**Program Requirements:**

Implement the required operations on the class example linked List program:

• Accessor: return true if the list is empty

• Accessor: return the length of the list as an unsigned integer

• Accessor: search the list for an item, return true on success

• Mutator: delete an item from the list, return true on success

(the logic for delete is described in the sample code as a comment)

• Operator: overload the assignment operator

• Constructor: create a copy constructor

**Program Inputs:**

Values of the Linked List being input

**Program Outputs:**

Various outputs of the functions performed on the Linked list via displayList() function.

**Test Plan:**

|  |  |
| --- | --- |
| **Values** | **addNode()** |
| 10,20,30,40,50 | **true** |
| **Value** | **delNode()** |
| 20 | **True** |
| **Value** | **displayList()** |
| 20 | **10,30,40,50** |
| **Value** | **searchList()** |
| 40 | True |
| **Expression** | **Return** |
| a = b | **Values in a are assigned to values in b** |
| **Expression** | **Return** |
| b = b | **Self assignment occurs** |
| **EmptyList(**a**)** | **Returns False** |
| **listSize(**b**)** | **Returns 4** |

|  |  |
| --- | --- |
| **Values** | **addNode()** |
| 100, 40, 60, 70 | **true** |
| **Value** | **delNode()** |
| 20 | **False** |
| **Value** | **displayList()** |
| None | **100, 40, 60, 70** |
| **Value** | **searchList()** |
| 40 | True |
| **Expression** | **Return** |
| a = b | **Values in a are assigned to values in b** |
| **Expression** | **Return** |
| b = b | **Self assignment occurs** |
| **EmptyList(**a**)** | **Returns False** |
| **listSize(**b**)** | **Returns 4** |

**Alogrithm:**

Driver: main()

1. Take input
2. Evaluate List functionality as per requirements stated

emptyList()

1. If head is not null return true else false

listSize()

1. Return the size of the list

searchNode(value):

1. Set Boolean variable success to False
2. Get a node variable named current
3. If head is not Null(list is not empty) assign head to current and continue to loop
4. Loop till current is not Null
5. If current node’s data is equal to the search value then set success to true
6. Otherwise set current to next node and repeat the steps 4 and 5 till current is Null
7. Return Boolean success

delNode(value):

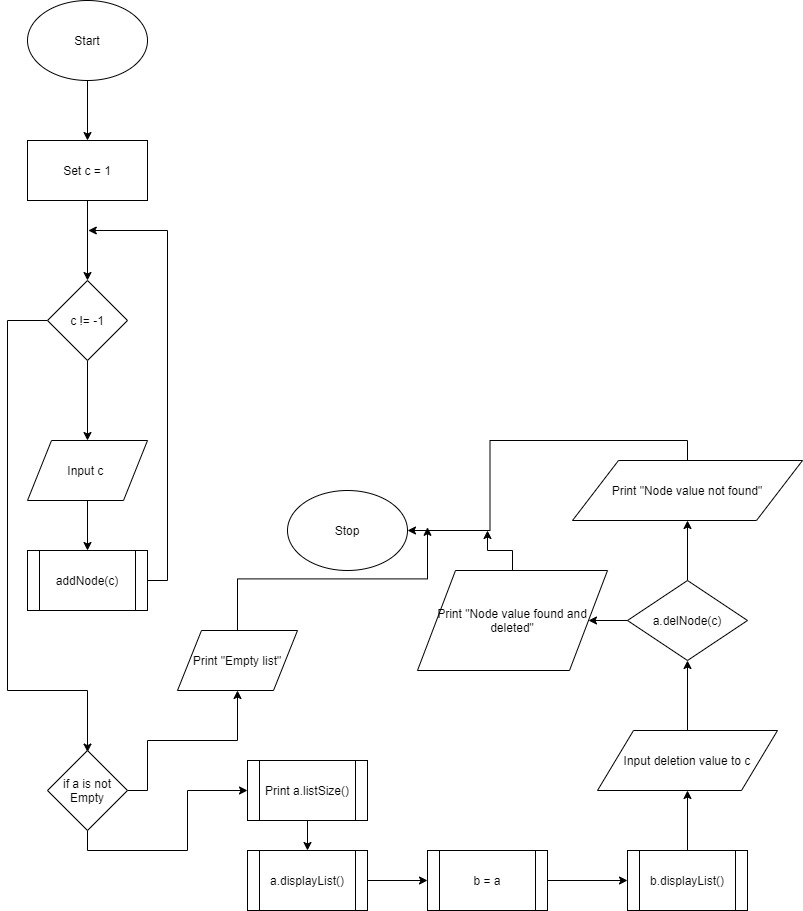
1. Set Boolean deleted to false and create new node variables current and temp
2. If head is not Null(list not empty) and value is present in list, continue otherwise return deleted value
3. If head data is equal to value data, assign temp to head, head to the next node and then delete memory allocated to temp while setting deleted value to true
4. Otherwise set current as head and loop till next node from current holds the value
5. Assign temp to current, current to the next node and then delete memory allocated to temp while setting deleted value to true and breaking the loop otherwise assign current to next node in list
6. Return value of deleted

