

Ex 3: Recursive descent parser

$S \rightarrow aAb$

$A \rightarrow bA \mid aB$

gen acacb

$B \rightarrow c$

`lex()`

`S()`

```
{ if (nextToken == 'a')
  {
    lex()
    A()
    lex()
    if (nextToken == 'b') Acc.
  }
else error
}
```

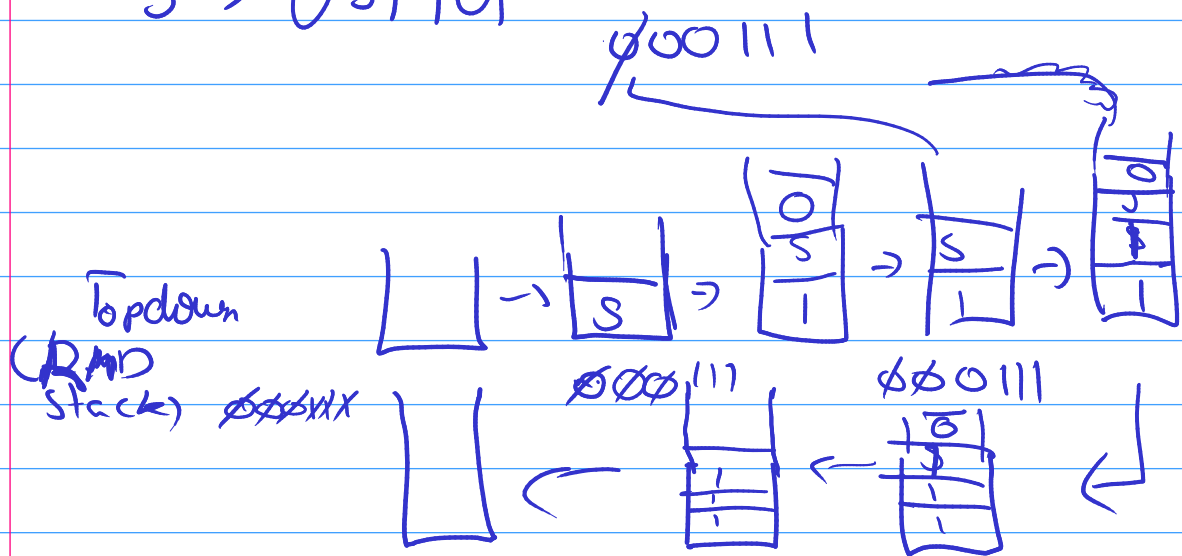
`A()`

```
{
  if (nextToken == b)
  {
    lex()
    A()
  }
  else if (nextToken == a)
  {
    lex()
    B()
  }
  else error
}
```

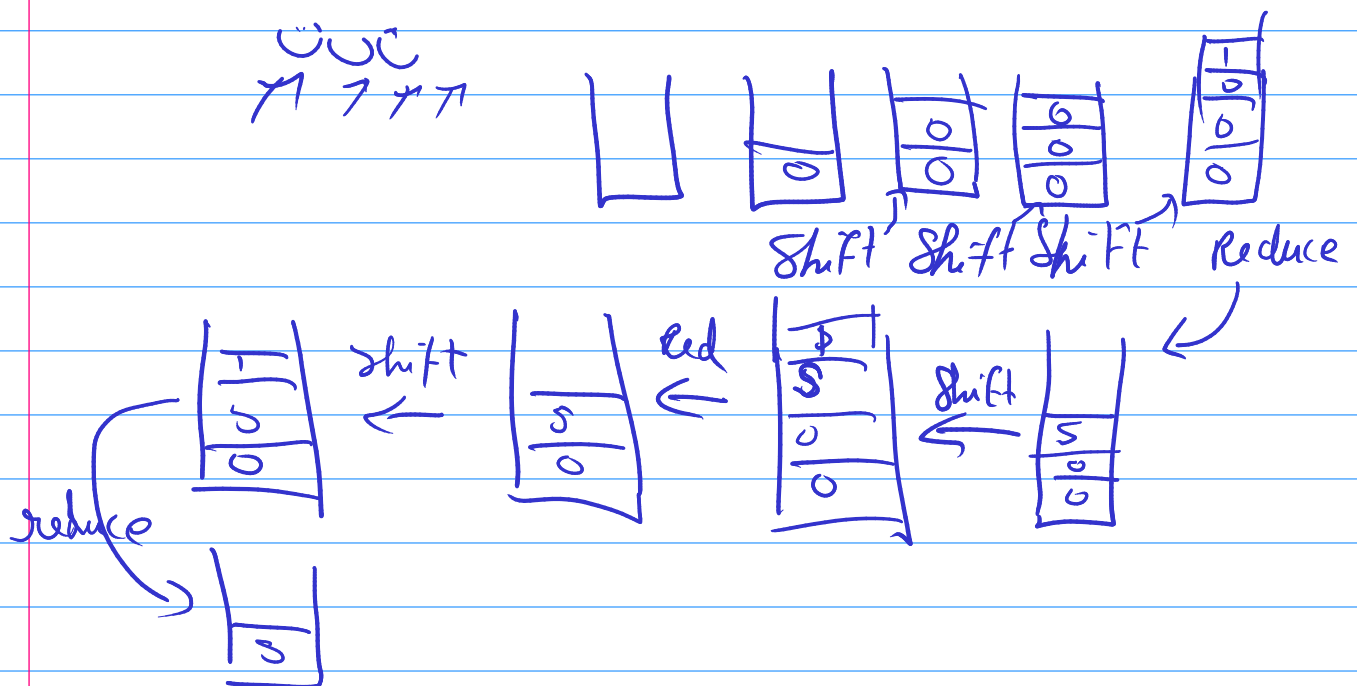
`B()`

```
{
  if (nextToken == 'c')
    get
  else
    Error
}
```

