**CP12 – Data Structures (Level 1)** Name: Total: \_\_\_\_\_ / 20

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Simple Linked List | | 0 | 1 | 2 |
| Contains  (Traversal) | While loop – target = target.next;  Uses .equals |  |  |  |
| Get/ findIndex | While loop – target = target.next; |  |  |  |
| Array Set | | 0 | 1 | 2 |
| Add | Finds the correct index, adds an element to an array slot that was null before  Duplicates not allowed |  |  |  |
| Add – When array gets full | capacity doubles: copy elements to new array |  |  |  |
| Remove in Middle | Finds index of target element  Elements after target get moved 1 slot to the left |  |  |  |
| Contains/findIndex | While-Loops |  |  |  |
| Sorted Double Linked List | | 0 | 1 | 2 |
| Add | New Node’s prev&next  Prev & Next’s links update to new Node (prev.next and target.next.prev) |  |  |  |
| Remove in Middle | Prev & Next’s links update to each other (prev.next and target.next.prev) |  |  |  |
| Remove from Head/Tail | Links to head/tail update in SDLL class  (Old head’s next).prev becomes null / (Old tail’s prev).next becomes null |  |  |  |

**CP12 – Data Structures (Level 1)** Name: Total: \_\_\_\_\_ / 20

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Simple Linked List | | 0 | 1 | 2 |
| Contains  (Traversal) | While loop – target = target.next;  Uses .equals |  |  |  |
| Get/ findIndex | While loop – target = target.next; |  |  |  |
| Array Set | | 0 | 1 | 2 |
| Add | Finds the correct index, adds an element to an array slot that was null before  Duplicates not allowed |  |  |  |
| Add – When array gets full | capacity doubles: copy elements to new array |  |  |  |
| Remove in Middle | Finds index of target element  Elements after target get moved 1 slot to the left |  |  |  |
| Contains/findIndex | While-Loops |  |  |  |
| Sorted Double Linked List | | 0 | 1 | 2 |
| Add | New Node’s prev&next  Prev & Next’s links update to new Node (prev.next and target.next.prev) |  |  |  |
| Remove in Middle | Prev & Next’s links update to each other (prev.next and target.next.prev) |  |  |  |
| Remove from Head/Tail | Links to head/tail update in SDLL class  (Old head’s next).prev becomes null / (Old tail’s prev).next becomes null |  |  |  |

**CP12 – Data Structures (Level 2)** Name: Total: \_\_\_\_\_ / 20

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Simple Linked List | | 0 | 1 | 2 |  |
| Contains  (Traversal) | While loop – target = target.next;  Uses .equals |  |  |  |  |
| Get/ findIndex | While loop – target = target.next; |  |  |  |  |
| Binary Tree | | 0 | 2 | 3 | 4 |
| Contains/Add (Traversal) | Use compareTo to move left/right  Finds correct element/location  Add: ignore duplicates |  |  |  |  |
| Traversing Trees (Size, dFS, bFS) | Goes through the tree to count/find the elements  Use an ArrayList / Recursion for dFS/bFS  Size: /2  dFS: /1  bFS: /1 |  |  |  |  |
| Remove | Keep track of parent while searching for target  Keep track of the direction from parent->target |  |  |  |  |
| (0/1 children)  (1 child) | Target is removed from parent.left/parent.right  parent.child becomes target.child |  |  |  |  |
| Remove (Two children) | extremeLeft & extremeLeftParent correctly found |  |  |  |  |

**CP12 – Data Structures (Level 2)** Name: Total: \_\_\_\_\_ / 20

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Simple Linked List | | 0 | 1 | 2 |  |
| Contains  (Traversal) | While loop – target = target.next;  Uses .equals |  |  |  |  |
| Get/ findIndex | While loop – target = target.next; |  |  |  |  |
| Binary Tree | | 0 | 2 | 3 | 4 |
| Contains/Add (Traversal) | Use compareTo to move left/right  Finds correct element/location  Add: ignore duplicates |  |  |  |  |
| Traversing Trees (Size, dFS, bFS) | Goes through the tree to count/find the elements  Use an ArrayList / Recursion for dFS/bFS  Size: /2  dFS: /1  bFS: /1 |  |  |  |  |
| Remove | Keep track of parent while searching for target  Keep track of the direction from parent->target |  |  |  |  |
| (0/1 children)  (1 child) | Target is removed from parent.left/parent.right  parent.child becomes target.child |  |  |  |  |
| Remove (Two children) | extremeLeft & extremeLeftParent correctly found |  |  |  |  |