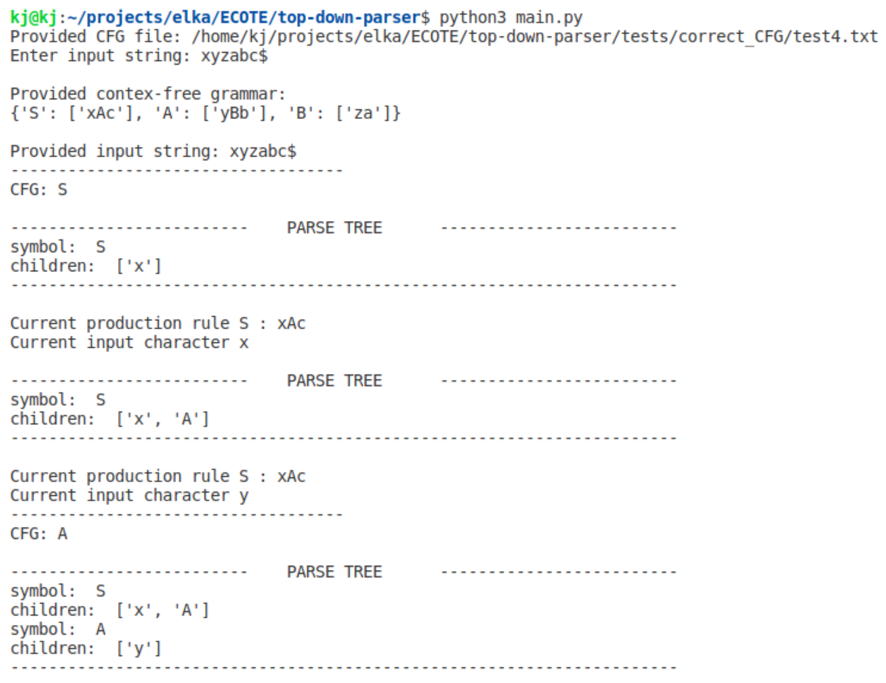
Input string: **xyzabc$**

CFG:



```
S ::= xAc
A ::= yBb
B ::= za
```

Output:



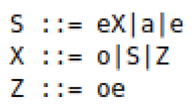
```
kj@kj:~/projects/elka/ECOTE/top-down-parser$ python3 main.py
Provided CFG file: /home/kj/projects/elka/ECOTE/top-down-parser/tests/correct_CFG/test4.txt
Enter input string: xyzabc$

Provided contex-free grammar:
{'S': ['xAc'], 'A': ['yBb'], 'B': ['za']}

Provided input string: xyzabc$
-----------------------------------
CFG: S

-------------------------     PARSE TREE      -------------------------
symbol:  S
children:  ['x']
-----------------------------------------------------------------------

Current production rule S : xAc
Current input character x

-------------------------     PARSE TREE      -------------------------
symbol:  S
children:  ['x', 'A']
-----------------------------------------------------------------------

Current production rule S : xAc
Current input character y
-----------------------------------
CFG: A

-------------------------     PARSE TREE      -------------------------
symbol:  S
children:  ['x', 'A']
symbol:  A
children:  ['y']
-----------------------------------------------------------------------
```

```
Current production rule A : yBb
Current input character y

-------------------------    PARSE TREE      -------------------------
symbol:  S
children:  ['x', 'A']
symbol:  A
children:  ['y', 'B']
-----------------------------------------------------------------------

Current production rule A : yBb
Current input character z
------------------------------------
CFG: B

-------------------------    PARSE TREE      -------------------------
symbol:  S
children:  ['x', 'A']
symbol:  A
children:  ['y', 'B']
symbol:  B
children:  ['z']
-----------------------------------------------------------------------

Current production rule B : za
Current input character z


-------------------------    PARSE TREE      -------------------------
symbol:  S
children:  ['x', 'A']
symbol:  A
children:  ['y', 'B']
symbol:  B
children:  ['z', 'a']
-----------------------------------------------------------------------

Current production rule B : za
Current input character a

-------------------------    PARSE TREE      -------------------------
symbol:  S
children:  ['x', 'A']
symbol:  A
children:  ['y', 'B', 'b']
symbol:  B
children:  ['z', 'a']
-----------------------------------------------------------------------

Current production rule A : yBb
Current input character b

-------------------------    PARSE TREE      -------------------------
symbol:  S
children:  ['x', 'A', 'c']
symbol:  A
children:  ['y', 'B', 'b']
symbol:  B
children:  ['z', 'a']
-----------------------------------------------------------------------

Current production rule S : xAc
Current input character c


End of input parsing
Input string accepted by the CFG


-------------------------    PARSE TREE      -------------------------
symbol:  S
children:  ['x', 'A', 'c']
symbol:  A
children:  ['y', 'B', 'b']
symbol:  B
children:  ['z', 'a']
-----------------------------------------------------------------------
```

