

Derivation tree: top-down vs. bottom-up construction

grammar:

```
p0: P ::= D
P1: D ::= FF
P2: D ::= FB
P3: FF ::= 'fun' FI '(' Ps ')' 'fwd'
P4: FB ::= 'fun' FI '(' Ps ')' B
P5: Ps ::= Ps PI
P6: Ps :=
p7: B ::= '{' '}'
p8: FI ::= Id
p9: PI ::= Id
```

top-down:

bottom-up:

P



fun Id (Id Id) fwd

accepted and next input

fun Id (Id Id) fwd

accepted and next input

Derivation tree: top-down vs. bottom-up construction

grammar:

```
p0: P ::= D
P1: D ::= FF
P2: D ::= FB
P3: FF ::= 'fun' FI '(' Ps ')' 'fwd'
P4: FB ::= 'fun' FI '(' Ps ')' B
P5: Ps ::= Ps PI
P6: Ps :=
p7: B ::= '{' '}'
p8: FI ::= Id
p9: PI ::= Id
```

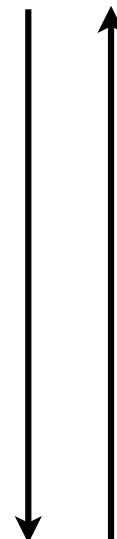
top-down:

bottom-up:

p0
P
D

fun Id (Id Id) fwd

accepted and next input



fun Id (Id Id) fwd

accepted and next input

Derivation tree: top-down vs. bottom-up construction

grammar:

```
p0: P ::= D
P1: D ::= FF
P2: D ::= FB
P3: FF ::= 'fun' FI '(' Ps ')' 'fwd'
P4: FB ::= 'fun' FI '(' Ps ')' B
P5: Ps ::= Ps PI
P6: Ps :=
p7: B ::= '{' '}'
p8: FI ::= Id
p9: PI ::= Id
```

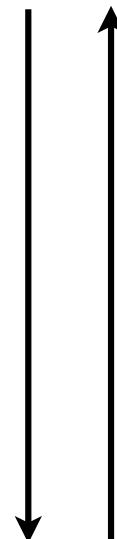
top-down:

bottom-up:

p0 P
p1 D
FF

fun Id (Id Id) fwd

accepted and next input



fun Id (Id Id) fwd

accepted and next input

Derivation tree: top-down vs. bottom-up construction

grammar:

```
p0: P ::= D
P1: D ::= FF
P2: D ::= FB
P3: FF ::= 'fun' FI '(' Ps ')' 'fwd'
P4: FB ::= 'fun' FI '(' Ps ')' B
P5: Ps ::= Ps PI
P6: Ps :=
p7: B ::= '{' '}'
p8: FI ::= Id
p9: PI ::= Id
```

top-down:

bottom-up:

p0 P
p1 D
p3 FF
p3 fun FI (Ps) fwd



fun Id (Id Id) fwd

accepted and next input

fun Id (Id Id) fwd

accepted and next input

Derivation tree: top-down vs. bottom-up construction

grammar:

```
p0: P ::= D
P1: D ::= FF
P2: D ::= FB
P3: FF ::= 'fun' FI '(' Ps ')' 'fwd'
P4: FB ::= 'fun' FI '(' Ps ')' B
P5: Ps ::= Ps PI
P6: Ps :=
p7: B ::= '{' '}'
p8: FI ::= Id
p9: PI ::= Id
```

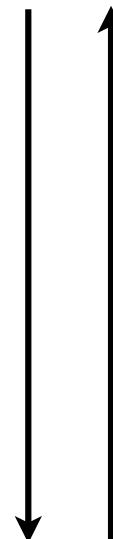
top-down:

bottom-up:

p0 P
p1 D
p3 FF
p8 fun FI (Ps) fwd
 Id

fun Id (Id Id) fwd

accepted and next input



fun Id (Id Id) fwd

accepted and next input

Derivation tree: top-down vs. bottom-up construction

grammar:

```
p0: P ::= D
P1: D ::= FF
P2: D ::= FB
P3: FF ::= 'fun' FI '(' Ps ')' 'fwd'
P4: FB ::= 'fun' FI '(' Ps ')' B
P5: Ps ::= Ps PI
P6: Ps :=
p7: B ::= '{' '}'
p8: FI ::= Id
p9: PI ::= Id
```

