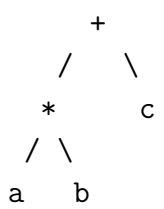
Linguagens de Programação

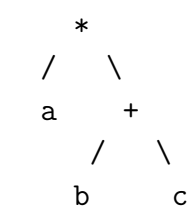
Gramáticas
– Soluções –

1. Árvores de sintaxe abstractas:

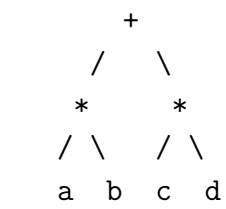
   (a)



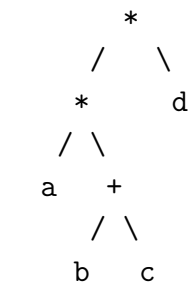
   (b)



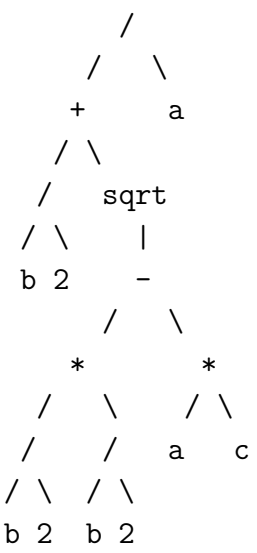
   (c)



   (d)



(e)



2. Prefixa:

   (a) $+ * a\ b\ c$

   (b) $*a + b\ c$

   (c) $+ * a\ b * c\ d$

   (d) $* * a + b\ c\ d$

   (e) $/ + /b\ 2\ sqrt - */b\ 2\ /b\ 2 * a\ c\ a$

3. Posfixa:

   (a) $a\ b *\ c+$

   (b) $a\ b\ c + *$

   (c) $a\ b * c\ d * +$

   (d) $a\ b\ c + * d *$

   (e) $b\ 2/\ b\ 2/\ b2/ * a\ c * -sqrt + a/$

4. Descrição da lista: num — elemento — separador/terminador — associatividade

   (a) 1+ — $< name >$ — , (sep) — associa à direita

   (b) 1+ — $< field >$ — ; (term) — associa à direita

   (c) 0+ — $< statement >$ — ; (term) — associa à direita

   (d) 1+ — $< factor >$ — * (sep) — associa à esquerda

(e) 1+ — $< name - list >:< type >$ — ; (term) — associa à direita

(f) 1+ — $< name >:< constant >$ — ; (term) — associa à esquerda

5. Gramática:

```
E ::= E || C | C
E ::= E && N | N
N ::= not N | A
A ::= true | false
```

(a)

```
E ::= E || C | C
E ::= E && N | N
N ::= not N | A
A ::= true | false | ( E )
```

(b)

```
I ::= E = I | E
E ::= E || C | C
E ::= E && N | N
N ::= not N | A
A ::= true | false | ( E )
```

6. Gramática:

```
E ::= E + T | E - T | T
T ::= T * N | T / N | N
N ::= N D | D
D ::= 0 | 1 | 2 ... | 9
```

(a)

```
E ::= E + T | E - T | T
T ::= T * N | T / N | N
F ::= N ^ F | N
N ::= N D | D
D ::= 0 | 1 | 2 ... | 9
```

(b)

```
E ::= E + T | E - T | T
T ::= T * N | T / N | N
F ::= P ^ F | P
P ::= N | ( E )
N ::= N D | D
D ::= 0 | 1 | 2 ... | 9
```

7. Dangling else

   (a) Porque a expressão `if E then if E then S else S` pode ser obtida através de duas derivações distintas.

   (b)

   ```
   S ::= CL | OP
   CL ::= if C then CL else CL
   OP ::= if C then OP | if C then CL | if C then CL else OP
   ```

   A última regra de `OP` permite ligar o `else` ao `if` solto que lhe está mais próximo. As restantes hipóteses não pertencem à gramática para proibir a ocorrência de um `else` que não esteja ligado ao `if` solto mais próximo (não permite que um `if` aberto exista entre um `else` e o seu `if`.