- test5.txt

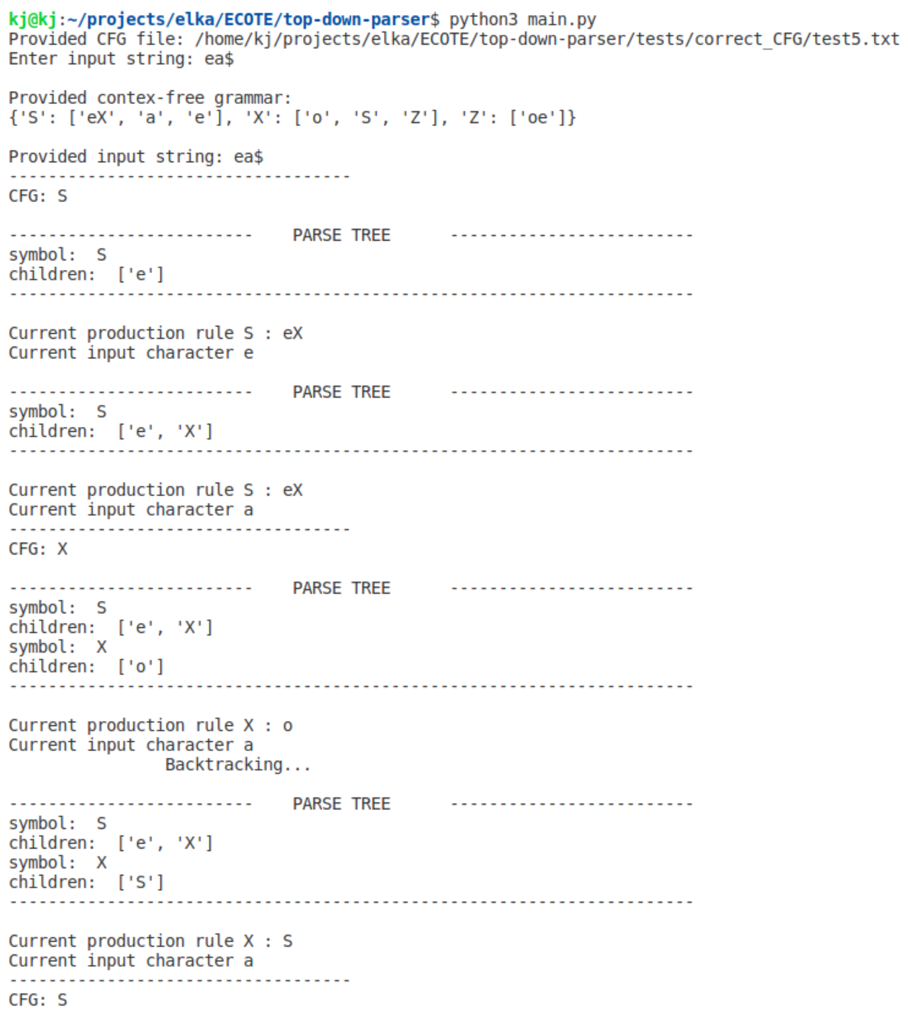
Input string: **ea$**

CFG:

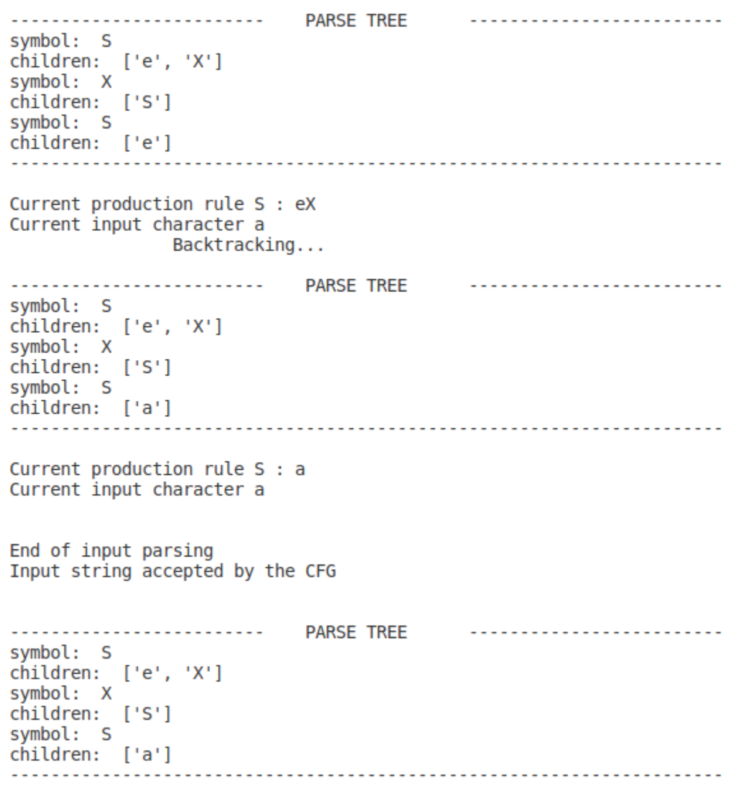


```
S ::= eX|a|e
X ::= o|S|Z
Z ::= oe
```