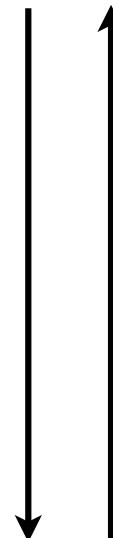
top-down:

bottom-up:

p0 P
p1 D
FF
p3 fun FI (Ps
)
p8 Id

fun Id (Id Id) fwd

accepted and next input



fun Id (Id Id) fwd

accepted and next input

Derivation tree: top-down vs. bottom-up construction

grammar:

```
p0: P ::= D
P1: D ::= FF
P2: D ::= FB
P3: FF ::= 'fun' FI '(' Ps ')' 'fwd'
P4: FB ::= 'fun' FI '(' Ps ')' B
P5: Ps ::= Ps PI
P6: Ps :=
p7: B ::= '{' '}'
p8: FI ::= Id
p9: PI ::= Id
```

top-down:

bottom-up:

P
D
FF
fun FI (Ps) fwd
Id
Ps PI

fun Id (Id Id) fwd

accepted and next input



fun Id (Id Id) fwd

accepted and next input

Derivation tree: top-down vs. bottom-up construction

grammar:

```
p0: P ::= D
P1: D ::= FF
P2: D ::= FB
P3: FF ::= 'fun' FI '(' Ps ')' 'fwd'
P4: FB ::= 'fun' FI '(' Ps ')' B
P5: Ps ::= Ps PI
P6: Ps :=
p7: B ::= '{' '}'
p8: FI ::= Id
p9: PI ::= Id
```

top-down:

bottom-up:

P
p0 D
p1 FF
p3 fun FI (Ps) fwd
p8 Id
p5 Ps PI
p5 Ps PI

fun Id (Id Id) fwd

accepted and next input



fun Id (Id Id) fwd

accepted and next input

Derivation tree: top-down vs. bottom-up construction

grammar:

```
p0: P ::= D
P1: D ::= FF
P2: D ::= FB
P3: FF ::= 'fun' FI '(' Ps ')' 'fwd'
P4: FB ::= 'fun' FI '(' Ps ')' B
P5: Ps ::= Ps PI
P6: Ps :=
p7: B ::= '{' '}'
p8: FI ::= Id
p9: PI ::= Id
```

top-down:

bottom-up:

P
p0 D
p1 FF
p3 fun FI (Ps) fwd
p8 Id
p5 Ps PI
p5 Ps PI
p6

fun Id (Id Id) fwd

accepted and next input



fun Id (Id Id) fwd

accepted and next input

Derivation tree: top-down vs. bottom-up construction

grammar:

```
p0: P ::= D
P1: D ::= FF
P2: D ::= FB
P3: FF ::= 'fun' FI '(' Ps ')' 'fwd'
P4: FB ::= 'fun' FI '(' Ps ')' B
P5: Ps ::= Ps PI
P6: Ps :=
p7: B ::= '{' '}'
p8: FI ::= Id
p9: PI ::= Id
```

top-down:

bottom-up:

P
D
FF
fun FI (Ps) fwd
Id
Ps PI
Ps PI
Id

fun Id (Id Id) fwd

accepted and next input



fun Id (Id Id) fwd

accepted and next input

Derivation tree: top-down vs. bottom-up construction

grammar:

```

p0: P ::= D
P1: D ::= FF
P2: D ::= FB
P3: FF ::= 'fun' FI '(' Ps ')' 'fwd'
P4: FB ::= 'fun' FI '(' Ps ')' B
P5: Ps ::= Ps PI
P6: Ps :=
p7: B ::= '{' '}'
p8: FI ::= Id
p9: PI ::= Id