$S \rightarrow 0S1|01$



Bottomup (LRD)



$$S \rightarrow SS^+ | SS^* | a$$

aaa*a++

Top down
Bottom up,

show stack

The top screenshot shows a Google Meet window with a PowerPoint presentation. The slide is titled "EXERCISE 1" and contains the following text:

Give a CFG that generates sentences as composed of noun- and verb-phrases:
 This should generate sentences like:
 The cat eats the egg
 The dog kills the cat
 The bread eats the cat (which is legal but meaningless)

Noun phrases : the cat the dog the bread
 Verbs phrases: kills the cat eats the egg

The bottom screenshot shows a Google Meet window with a PowerPoint presentation. The slide is titled "Exercise 2" and contains the following text:

Now eliminate Left recursion from the following Grammar.
 $S \rightarrow SabA | Sc$
 $S \rightarrow a$
 $S \rightarrow b$
 $A \rightarrow cA | c$

Can generate: $S \rightarrow SabA \rightarrow ScabA \rightarrow bcabA \rightarrow bcabc$

(1)

$$S \rightarrow NP VP$$

$$NP \rightarrow the N'$$

$$VP \rightarrow VNP$$

$$N \rightarrow cat | dog | bread | egg$$

$$V \rightarrow kills | eats$$

Meeting details

REC R Gururaj is presenting

SAI SATVIK VUPPALA

SARWADNYA NIVRUTTI M...

R Gururaj

RUDRAJIT KARGUPTA

SHIVANG SINGH

Meeting details

VARUN PARTHASARATHY 8:01 AM
Mic is not working properly sir

SARWADNYA NIVRUTTI MUTKULE 8:05 AM
S->ABA
A->the cat | the dog | the egg | the bread
B->eats | kills

SAI SATVIK VUPPALA 8:06 AM
Kills the cat shld be considered as one?

Click to add notes

Turn on captions

R Gururaj is presenting

Send a message to everyone

Left recursion

$E \rightarrow E + T$
Creates problems. This is known as **Left Recursion**.

Whenever we have production of the form-
 $A \rightarrow A\alpha_1, A \rightarrow A\alpha_2, \dots, A \rightarrow A\alpha_n$ and
 $A \rightarrow \beta_1, A \rightarrow \beta_2, \dots, A \rightarrow \beta_m$

Replace these by
 $A \rightarrow \beta_1 A', A \rightarrow \beta_2 A', \dots, A \rightarrow \beta_m A'$ and
 $A' \rightarrow \alpha_1 A', A' \rightarrow \alpha_2 A', \dots, A' \rightarrow \alpha_n A'$ and
 $A' \rightarrow e$

Meeting details

REC R Gururaj is presenting

SAI SATVIK VUPPALA

SARWADNYA NIVRUTTI M...

R Gururaj

RUDRAJIT KARGUPTA

SHIVANG SINGH

Meeting details

VARUN PARTHASARATHY 8:01 AM
Mic is not working properly sir

SARWADNYA NIVRUTTI MUTKULE 8:05 AM
S->ABA
A->the cat | the dog | the egg | the bread
B->eats | kills

SAI SATVIK VUPPALA 8:06 AM
Kills the cat shld be considered as one?

Click to add text

Exercise 2:
Now eliminate Left recursion from the following Grammar.
 $S \rightarrow SabA | Sc$
 $S \rightarrow a$
 $S \rightarrow b$
 $A \rightarrow cA | c$
Can generate: $S \rightarrow SabA \rightarrow ScabA \rightarrow bcabA \rightarrow bcabc$
Whenever we have production of the form-
 $A \rightarrow A\alpha_1, A \rightarrow A\alpha_2, \dots, A \rightarrow A\alpha_n$ and
 $A \rightarrow \beta_1, A \rightarrow \beta_2, \dots, A \rightarrow \beta_m$

Replace these by
 $A \rightarrow \beta_1 A', A \rightarrow \beta_2 A', \dots, A \rightarrow \beta_m A'$ and
 $A' \rightarrow \alpha_1 A', A' \rightarrow \alpha_2 A', \dots, A' \rightarrow \alpha_n A'$ and
 $A' \rightarrow e$