Output:

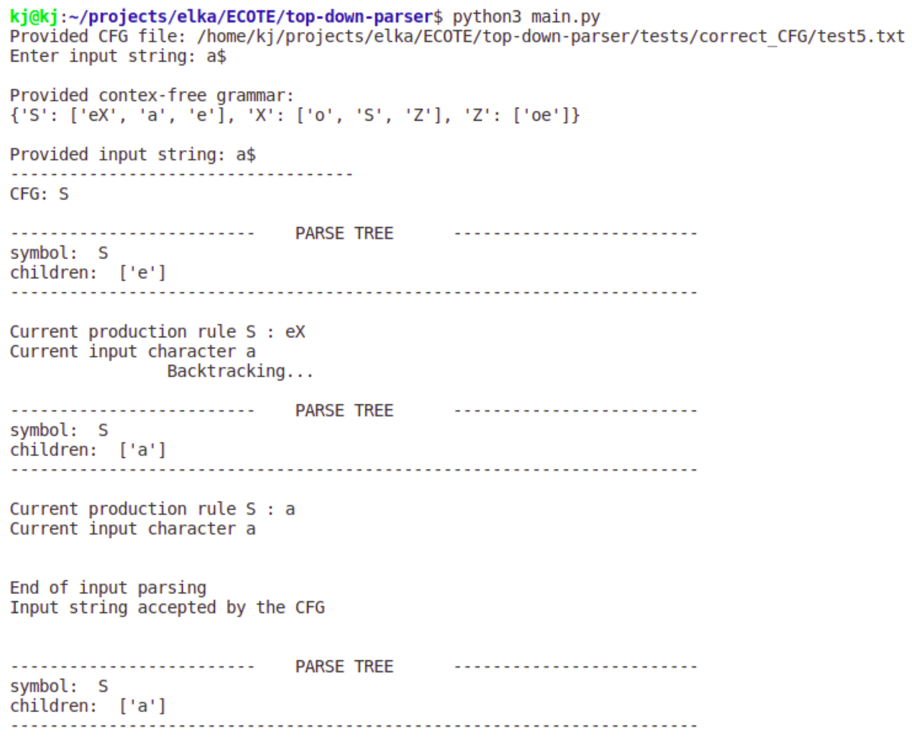


```
kj@kj:~/projects/elka/ECOTE/top-down-parser$ python3 main.py
Provided CFG file: /home/kj/projects/elka/ECOTE/top-down-parser/tests/correct_CFG/test5.txt
Enter input string: ea$

Provided contex-free grammar:
{'S': ['eX', 'a', 'e'], 'X': ['o', 'S', 'Z'], 'Z': ['oe']}

Provided input string: ea$
-----------------------------------
CFG: S

-------------------------    PARSE TREE     -------------------------
symbol:  S
children:  ['e']
----------------------------------------------------------------------

Current production rule S : eX
Current input character e

-------------------------    PARSE TREE     -------------------------
symbol:  S
children:  ['e', 'X']
----------------------------------------------------------------------

Current production rule S : eX
Current input character a
-----------------------------------
CFG: X

-------------------------    PARSE TREE     -------------------------
symbol:  S
children:  ['e', 'X']
symbol:  X
children:  ['o']
----------------------------------------------------------------------

Current production rule X : o
Current input character a
                Backtracking...

-------------------------    PARSE TREE     -------------------------
symbol:  S
children:  ['e', 'X']
symbol:  X
children:  ['S']
----------------------------------------------------------------------

Current production rule X : S
Current input character a
-----------------------------------
CFG: S
```



```
-------------------------    PARSE TREE     -------------------------
symbol:  S
children:  ['e', 'X']
symbol:  X
children:  ['S']
symbol:  S
children:  ['e']
----------------------------------------------------------------------

Current production rule S : eX
Current input character a
                Backtracking...

-------------------------    PARSE TREE     -------------------------
symbol:  S
children:  ['e', 'X']
symbol:  X
children:  ['S']
symbol:  S
children:  ['a']
----------------------------------------------------------------------

Current production rule S : a
Current input character a


End of input parsing
Input string accepted by the CFG


-------------------------    PARSE TREE     -------------------------
symbol:  S
children:  ['e', 'X']
symbol:  X
children:  ['S']
symbol:  S
children:  ['a']
----------------------------------------------------------------------
```