```

top-down:

bottom-up:

p0	P
p1	D
p3	FF
p3	fun FI (Ps) fwd
p8	Id
p5	Ps PI
p5	Ps PI
p6	
p9	Id
p9	Id
	fun Id (Id Id) fwd

accepted and next input



fun Id (Id Id) fwd

accepted and next input

Derivation tree: top-down vs. bottom-up construction

grammar:

```
p0: P ::= D
P1: D ::= FF
P2: D ::= FB
P3: FF ::= 'fun' FI '(' Ps ')' 'fwd'
P4: FB ::= 'fun' FI '(' Ps ')' B
P5: Ps ::= Ps PI
P6: Ps :=
p7: B ::= '{' '}'
p8: FI ::= Id
p9: PI ::= Id
```

top-down:

bottom-up:

P
p0 D
p1 FF
p3 fun FI (Ps) fwd
p8 Id
p5 Ps PI
p5 Ps PI
p6
p9 Id
p9 Id
fun Id (Id Id) fwd



accepted and next input

fun Id (Id Id) fwd

accepted and next input

Derivation tree: top-down vs. bottom-up construction

grammar:

```

p0: P ::= D
P1: D ::= FF
P2: D ::= FB
P3: FF ::= 'fun' FI '(' Ps ')' 'fwd'
P4: FB ::= 'fun' FI '(' Ps ')' B
P5: Ps ::= Ps PI
P6: Ps :=
p7: B ::= '{' '}'
p8: FI ::= Id
p9: PI ::= Id

```

top-down:

bottom-up:

```

p0 P
p1 D
p3 FF
p3 fun FI ( Ps      ) fwd
p8   Id
p5     Ps    PI
p5     Ps  PI
p6
p9
p9   Id
p9     Id
fun Id (   Id Id) fwd

```

accepted and next input

fun Id (Id Id) fwd

accepted and next input



Derivation tree: top-down vs. bottom-up construction

grammar:

```

p0: P ::= D
P1: D ::= FF
P2: D ::= FB
P3: FF ::= 'fun' FI '(' Ps ')' 'fwd'
P4: FB ::= 'fun' FI '(' Ps ')' B
P5: Ps ::= Ps PI
P6: Ps :=
p7: B ::= '{' '}'
p8: FI ::= Id
p9: PI ::= Id

```

top-down:

```

p0 P
p1 D
p3 FF
p3 fun FI ( Ps      ) fwd
p8   Id
p5     Ps    PI
p5     Ps  PI
p6
p9
p9   Id
           Id
fun Id (   Id Id) fwd

```

accepted and next input

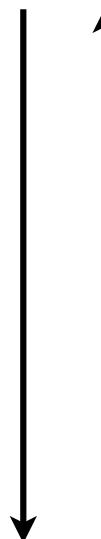
bottom-up:

```

fun
fun Id (   Id Id) fwd

```

accepted and next input



Derivation tree: top-down vs. bottom-up construction

grammar:

```

p0: P ::= D
P1: D ::= FF
P2: D ::= FB
P3: FF ::= 'fun' FI '(' Ps ')' 'fwd'
P4: FB ::= 'fun' FI '(' Ps ')' B
P5: Ps ::= Ps PI
P6: Ps :=
p7: B ::= '{' '}'
p8: FI ::= Id
p9: PI ::= Id

```

top-down:

```

p0 P
p1 D
p3 FF
p3 fun FI ( Ps      ) fwd
p8   Id
p5   Ps     PI
p5   Ps PI
p6
p9
p9   Id
           Id
fun Id (   Id Id) fwd

```

accepted and next input

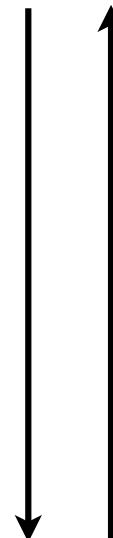
bottom-up:

```

fun id
fun Id (   Id Id) fwd

```

accepted and next input



Derivation tree: top-down vs. bottom-up construction

grammar:

```

p0: P ::= D
P1: D ::= FF
P2: D ::= FB
P3: FF ::= 'fun' FI '(' Ps ')' 'fwd'
P4: FB ::= 'fun' FI '(' Ps ')' B
P5: Ps ::= Ps PI
P6: Ps :=
p7: B ::= '{' '}'
p8: FI ::= Id
p9: PI ::= Id

