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:

P

```
fun Id (    Id Id) fwd
```

accepted and next input

bottom-up:

```
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
```

$$\downarrow \quad \uparrow$$

```
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
```

```
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:

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




    fun Id (    Id Id) fwd
```

accepted and next input

bottom-up:

```
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:

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


    fun Id (     Id Id) fwd
```

accepted and next input

bottom-up:

```
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:

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

```
fun Id (     Id Id) fwd
```

accepted and next input

bottom-up:

```
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:

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

```
fun Id (    Id Id) fwd
```

accepted and next input

bottom-up:

```
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:

```
    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

bottom-up:

```
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:

```
    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

bottom-up:

```
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:

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

    fun Id (     Id Id) fwd
```

accepted and next input

bottom-up:

```
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:

```
     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

bottom-up:

```
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:

```
    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

bottom-up:

```
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:

```
     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

bottom-up:

```
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:

```
    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

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:

```
    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

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:

```
    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

bottom-up:

```
          FI
fun id                            p8
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:

```
    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

bottom-up:

```
         FI (
fun id                    p8
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:

```
     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

bottom-up:

```
         Ps
     FI (                p6
fun id                   p8
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:

```
     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

bottom-up:

```
                 Ps Id
                                 p6
            FI (
                                 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:

```
    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

bottom-up:

```
                PI
             Ps Id          p9
         FI (               p6
     fun id                 p8
     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:

```
       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
```

bottom-up:

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

accepted and next input

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:

```
    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

bottom-up:

```
                 Ps    Id
                                   p5
                    PI
                 Ps Id             p9
                                   p6
            FI (
        fun id                     p8
        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:

```
    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

bottom-up:

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

accepted and next input

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

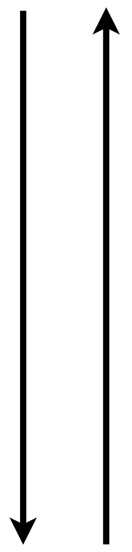
grammar:

```
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:

```
     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



bottom-up:

```
         Ps                 p5
               PI           p9
         Ps    Id           p5
            PI              p9
         Ps Id              p6
    FI (                    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:

```
    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

bottom-up:

```
         Ps      )
                          p5
               PI
                          p9
         Ps    Id
                          p5
            PI
                          p9
         Ps Id
                          p6
    FI (
                          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:

```
    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

bottom-up:

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

      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:

```
     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
```

bottom-up:

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

accepted and next input

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:

```
     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

bottom-up:

```
D
                         p1
FF
                         p3
         Ps      ) fwd
                         p5
               PI
                         p9
         Ps    Id
                         p5
            PI
                         p9
         Ps Id
                         p6
    FI  (
                         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:

```
    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
```

bottom-up:

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

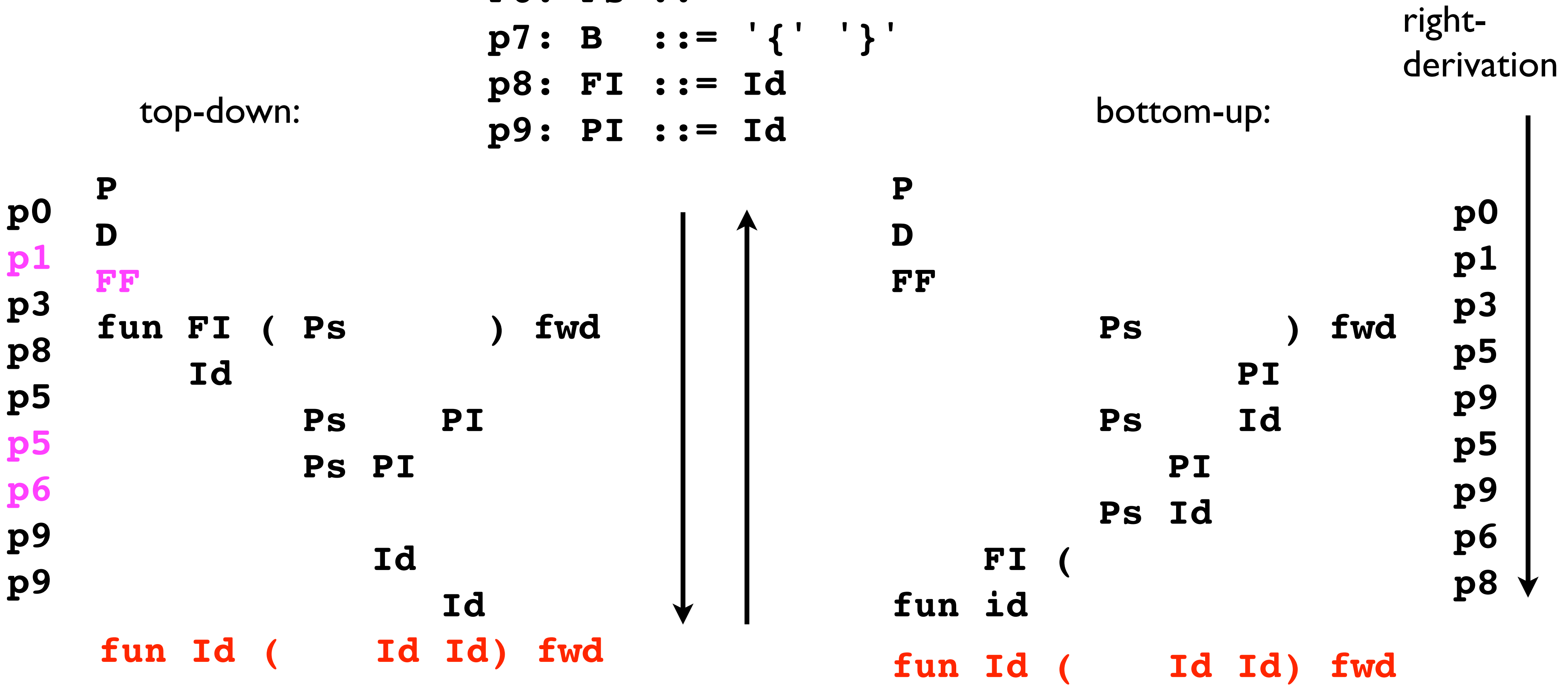
accepted and next input

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:

```
     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
```

bottom-up:

```
P                                 right-
D                                 derivation
FF
                Ps        ) fwd   p0
                       PI         p1
                Ps     Id         p3
                    PI            p5
                Ps  Id            p9
      FI (                        p5
fun   id                          p9
                                  p6
                                  p8
fun   Id (          Id Id) fwd
```

accepted and next input

accepted and next input