Operator=():

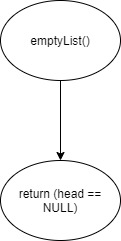
1. Declare two nodes c1 and c2
2. Assign c1 to the assigned value head and c2 to head of current object
3. If the current object has same reference as the assigned object then return current object’s reference to avoid self assignment otherwise proceed
4. Loop step 5 till c1 does not point to Null
5. Set c2 data to c1 data, c2 next to c1 next and then set them both to their next nodes
6. Return a reference to the current pointer

**Flowchart:**

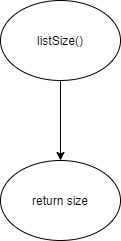
Main:



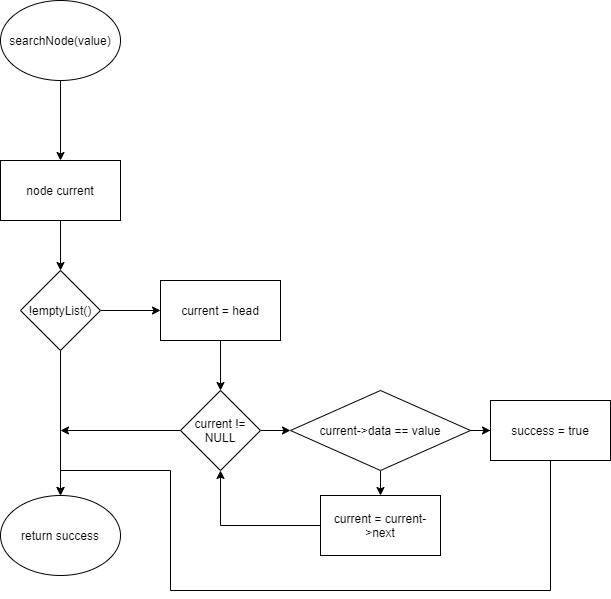
emptyList():



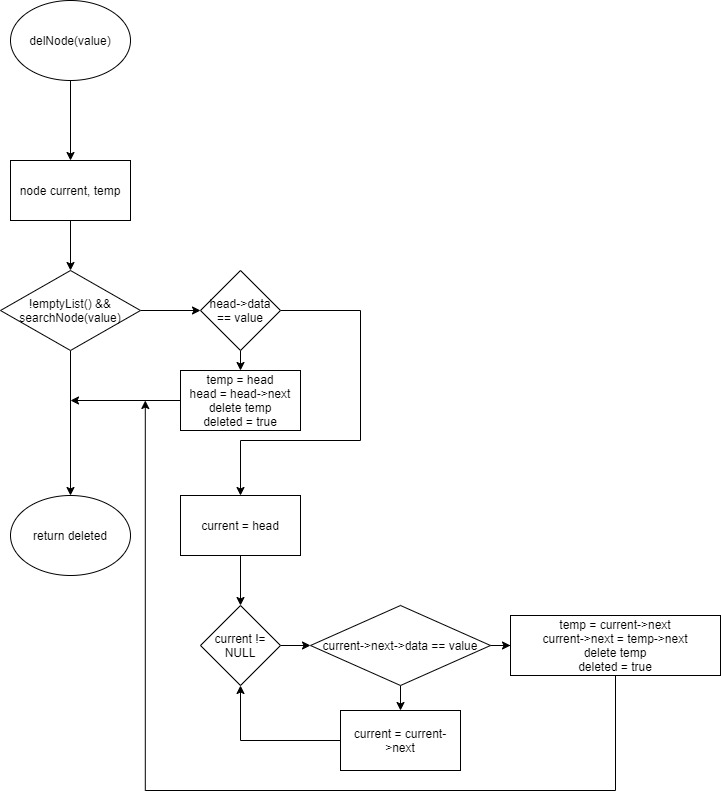
listSize():



searchNode():



delNode():



Operator=():