```

top-down:

```

p0 P
p1 D
p3 FF
p3 fun FI ( Ps      ) fwd
p8   Id
p5     Ps      PI
p5     Ps  PI
p6
p9
p9   Id
p9     Id
fun Id (   Id Id) fwd

```

accepted and next input

bottom-up:

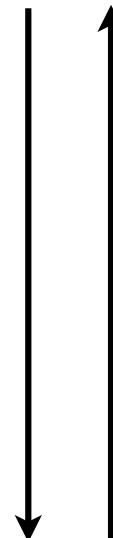
```

FI
fun id
fun Id (   Id Id) fwd

```

p8

accepted and next input



Derivation tree: top-down vs. bottom-up construction

grammar:

```

p0: P ::= D
P1: D ::= FF
P2: D ::= FB
P3: FF ::= 'fun' FI '(' Ps ')' 'fwd'
P4: FB ::= 'fun' FI '(' Ps ')' B
P5: Ps ::= Ps PI
P6: Ps :=
p7: B ::= '{' '}'
p8: FI ::= Id
p9: PI ::= Id

```

top-down:

```

p0 P
p1 D
p3 FF
p3 fun FI ( Ps      ) fwd
p8   Id
p5     Ps      PI
p5     Ps  PI
p6
p9
p9   Id
p9     Id
fun Id (   Id Id) fwd

```

accepted and next input

bottom-up:

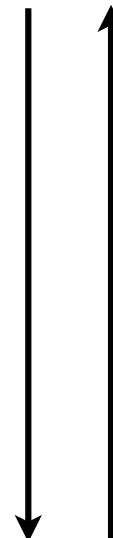
```

FI (
fun id
fun Id (   Id Id) fwd

```

p8

accepted and next input



Derivation tree: top-down vs. bottom-up construction

grammar:

```

p0: P ::= D
P1: D ::= FF
P2: D ::= FB
P3: FF ::= 'fun' FI '(' Ps ')' 'fwd'
P4: FB ::= 'fun' FI '(' Ps ')' B
P5: Ps ::= Ps PI
P6: Ps :=
p7: B ::= '{' '}'
p8: FI ::= Id
p9: PI ::= Id

```

top-down:

```

p0 P
p1 D
p3 FF
p3 fun FI ( Ps      ) fwd
p8   Id
p5     Ps      PI
p5     Ps  PI
p6
p9       Id
p9       Id
fun Id (   Id Id) fwd

```

accepted and next input

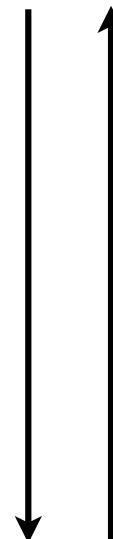
bottom-up:

```

Ps
p6
p8
FI (
fun id
fun Id (   Id Id) fwd

```

accepted and next input



Derivation tree: top-down vs. bottom-up construction

grammar:

```

p0: P ::= D
P1: D ::= FF
P2: D ::= FB
P3: FF ::= 'fun' FI '(' Ps ')' 'fwd'
P4: FB ::= 'fun' FI '(' Ps ')' B
P5: Ps ::= Ps PI
P6: Ps :=
p7: B ::= '{' '}'
p8: FI ::= Id
p9: PI ::= Id

```

top-down:

```

p0 P
p1 D
p3 FF
p3 fun FI ( Ps      ) fwd
p8   Id
p5     Ps      PI
p5     Ps  PI
p6
p9       Id
p9       Id
fun Id (   Id Id) fwd

```

accepted and next input

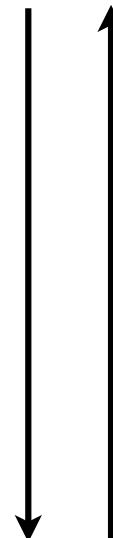
bottom-up:

```

Ps Id
FI (
fun id
fun Id (   Id Id) fwd