Input string: **a$**

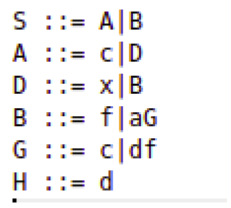


```
kj@kj:~/projects/elka/ECOTE/top-down-parser$ python3 main.py
Provided CFG file: /home/kj/projects/elka/ECOTE/top-down-parser/tests/correct_CFG/test5.txt
Enter input string: a$

Provided contex-free grammar:
{'S': ['eX', 'a', 'e'], 'X': ['o', 'S', 'Z'], 'Z': ['oe']}

Provided input string: a$
-----------------------------------
CFG: S

-------------------------    PARSE TREE     -------------------------
symbol:  S
children:  ['e']
----------------------------------------------------------------------

Current production rule S : eX
Current input character a
                Backtracking...

-------------------------    PARSE TREE     -------------------------
symbol:  S
children:  ['a']
----------------------------------------------------------------------

Current production rule S : a
Current input character a


End of input parsing
Input string accepted by the CFG


-------------------------    PARSE TREE     -------------------------
symbol:  S
children:  ['a']
----------------------------------------------------------------------
```

- test6.txt

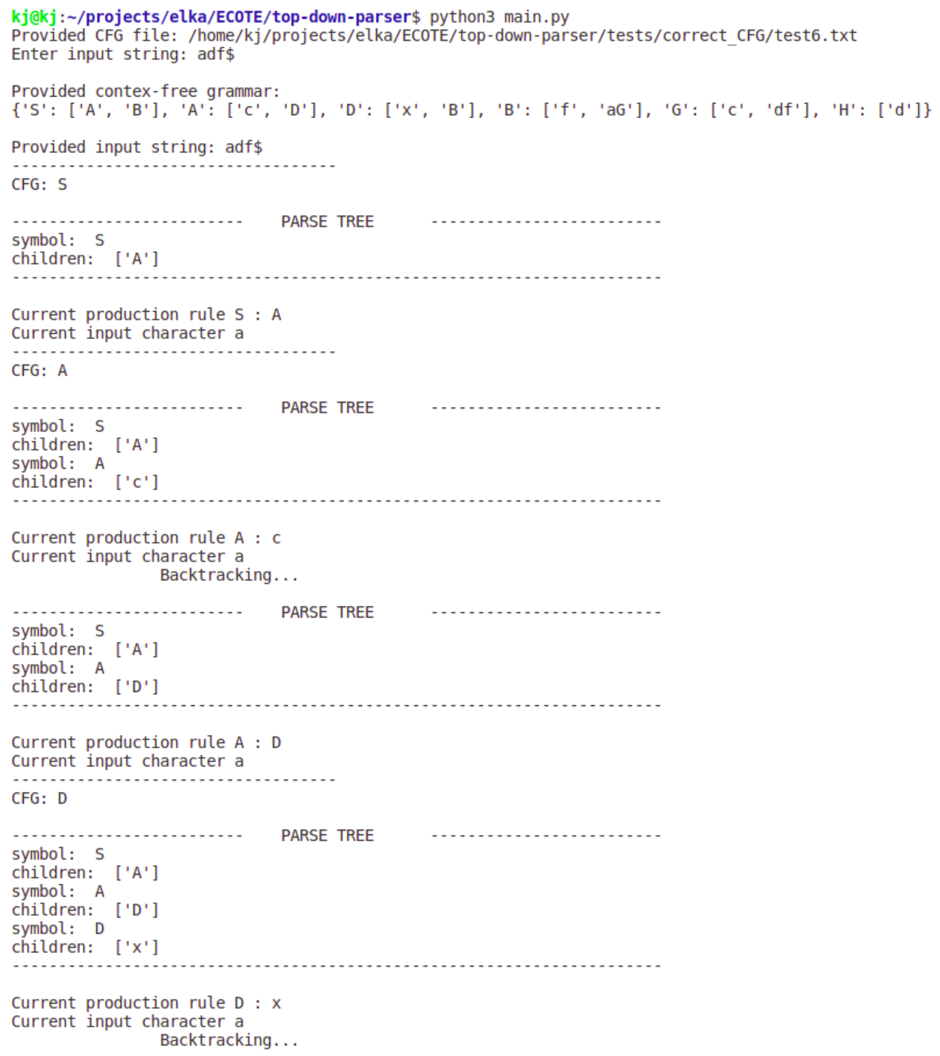
More complicated backtracking test.

