1 Iterators

1. Iterators provide a useful ______ for iterating a collection:
   (a) Encapsulation
   (b) Modularity
   (c) Abstraction
   (d) Inheritance

2. The Iterator interface includes method signatures for:
   (a) iterator
   (b) hasNext, next
   (c) isEmpty, size
   (d) ArrayIterator

3. The Iterable interface includes method signatures for:
   (a) iterator
   (b) hasNext, next
   (c) isEmpty, size
   (d) ArrayIterator

4. For an iterable collection:
   (a) It’s possible that nobody knows how the collection works
   (b) At least one person must know how the collection works
   (c) The collection must use arrays
   (d) The collection must be linear

5. It’s possible to use the exact same toString method for every iterable collection.
   (a) True
   (b) False

6. An ArrayIterator works exactly the same way as a LinkedIterator (same implementation):
   (a) True
   (b) False

7. Iterators guarantee that you’ll traverse the elements in your preferred order:
   (a) True
   (b) False
2 Binary Trees

Consider the following binary search tree for the next few questions:

8. Trace out the pre-order traversal of the above binary search tree.
   (a) P, E, W, C, L, S, Z
   (b) C, L, S, Z, E, W, P
   (c) C, E, L, P, S, W, Z
   (d) P, E, C, L, W, S, Z
   (e) C, L, E, S, Z, W, P

9. Trace out the in-order traversal of the above binary search tree.
   (a) P, E, W, C, L, S, Z
   (b) C, L, S, Z, E, W, P
   (c) C, E, L, P, S, W, Z
   (d) P, E, C, L, W, S, Z
   (e) C, L, E, S, Z, W, P

10. Trace out the post-order traversal of the above binary search tree.
    (a) P, E, W, C, L, S, Z
    (b) C, L, S, Z, E, W, P
    (c) C, E, L, P, S, W, Z
    (d) P, E, C, L, W, S, Z
    (e) C, L, E, S, Z, W, P

11. Trace out the level-order traversal of the above binary search tree.
    (a) P, E, W, C, L, S, Z
    (b) C, L, S, Z, E, W, P
    (c) C, E, L, P, S, W, Z
    (d) P, E, C, L, W, S, Z
    (e) C, L, E, S, Z, W, P
Consider the following binary search tree for the next few questions:

![Binary Search Tree Diagram](image)

**12.** Trace out the pre-order traversal of the above binary search tree.
(a) G, B, F, J, I, K, M, L, P, X, T
(c) F, B, I, J, G, L, P, M, X, T, K
(d) K, G, T, B, J, M, X, F, I, L, P
(e) K, G, B, F, J, I, T, M, L, P, X

**13.** Trace out the in-order traversal of the above binary search tree.
(a) G, B, F, J, I, K, M, L, P, X, T
(c) F, B, I, J, G, L, P, M, X, T, K
(d) K, G, T, B, J, M, X, F, I, L, P
(e) K, G, B, F, J, I, T, M, L, P, X

**14.** Trace out the post-order traversal of the above binary search tree.
(a) G, B, F, J, I, K, M, L, P, X, T
(c) F, B, I, J, G, L, P, M, X, T, K
(d) K, G, T, B, J, M, X, F, I, L, P
(e) K, G, B, F, J, I, T, M, L, P, X

**15.** Trace out the level-order traversal of the above binary search tree.
(a) G, B, F, J, I, K, M, L, P, X, T
(c) F, B, I, J, G, L, P, M, X, T, K
(d) K, G, T, B, J, M, X, F, I, L, P
(e) K, G, B, F, J, I, T, M, L, P, X

**16.** A binary tree is a linear abstract data type:
(a) True
(b) False
17. Trees can have multiple distinguished elements that are the root (nodes whose parents are null):
   (a) True
   (b) False

18. The height of a tree is:
   (a) The number of nodes in the tree
   (b) The number of edges in the tree
   (c) The number of edges in the longest path
   (d) The number of nodes in the longest path

19. Binary trees are trees that:
   (a) Can be encoded using binary values
   (b) Have either 1 or 2 roots
   (c) Each node has at most 2 children
   (d) Each node has at most 2 parents

20. Trees can only be traversed using recursive algorithms:
   (a) True
   (b) False

21. Binary search trees are binary trees that:
   (a) Are ordered
   (b) Can be traversed easier
   (c) Can have more children
   (d) Can have more parents
3 Sorting

22. Compute the number of times a value has to be pushed onto the sorted stack when performing the stack-based insertion sort on the array A = [3, 1, 4, 2]:
(a) 4
(b) 6
(c) 7
(d) 8

23. Compute the number of times a value has to be pushed onto the sorted stack when performing the stack-based insertion sort on the array A = [4, 3, 2, 1]:
(a) 1
(b) 4
(c) 8
(d) 16

24. Compute the number of times a value has to be pushed onto the sorted stack when performing the stack-based insertion sort on the array A = [1, 2, 3, 4]:
(a) 4
(b) 8
(c) 10
(d) 16

25. Compute the number of times a value has to be pushed onto the sorted stack when performing the stack-based insertion sort on an array with n elements that are already ordered in decreasing (reverse) order:
(a) 1
(b) n
(c) 1 + 2 + ... + n
(d) \( n^2 \)

26. Compute the number of times a value has to be pushed onto the sorted stack when performing the stack-based insertion sort on an array with n elements that are already ordered in increasing order:
(a) 1
(b) n
(c) 1 + 2 + ... + n
(d) \( n^2 \)

27. Suppose you are running Quick Sort on the array A = [3, 6, 1, 8, 9], what would be the optimal first pivot?
(a) 3
(b) 6
(c) 1
(d) 8
28. Suppose you are running Quick Sort on the array $A = [3, 6, 1, 8, 9]$, and the first pivot chosen arbitrarily was 3. How many items would be in the smaller and larger containers?
(a) 1 (in smaller) and 3 (in larger)
(b) 1 (in smaller) and 4 (in larger)
(c) 0 (in smaller) and 3 (in larger)
(d) 0 (in smaller) and 4 (in larger)
(e) 3 (in smaller) and 3 (in larger)

29. The worst case time complexities of each of the sorting algorithms covered in class are the same:
(a) True
(b) False