```

accepted and next input



Derivation tree: top-down vs. bottom-up construction

grammar:

```

p0: P ::= D
P1: D ::= FF
P2: D ::= FB
P3: FF ::= 'fun' FI '(' Ps ')' 'fwd'
P4: FB ::= 'fun' FI '(' Ps ')' B
P5: Ps ::= Ps PI
P6: Ps :=
p7: B ::= '{' '}'
p8: FI ::= Id
p9: PI ::= Id

```

top-down:

```

p0 P
p1 D
p3 FF
p3 fun FI ( Ps      ) fwd
p8   Id
p5     Ps      PI
p5     Ps  PI
p6
p9       Id
p9       Id
fun Id (   Id Id) fwd

```

accepted and next input

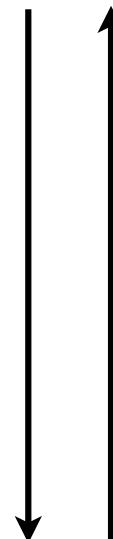
bottom-up:

```

PI
Ps Id
FI (
fun id
fun Id (   Id Id) fwd

```

accepted and next input



Derivation tree: top-down vs. bottom-up construction

grammar:

```

p0: P ::= D
P1: D ::= FF
P2: D ::= FB
P3: FF ::= 'fun' FI '(' Ps ')' 'fwd'
P4: FB ::= 'fun' FI '(' Ps ')' B
P5: Ps ::= Ps PI
P6: Ps :=
p7: B ::= '{' '}'
p8: FI ::= Id
p9: PI ::= Id

```

top-down:

```

p0 P
p1 D
p3 FF
p3 fun FI ( Ps      ) fwd
p8   Id
p5     Ps      PI
p5     Ps  PI
p6
p9       Id
p9       Id
fun Id (   Id Id) fwd

```

accepted and next input

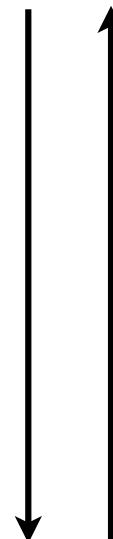
bottom-up:

```

Ps
PI
Ps Id
FI (
fun id
fun Id (   Id Id) fwd

```

accepted and next input



p5
p9
p6
p8

Derivation tree: top-down vs. bottom-up construction

grammar:

```

p0: P ::= D
P1: D ::= FF
P2: D ::= FB
P3: FF ::= 'fun' FI '(' Ps ')' 'fwd'
P4: FB ::= 'fun' FI '(' Ps ')' B
P5: Ps ::= Ps PI
P6: Ps :=
p7: B ::= '{' '}'
p8: FI ::= Id
p9: PI ::= Id

```

top-down:

```

p0 P
p1 D
p3 FF
p3 fun FI ( Ps      ) fwd
p8   Id
p5     Ps      PI
p5       Ps PI
p6
p9     Id
p9       Id
fun Id (   Id Id) fwd

```

accepted and next input

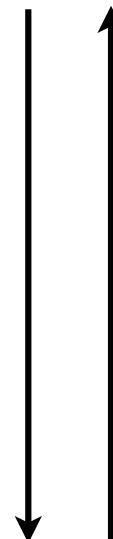
bottom-up:

```

Ps      Id
      PI
      Ps Id
        FI (
        fun id
fun Id (   Id Id) fwd

```

accepted and next input



Derivation tree: top-down vs. bottom-up construction

grammar:

```

p0: P ::= D
P1: D ::= FF
P2: D ::= FB
P3: FF ::= 'fun' FI '(' Ps ')' 'fwd'
P4: FB ::= 'fun' FI '(' Ps ')' B
P5: Ps ::= Ps PI
P6: Ps :=
p7: B ::= '{' '}'
p8: FI ::= Id
p9: PI ::= Id

```

top-down:

```

p0 P
p1 D
p3 FF
p3 fun FI ( Ps      ) fwd
p8   Id
p5     Ps    PI
p5     Ps  PI
p6
p9       Id
p9       Id
fun Id (   Id Id) fwd