CFG:



```
S ::= A|B
A ::= c|D
D ::= x|B
B ::= f|aG
G ::= c|df
H ::= d
```

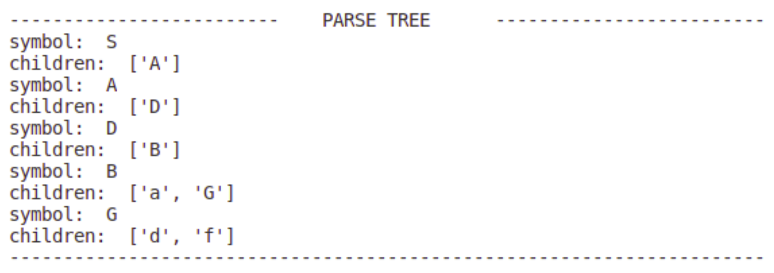
Output:



```
kj@kj:~/projects/elka/ECOTE/top-down-parser$ python3 main.py
Provided CFG file: /home/kj/projects/elka/ECOTE/top-down-parser/tests/correct_CFG/test6.txt
Enter input string: adf$

Provided contex-free grammar:
{'S': ['A', 'B'], 'A': ['c', 'D'], 'D': ['x', 'B'], 'B': ['f', 'aG'], 'G': ['c', 'df'], 'H': ['d']}

Provided input string: adf$
-----------------------------------
CFG: S

-------------------------    PARSE TREE     -------------------------
symbol:  S
children:  ['A']
----------------------------------------------------------------------

Current production rule S : A
Current input character a
-----------------------------------
CFG: A

-------------------------    PARSE TREE     -------------------------
symbol:  S
children:  ['A']
symbol:  A
children:  ['c']
----------------------------------------------------------------------

Current production rule A : c
Current input character a
                Backtracking...

-------------------------    PARSE TREE     -------------------------
symbol:  S
children:  ['A']
symbol:  A
children:  ['D']
----------------------------------------------------------------------

Current production rule A : D
Current input character a
-----------------------------------
CFG: D

-------------------------    PARSE TREE     -------------------------
symbol:  S
children:  ['A']
symbol:  A
children:  ['D']
symbol:  D
children:  ['x']
----------------------------------------------------------------------

Current production rule D : x
Current input character a
                Backtracking...
```

```
-------------------------    PARSE TREE      -------------------------
symbol:  S
children:  ['A']
symbol:  A
children:  ['D']
symbol:  D
children:  ['B']
----------------------------------------------------------------------

Current production rule D : B
Current input character a
-----------------------------------
CFG: B

-------------------------    PARSE TREE      -------------------------
symbol:  S
children:  ['A']
symbol:  A
children:  ['D']
symbol:  D
children:  ['B']
symbol:  B
children:  ['f']
----------------------------------------------------------------------

Current production rule B : f
Current input character a
                Backtracking...

-------------------------    PARSE TREE      -------------------------
symbol:  S
children:  ['A']
symbol:  A
children:  ['D']
symbol:  D
children:  ['B']
symbol:  B
children:  ['a']
----------------------------------------------------------------------

Current production rule B : aG
Current input character a

-------------------------    PARSE TREE      -------------------------
symbol:  S
children:  ['A']
symbol:  A
children:  ['D']
symbol:  D
children:  ['B']
symbol:  B
children:  ['a', 'G']
----------------------------------------------------------------------

Current production rule B : aG
Current input character d
-----------------------------------
```

```
CFG: G

------------------------    PARSE TREE     ------------------------
symbol:  S
children:  ['A']
symbol:  A
children:  ['D']
symbol:  D
children:  ['B']
symbol:  B
children:  ['a', 'G']
symbol:  G
children:  ['c']
--------------------------------------------------------------------

Current production rule G : c
Current input character d
               Backtracking...

------------------------    PARSE TREE     ------------------------
symbol:  S
children:  ['A']
symbol:  A
children:  ['D']
symbol:  D
children:  ['B']
symbol:  B
children:  ['a', 'G']
symbol:  G
children:  ['d']
--------------------------------------------------------------------

Current production rule G : df
Current input character d

------------------------    PARSE TREE     ------------------------
symbol:  S
children:  ['A']
symbol:  A
children:  ['D']
symbol:  D
children:  ['B']
symbol:  B
children:  ['a', 'G']
symbol:  G
children:  ['d', 'f']
--------------------------------------------------------------------

Current production rule G : df
Current input character f


End of input parsing
Input string accepted by the CFG
```



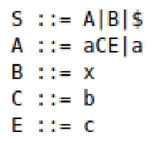
```
------------------------    PARSE TREE     ------------------------
symbol:  S
children:  ['A']
symbol:  A
children:  ['D']
symbol:  D
children:  ['B']
symbol:  B
children:  ['a', 'G']
symbol:  G
children:  ['d', 'f']
--------------------------------------------------------------------
```

