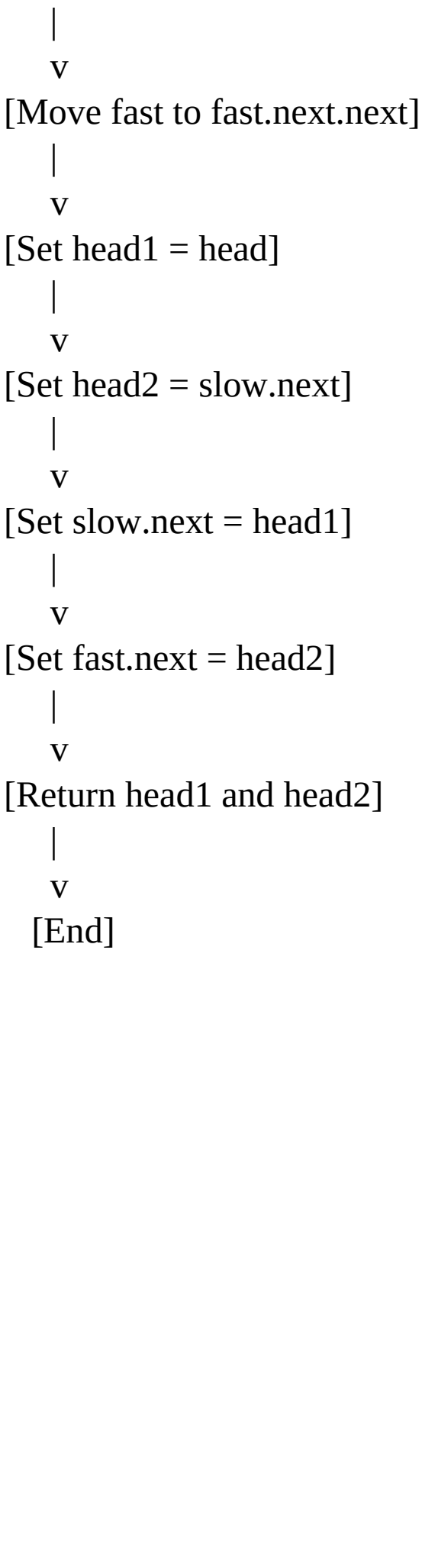
❖ Split Circular List:

- Explanation:
  The program splits a circular linked list into two halves by using two pointers, `slow` and `fast`, to find the midpoint of the list. Once the midpoint is identified, the circular links are broken to form two separate lists. Finally, the heads of both newly created lists are returned for further processing or use.
- Time Complexity and Space Complexity:
  - O(n) O(1)
- Flowchart:
  [Start]
  |
  v
  [Input head of circular linked list]
  |
  v
  [Is head null?] -- Yes --> [Return null for both halves]
  | No
  v
  [Set slow = head, fast = head]
  |
  v
  [While fast.next != head AND fast.next.next != head]
  |
  v
  [Move slow to slow.next]


[Move fast to fast.next.next]
[Set head1 = head]
[Set head2 = slow.next]
[Set slow.next = head1]
[Set fast.next = head2]
[Return head1 and head2]
[End]