```

accepted and next input

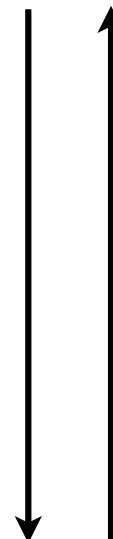
bottom-up:

```

PI
Ps   PI
Ps   Id
p9
p5
p9
p6
p8
FI (
fun id
fun Id (   Id Id) fwd

```

accepted and next input



Derivation tree: top-down vs. bottom-up construction

grammar:

```

p0: P ::= D
P1: D ::= FF
P2: D ::= FB
P3: FF ::= 'fun' FI '(' Ps ')' 'fwd'
P4: FB ::= 'fun' FI '(' Ps ')' B
P5: Ps ::= Ps PI
P6: Ps :=
p7: B ::= '{' '}'
p8: FI ::= Id
p9: PI ::= Id

```

top-down:

```

p0 P
p1 D
p3 FF
p3 fun FI ( Ps      ) fwd
p8   Id
p5     Ps    PI
p5     Ps  PI
p6
p9       Id
p9       Id
fun Id (   Id Id) fwd

```

accepted and next input

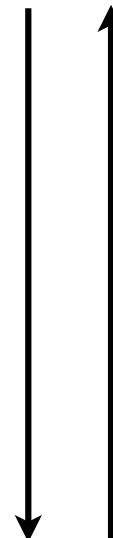
bottom-up:

```

Ps
Ps
Ps
Ps
Ps
Ps
PI
Id
PI
Id
PI
Id
PI
Id
FI (
fun id
fun Id (   Id Id) fwd

```

accepted and next input



Derivation tree: top-down vs. bottom-up construction

grammar:

```

p0: P ::= D
P1: D ::= FF
P2: D ::= FB
P3: FF ::= 'fun' FI '(' Ps ')' 'fwd'
P4: FB ::= 'fun' FI '(' Ps ')' B
P5: Ps ::= Ps PI
P6: Ps :=
p7: B ::= '{' '}'
p8: FI ::= Id
p9: PI ::= Id

```

top-down:

```

p0 P
p1 D
p3 FF
p3 fun FI ( Ps      ) fwd
p8   Id
p5     Ps    PI
p5     Ps  PI
p6
p9       Id
p9       Id
fun Id (   Id Id) fwd

```

accepted and next input

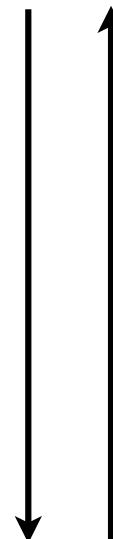
bottom-up:

```

Ps      )
p5
Ps    PI
p9
Ps  Id
p5
Ps  PI
p9
Ps Id
p6
p8
fun FI (
fun id
fun Id (   Id Id) fwd

```

accepted and next input



Derivation tree: top-down vs. bottom-up construction

grammar:

```

p0: P ::= D
P1: D ::= FF
P2: D ::= FB
P3: FF ::= 'fun' FI '(' Ps ')' 'fwd'
P4: FB ::= 'fun' FI '(' Ps ')' B
P5: Ps ::= Ps PI
P6: Ps :=
p7: B ::= '{' '}'
p8: FI ::= Id
p9: PI ::= Id

```

top-down:

```

p0 P
p1 D
p3 FF
p3 fun FI ( Ps      ) fwd
p8   Id
p5     Ps    PI
p5     Ps  PI
p6
p9       Id
p9       Id
fun Id (   Id Id) fwd

```

accepted and next input

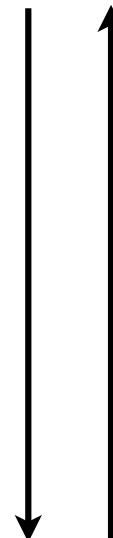
bottom-up:

```

Ps      ) fwd p5
        PI
Ps      Id p9
        PI
Ps  Id p5
        PI
        Ps Id p9
        FI (
        fun id p6
        fun id (   Id Id) fwd p8