2. String is not accepted.

- test4.txt

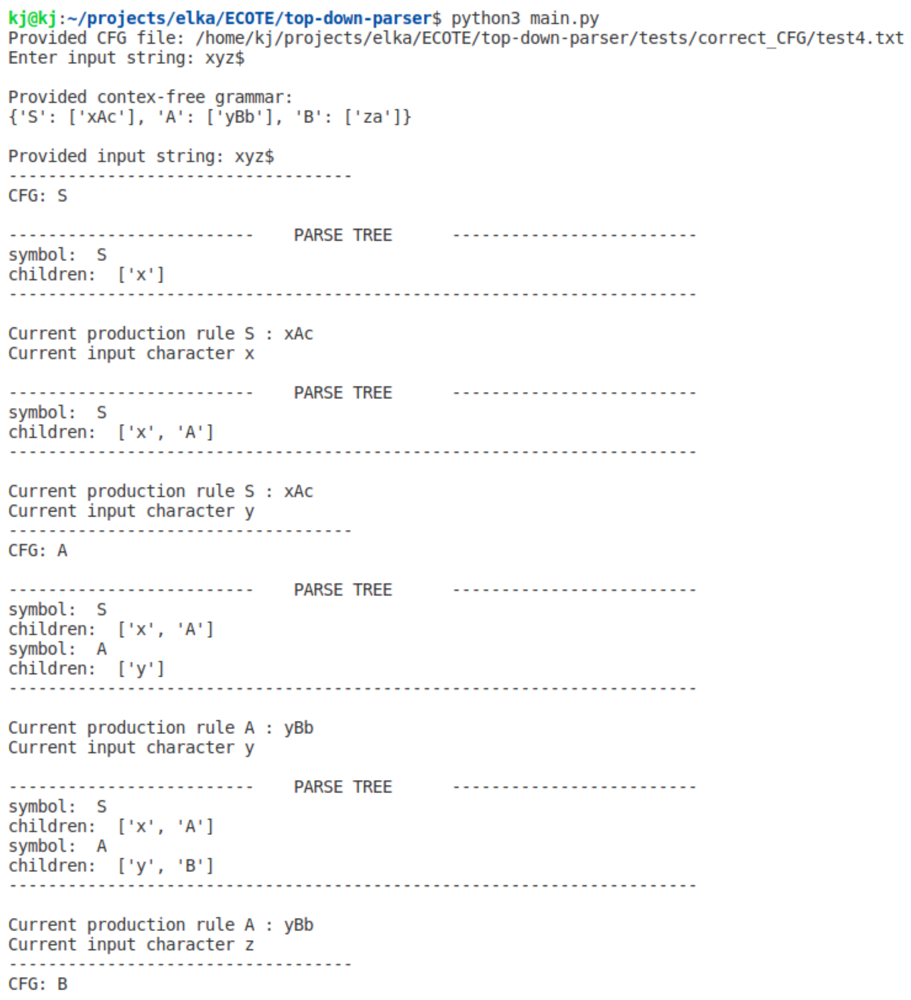
Input string: **xyz$**

CFG:

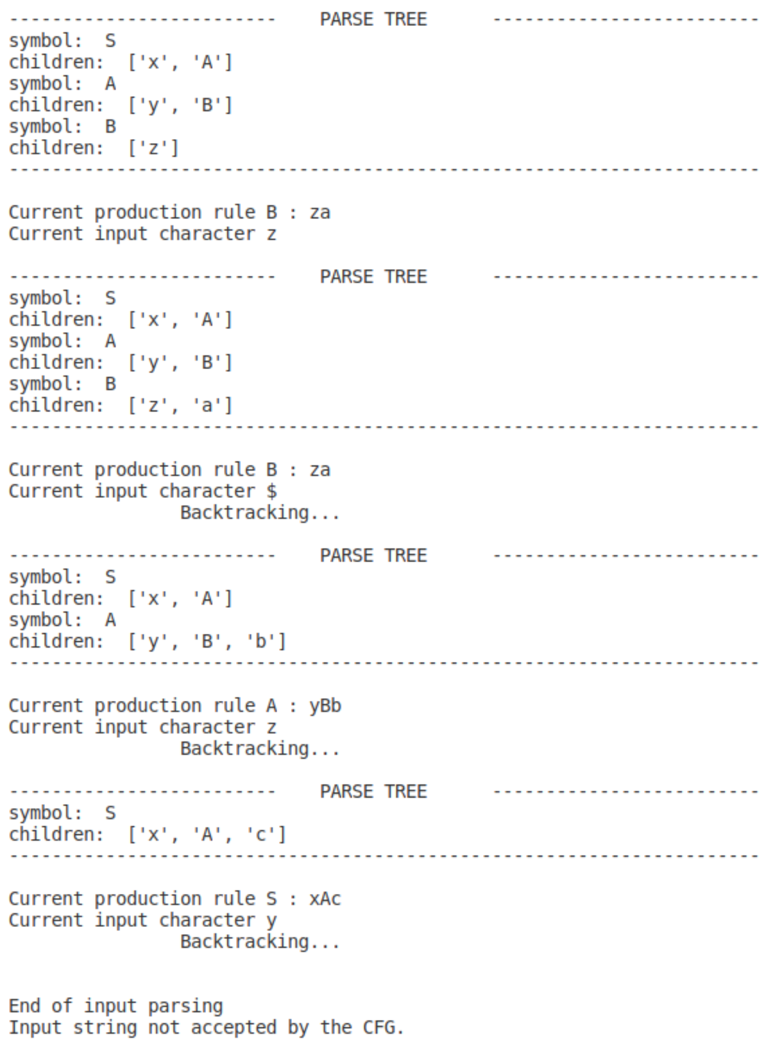


```
S ::= A|B|$
A ::= aCE|a
B ::= x
C ::= b
E ::= c
```

