```
(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
```

```
    (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)))))
;(inorder '(E (F (C) (J (K))) (B (D (G) (I)) (A (H)))))
(postorder '(E (F (C) (J (K))) (B (D (G) (I)) (A (H)))))

```

Output:

```
C
K
J
F
G
I
D
H
A
B
E
```

Search

HelloWorld.lsp 44akfv8gn AI NEW COMMONLISP RUN

```
    ((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)))))
(inorder '(E (F (C) (J (K))) (B (D (G) (I)) (A (H)))))
;(postorder '(E (F (C) (J (K))) (B (D (G) (I)) (A (H)))))
```

STDIN

Output:

```
C
F
K
J
E
G
D
I
B
H
A
```


OneCompiler
Pricing
Learn
Code
Deploy
More
HelloWorld.lsp
44akfv8gn
AI
NEW
COMMONLISP
RUN
((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)))))
;(inorder '(E (F (C) (J (K))) (B (D (G) (I)) (A (H)))))
;(postorder '(E (F (C) (J (K))) (B (D (G) (I)) (A (H)))))
STDIN
Output:
E
F
C
J
K
B
D
G
I
A
H