```

accepted and next input



Derivation tree: top-down vs. bottom-up construction

grammar:

```

p0: P ::= D
P1: D ::= FF
P2: D ::= FB
P3: FF ::= 'fun' FI '(' Ps ')' 'fwd'
P4: FB ::= 'fun' FI '(' Ps ')' B
P5: Ps ::= Ps PI
P6: Ps :=
p7: B ::= '{' '}'
p8: FI ::= Id
p9: PI ::= Id

```

top-down:

```

p0 P
p1 D
FF
p3 fun FI ( Ps      ) fwd
p8   Id
p5     Ps    PI
p5     Ps  PI
p6
p9       Id
p9       Id
fun Id (   Id Id) fwd

```

accepted and next input

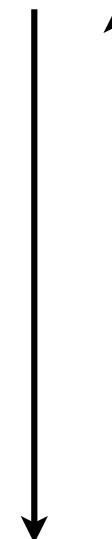
bottom-up:

```

FF
Ps      ) fwd p3
PI      p5
p9
Ps      Id p5
PI      p5
Ps  Id p9
p6
p8
fun id
fun Id (   Id Id) fwd

```

accepted and next input



Derivation tree: top-down vs. bottom-up construction

grammar:

```

p0: P ::= D
P1: D ::= FF
P2: D ::= FB
P3: FF ::= 'fun' FI '(' Ps ')' 'fwd'
P4: FB ::= 'fun' FI '(' Ps ')' B
P5: Ps ::= Ps PI
P6: Ps :=
p7: B ::= '{' '}'
p8: FI ::= Id
p9: PI ::= Id

```

top-down:

```

p0 P
p1 D
p1 FF
p3 fun FI ( Ps      ) fwd
p8   Id
p5     Ps    PI
p5     Ps  PI
p6
p9       Id
p9       Id
fun Id (   Id Id) fwd

```

accepted and next input

bottom-up:

```

D
FF
Ps      ) fwd
p1 p3 p5 p9 p5 p9 p6 p8
PI
Id
PI
Ps Id
FI (
fun id
fun Id (   Id Id) fwd

```

accepted and next input



Derivation tree: top-down vs. bottom-up construction

grammar:

```

p0: P ::= D
P1: D ::= FF
P2: D ::= FB
P3: FF ::= 'fun' FI '(' Ps ')' 'fwd'
P4: FB ::= 'fun' FI '(' Ps ')' B
P5: Ps ::= Ps PI
P6: Ps :=
p7: B ::= '{' '}'
p8: FI ::= Id
p9: PI ::= Id

```

top-down:

```

p0 P
p1 D
p1 FF
p3 fun FI ( Ps      ) fwd
p8     Id
p5       Ps   PI
p5       Ps PI
p6
p9         Id
p9         Id
fun Id (   Id Id) fwd

```

accepted and next input

bottom-up:

```

P
D
FF
Ps      ) fwd
PI
Ps   Id
PI
Ps Id
FI (
fun id
fun Id (   Id Id) fwd

```

accepted and next input

Derivation tree: top-down vs. bottom-up construction

grammar:

```

p0: P ::= D
P1: D ::= FF
P2: D ::= FB
P3: FF ::= 'fun' FI '(' Ps ')' 'fwd'
P4: FB ::= 'fun' FI '(' Ps ')' B
P5: Ps ::= Ps PI
P6: Ps ::=
p7: B ::= '{' '}'
p8: FI ::= Id
p9: PI ::= Id

```

top-down:

```

p0 P
p1 D
p2 FF
p3 fun FI ( Ps      ) fwd
p4     Id
p5       Ps   PI
p6       Ps PI
p7         PI
p8         Ps Id
p9           ID
p9             ID
fun Id (    ID ID) fwd

```

accepted and next input

bottom-up:

```

P
D
FF
Ps      ) fwd
PI
Ps Id
PI
Ps Id
FI (
fun id
fun Id (    ID ID) fwd

```

accepted and next input

right-
derivation



p0
p1
p3
p5
p9
p5
p9
p6
p8

↓