Postorder 1: Recursive Solution
Filename: po_01_standard.cpp
Source: https://github.com/prithviraj-it20/postorderTraversal/blob/main/PostorderTraversal.cpp
Key OD: The integer sequence printed to stdout (Left -> Right -> Root)

Postorder 2: Iterative Postorder (Two Stacks)
Filename: po_02_iterative_2stack.cpp
Source:https://github.com/amanhex/Leetcode-Solutions/blob/main/145.%20Binary%20Tree%20Postorder%20Traversal.cpp
Key OD: The integer sequence printed to stdout (Left -> Right -> Root)

Postorder 3: Iterative Postorder (1-Stack Reverse Strategy)
Filename: po_03_iterative_reverse.cpp
Source: https://github.com/cruxrebels/InterviewBit/blob/master/Trees/PostorderTraversal.cpp
Key OD: The integer sequence printed to stdout (Left -> Right -> Root)

Postorder 4: Morris Postorder Traversal
Filename: po_04_morris.cpp
Source:https://github.com/liuyubobobo/Play-with-Algorithms/blob/master/05-Binary-Search-Tree/Course%20Code%20(C%2B%2B)/Optional-10-Binary-Tree-Morris-Traversal/postorder.cpp
Key OD: The integer sequence printed to stdout (Left -> Right -> Root)

Postorder 5: Iterative Postorder (Complex 1-Stack with Visited Pointer)
Filename: po_05_iterative_complex.cpp
Source:https://github.com/fit-coder/fitcoderyoutube/blob/master/tree/construct_tree_from_inorder_and_postorder_iterative.cpp
Key OD: The integer sequence printed to stdout (Left -> Right -> Root)

Postorder 6: Pure C Recursive Implementation
Filename: po_06_pure_c.cpp
Source: Generated by Gemini 3 pro
Key OD: The integer sequence printed to stdout (Left -> Right -> Root)

Postorder 7: Iterative Postorder (Stack with Boolean Flag) Filename: po_07_iterative_flag.cpp
Source:https://github.com/kamyu104/LeetCode-Solutions/blob/master/C%2B%2B/binary-tree-postorder-traversal.cpp
Key OD: The integer sequence printed to stdout (Left -> Right -> Root)

Postorder 8: : Iterative with State Enum (0, 1, 2)
Filename: po_08_iterative_state.cpp
Source:https://github.com/wisdompeak/LeetCode/tree/master/Tree/145.Binary-Tree-Postorder-Traversal
Key OD: The integer sequence printed to stdout (Left -> Right -> Root)

Postorder 9 : Recursive with Helper Reference
Filename: po_09_recursive_helper.cpp
Source:https://github.com/haoel/leetcode/blob/master/algorithms/cpp/binaryTreePostorderTraversal/binaryTreePostorderTraversal.cpp
Key OD: The integer sequence printed to stdout (Left -> Right -> Root)

Postorder 10 : Recursive Functor Object
Filename: po_10_functor.cpp
Source: Gemini 3 Pro
Key OD: The integer sequence printed to stdout (Left -> Right -> Root)

Postorder 11 : Iterative Stack with NULL Marker
Filename: po_11_null_marker.cpp
Source: Generated by Gemini 3 Pro
Key OD: The integer sequence printed to stdout (Left -> Right -> Root)

Postorder 12 : Recursive with Member Variable
Filename: po_12_recursive_member.cpp
Source: Generated by Gemini 3 Pro
Key OD: The integer sequence printed to stdout (Left -> Right -> Root)

Postorder 13 : Iterative Simulated Stack Frame (Program Counter)
Filename: po_13_simulated_pc.cpp
Source: LLM Generated (Gemini)
Key OD: The integer sequence printed to stdout (Left -> Right -> Root)

Postorder 14 : Iterative using Vector Stack (Contiguous Memory)
Filename: po_14_vector_stack.cpp
Source: LLM Generated (Gemini)
Key OD: The integer sequence printed to stdout (Left -> Right -> Root)

Postorder 15 : Iterative using List Stack (Heap Fragmentation)
Filename: po_15_list_stack.cpp
Source: LLM Generated (Gemini)
Key OD: The integer sequence printed to stdout (Left -> Right -> Root)

Postorder 16 : Iterative Raw Array Stack (No STL)
Filename: po_16_raw_array_stack.cpp
Source: LLM Generated (Gemini)
Key OD: The integer sequence printed to stdout (Left -> Right -> Root)

Postorder 17 : Polymorphic Visitor Pattern (Virtual Calls)
Filename: po_17_visitor.cpp
Source: LLM Generated (Gemini)
Key OD: The integer sequence printed to stdout (Left -> Right -> Root)

Postorder 18 : C-Style Function Pointers (Callback)
Filename: po_18_function_pointer.cpp
Source: LLM Generated (Gemini)
Key OD: The integer sequence printed to stdout (Left -> Right -> Root)

Postorder 19 : Iterative using Deque (Double Indirection)
Filename: po_19_deque.cpp
Source: LLM Generated (Gemini)
Key OD: The integer sequence printed to stdout (Left -> Right -> Root)

Postorder 20 : Struct of Arrays (Global Indexing)
Filename: po_20_struct_of_arrays.cpp
Source: LLM Generated (Gemini)
Key OD: The integer sequence printed to stdout (Left -> Right -> Root)

Postorder 21 : Iterative Destructive (Sign Bit Marking)
Filename: po_21_destructive_sign.cpp
Source: LLM Generated (Gemini)
Key OD: The integer sequence printed to stdout (Left -> Right -> Root)

Postorder 22 : Iterative with Goto (Unstructured)
Filename: po_22_goto_spaghetti.cpp
Source: LLM Generated (Gemini)
Key OD: The integer sequence printed to stdout (Left -> Right -> Root)

Postorder 23 : Iterative Tagged Pointer (Bitwise State)
Filename: po_23_tagged_pointer.cpp
Source: LLM Generated (Gemini)
Key OD: The integer sequence printed to stdout (Left -> Right -> Root)

Postorder 24 : Iterative Explicit Continuation (Task Stack)
Filename: po_24_task_stack.cpp
Source: LLM Generated (Gemini)
Key OD: The integer sequence printed to stdout (Left -> Right -> Root)

Postorder 25 : Iterative Stack (Peek & Prune)
Filename: po_25_peek_prune.cpp
Source: LLM Generated (Gemini)
Key OD: The integer sequence printed to stdout (Left -> Right -> Root)

Postorder 26 : Iterative with std::set (External State)
Filename: po_26_std_set.cpp
Source: LLM Generated (Gemini)
Key OD: The integer sequence printed to stdout (Left -> Right -> Root)

Postorder 27 : Iterative with Bit-Vector (Bitwise Logic)
Filename: po_27_bit_vector.cpp
Source: LLM Generated (Gemini)
Key OD: The integer sequence printed to stdout (Left -> Right -> Root)

Postorder 28 : Recursive with Exception Flow (Throw/Catch)
Filename: po_28_exception_flow.cpp
Source: LLM Generated (Gemini)
Key OD: The integer sequence printed to stdout (Left -> Right -> Root)

Postorder 29 : Iterative with Map State (External Lookup)
Filename: po_29_map_state.cpp Source: LLM Generated (Gemini)
Key OD: The integer sequence printed to stdout (Left -> Right -> Root)

Postorder 30 : Stackless Iterative (Parent Map Backtracking)
Filename: po_30_parent_map.cpp
Source: LLM Generated (Gemini)
Key OD: The integer sequence printed to stdout (Left -> Right -> Root)