Output:



```
kj@kj:~/projects/elka/ECOTE/top-down-parser$ python3 main.py
Provided CFG file: /home/kj/projects/elka/ECOTE/top-down-parser/tests/correct_CFG/test4.txt
Enter input string: xyz$

Provided contex-free grammar:
{'S': ['xAc'], 'A': ['yBb'], 'B': ['za']}

Provided input string: xyz$
-----------------------------------
CFG: S

-------------------------    PARSE TREE     -------------------------
symbol:  S
children:  ['x']
---------------------------------------------------------------------

Current production rule S : xAc
Current input character x

-------------------------    PARSE TREE     -------------------------
symbol:  S
children:  ['x', 'A']
---------------------------------------------------------------------

Current production rule S : xAc
Current input character y
-----------------------------------
CFG: A

-------------------------    PARSE TREE     -------------------------
symbol:  S
children:  ['x', 'A']
symbol:  A
children:  ['y']
---------------------------------------------------------------------

Current production rule A : yBb
Current input character y

-------------------------    PARSE TREE     -------------------------
symbol:  S
children:  ['x', 'A']
symbol:  A
children:  ['y', 'B']
---------------------------------------------------------------------

Current production rule A : yBb
Current input character z
-----------------------------------
CFG: B
```

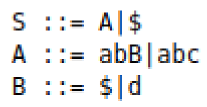


```
-------------------------    PARSE TREE     -------------------------
symbol:  S
children:  ['x', 'A']
symbol:  A
children:  ['y', 'B']
symbol:  B
children:  ['z']
----------------------------------------------------------------------

Current production rule B : za
Current input character z

-------------------------    PARSE TREE     -------------------------
symbol:  S
children:  ['x', 'A']
symbol:  A
children:  ['y', 'B']
symbol:  B
children:  ['z', 'a']
----------------------------------------------------------------------

Current production rule B : za
Current input character $
                Backtracking...

-------------------------    PARSE TREE     -------------------------
symbol:  S
children:  ['x', 'A']
symbol:  A
children:  ['y', 'B', 'b']
----------------------------------------------------------------------

Current production rule A : yBb
Current input character z
                Backtracking...

-------------------------    PARSE TREE     -------------------------
symbol:  S
children:  ['x', 'A', 'c']
----------------------------------------------------------------------

Current production rule S : xAc
Current input character y
                Backtracking...


End of input parsing
Input string not accepted by the CFG.
```

