

```
(defun preorder (tree)
  (cond
    ((null tree) nil)

    ((atom tree)
     (print tree))

    (t
     (print (car tree))
     (mapcar #'preorder (cdr tree))
     )
    )
  )

(defun inorder (tree)
  (cond
    ((null tree) nil)

    ((atom tree)
     (print tree))

    (t
     (inorder (car (cdr tree)))
     (print (car tree))
     (mapcar #'inorder (cdr (cdr tree)))
     )
    )
  )

(defun postorder (tree)
  (cond
    ((null tree) nil)

    ((atom tree)
     (print tree))

    (t
     (mapcar #'postorder (cdr tree))
     (print (car tree))
     )
    )
  )
```

(preorder '(E (F (C) (J (K))) (B (D (G) (I)) (A (H))))) =>

E
F
C
J
K
B
D
G
I
A
H

(inorder '(E (F (C) (J (K))) (B (D (G) (I)) (A (H))))) =>

C
F
K
J
E
G
D
I
B
H
A

(postorder '(E (F (C) (J (K))) (B (D (G) (I)) (A (H))))) =>

C
K
J
F
G
I
D
H
A
B
E

The image shows a screenshot of a code editor with two identical windows side-by-side, illustrating the execution of a Common Lisp script. Both windows display the same code and output.

Code:

```
22 |   (t
23 |     (inorder (car (cdr tree)))
24 |     (print (car tree))
25 |     (mapcar #'inorder (cdr (cdr tree)))
26 |   )
27 )
28 )
29
30 (defun postorder (tree)
31   (cond
32     ((null tree) nil)
33     ((atom tree)
34       (print tree))
35       (t
36         (mapcar #'postorder (cdr tree))
37         (print (car tree)))
38       )
39     )
40   )
41 )
42 )
43
44
45 ;(preorder '(E (F (C) (J (K))) (B (D (G) (I)) (A (H)))))
46 ;(inorder '(E (F (C) (J (K))) (B (D (G) (I)) (A (H)))))
47 (postorder '(E (F (C) (J (K))) (B (D (G) (I)) (A (H)))))

19 |   ((atom tree)
20 |     (print tree))
21 |
22 |   (t
23 |     (inorder (car (cdr tree)))
24 |     (print (car tree))
25 |     (mapcar #'inorder (cdr (cdr tree)))
26 |   )
27 )
28 )
29
30 (defun postorder (tree)
31   (cond
32     ((null tree) nil)
33     ((atom tree)
34       (print tree))
35       (t
36         (mapcar #'postorder (cdr tree))
37         (print (car tree)))
38       )
39     )
40   )
41 )
42 )
43
44
45 ;(preorder '(E (F (C) (J (K))) (B (D (G) (I)) (A (H)))))
46 ;(inorder '(E (F (C) (J (K))) (B (D (G) (I)) (A (H)))))
47 (postorder '(E (F (C) (J (K))) (B (D (G) (I)) (A (H))))
```

Output:

C
K
J
F
G
I
D
H
A
B
E

STDIN

Output:

C
F
K
J
E
G
D
I
B
H
A

HelloWorld.lsp 44akfv8gn

```
19 |   ((atom tree)
20 |     (print tree)
21 |
22 |   (t
23 |     (inorder (car (cdr tree)))
24 |     (print (car tree))
25 |     (mapcar #'inorder (cdr (cdr tree)))
26 |   )
27 )
28 )
29
30 (defun postorder (tree)
31   (cond
32     ((null tree) nil)
33     ((atom tree)
34       (print tree))
35     (t
36       (mapcar #'postorder (cdr tree))
37       (print (car tree))
38     )
39   )
40 )
41 )
42
43
44
45 (preorder '(E (F (C) (J (K))) (B (D (G) (I)) (A (H)))))
46 ;(inorder '(E (F (C) (J (K))) (B (D (G) (I)) (A (H)))))
47 ;(postorder '(E (F (C) (J (K))) (B (D (G) (I)) (A (H)))))
```

AI

NEW

COMMONLISP ▾

RUN

STDIN

Output:
E
F
C
J
K
B
D
G
I
A
H