Self-Check Exercises 1-5 Section 2.5

Big-O

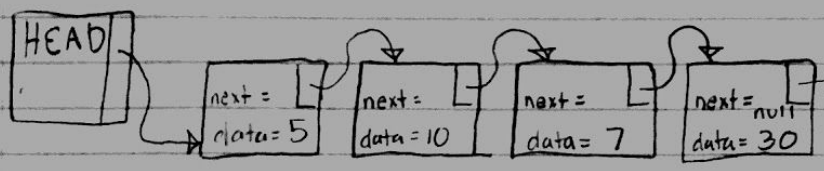
1.) Single-linked list get() operation.

$\Theta(n)$

2.) set() operation
$\Theta(n)$

3.) add() methods:
add(int index, E item) - $\Theta(1)$
add(E item) - $\Theta(n)$
bool add(E item) - $\Theta(n)$

4.) Draw; Single Linked List of Integer objects 5, 10, 7, and 30
Complete fragment: adds all Integer objects.




```
1 fragment addNums(){
2 int sum = 0;
3 Node<Integer> nodeRef = __
4 while (nodeRef != null){
5 int next = __;
6 sum += next;
7 nodeRef = __; }
}
```

line 3 Node<Integer> nodeRef = head;

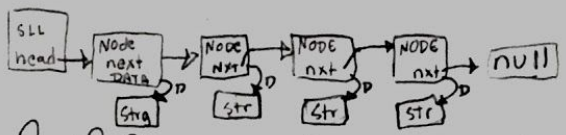
...

line 5 int next = nodeRef.next;

...

line 7 nodeRef = nodeRef.next;

Fig. 2.16



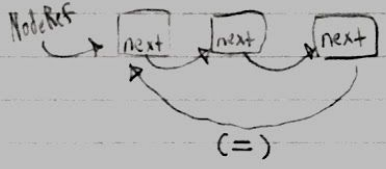
Self Check, Section 2.5 cont.

5.) For Linked List in Figure 2.16, data field head (type Node) references the first node;

("Shakira")

a.) Sets current head to the new Node passing the data (new the--head) and the next pointer to old head's. next

b.) Sets the nodeRef pointer to next node after the head sets nodeRefs next pointer to the next pointer's pointer.



c.) Sets the nodeRef pointer to head
traverses list by following the reference (and assigning nodeRef to it's own next pointer, while the next node is not null. Adds a new node to end of list with String "Tamitka"

d.) Sets nodeRef to head, traverses List by assigning nodeRef to next node, while next node is not null and does not equal the string "Harry", If "Harry" is found, sets that node's data to "Sally" and sets the next node to "Harry" and sets the next node to "Harry"'s next node reference.