- text1.txt

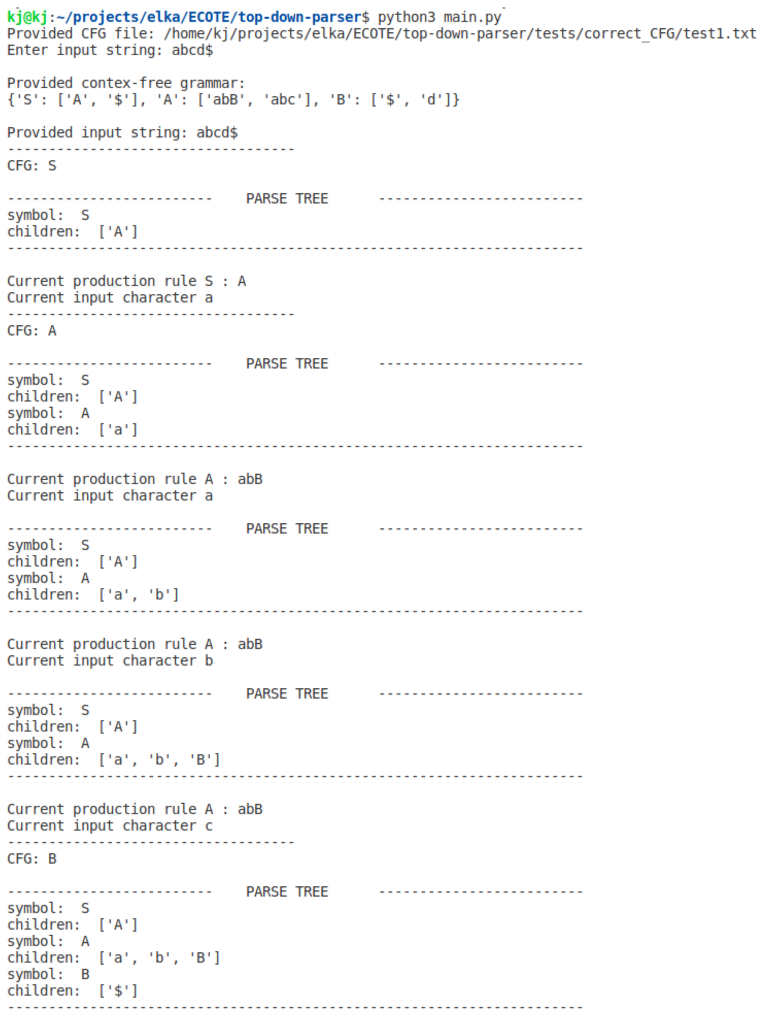
Input string: **abcd$**

CFG:

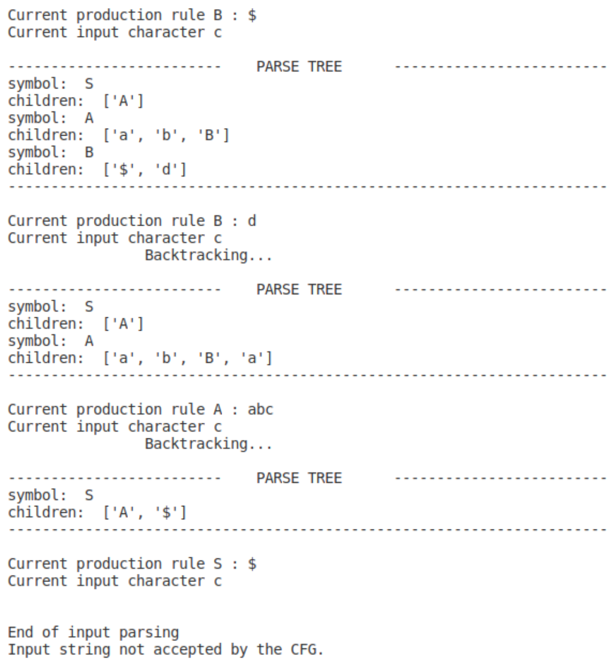


```
S ::= A|$
A ::= abB|abc
B ::= $|d
```

Output:



```
kj@kj:~/projects/elka/ECOTE/top-down-parser$ python3 main.py
Provided CFG file: /home/kj/projects/elka/ECOTE/top-down-parser/tests/correct_CFG/test1.txt
Enter input string: abcd$

Provided contex-free grammar:
{'S': ['A', '$'], 'A': ['abB', 'abc'], 'B': ['$', 'd']}

Provided input string: abcd$
-----------------------------------
CFG: S

-------------------------    PARSE TREE      -------------------------
symbol:  S
children:  ['A']
----------------------------------------------------------------------

Current production rule S : A
Current input character a
-----------------------------------
CFG: A

-------------------------    PARSE TREE      -------------------------
symbol:  S
children:  ['A']
symbol:  A
children:  ['a']
----------------------------------------------------------------------

Current production rule A : abB
Current input character a

-------------------------    PARSE TREE      -------------------------
symbol:  S
children:  ['A']
symbol:  A
children:  ['a', 'b']
----------------------------------------------------------------------

Current production rule A : abB
Current input character b

-------------------------    PARSE TREE      -------------------------
symbol:  S
children:  ['A']
symbol:  A
children:  ['a', 'b', 'B']
----------------------------------------------------------------------

Current production rule A : abB
Current input character c
-----------------------------------
CFG: B

-------------------------    PARSE TREE      -------------------------
symbol:  S
children:  ['A']
symbol:  A
children:  ['a', 'b', 'B']
symbol:  B
children:  ['$']
----------------------------------------------------------------------
```



```
Current production rule B : $
Current input character c

-------------------------    PARSE TREE      -------------------------
symbol:  S
children:  ['A']
symbol:  A
children:  ['a', 'b', 'B']
symbol:  B
children:  ['$', 'd']
----------------------------------------------------------------------

Current production rule B : d
Current input character c
                Backtracking...

-------------------------    PARSE TREE      -------------------------
symbol:  S
children:  ['A']
symbol:  A
children:  ['a', 'b', 'B', 'a']
----------------------------------------------------------------------

Current production rule A : abc
Current input character c
                Backtracking...

-------------------------    PARSE TREE      -------------------------
symbol:  S
children:  ['A', '$']
----------------------------------------------------------------------

Current production rule S : $
Current input character c


End of input parsing
Input string not accepted by the CFG.
```