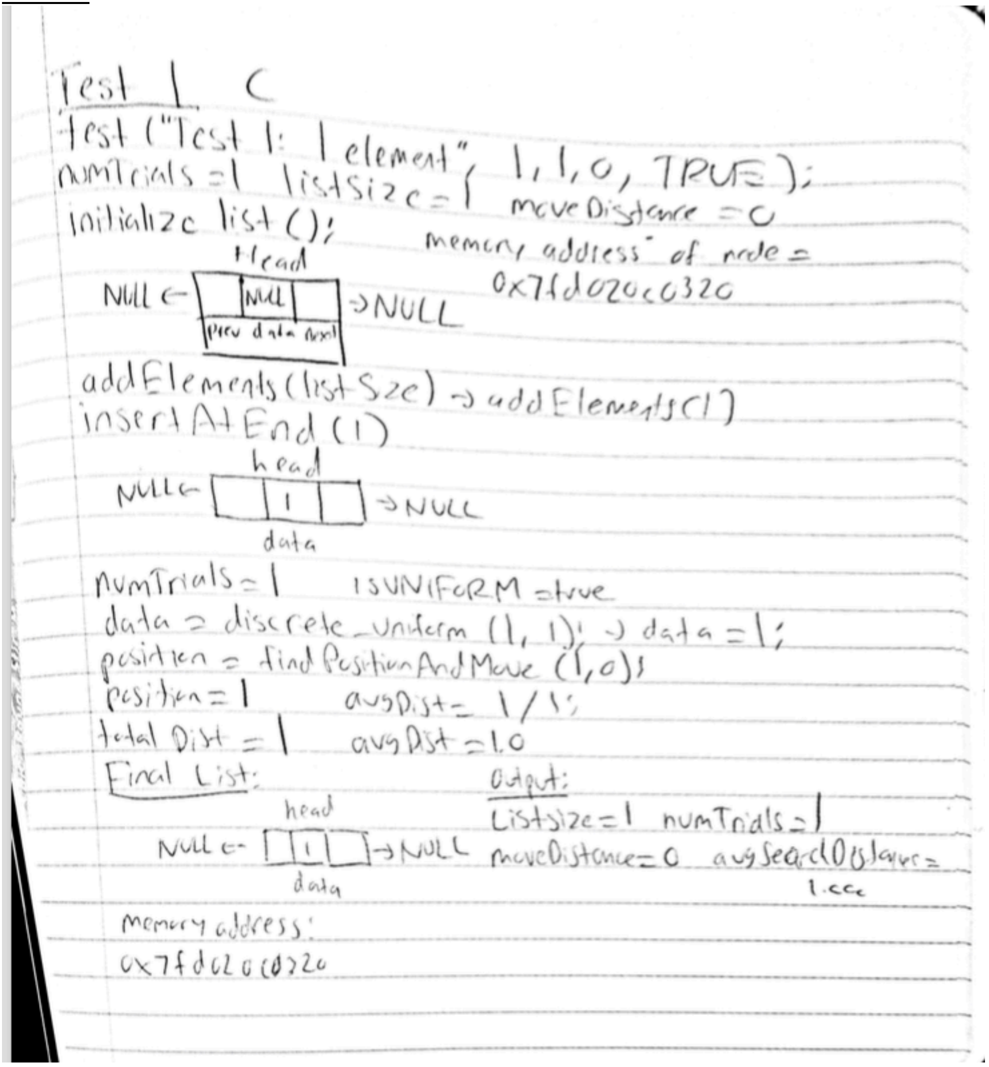
**Exercise 3:** PDF drawings of the list at each stage of insertion and list adjustment.
(Linked List in C)
Test 1:



Test 1 C

test ("Test 1: 1 element", 1, 1, 0, TRUE);

numTrials = 1 listSize = 1 moveDistance = 0

initialize list();

memory address of node =
0x7fd020c0320

Head

NULL ← | | NULL | | → NULL

prev data next

addElements(listSize) → addElements(1)

insertAtEnd(1)

head

NULL ← | | 1 | | → NULL

data

numTrials = 1 ISUNIFORM = true

data = discrete_uniform (1, 1); → data = 1;

position = findPositionAndMove (1, 0);

position = 1 avgDist = 1 / 1;

total Dist = 1 avgDist = 1.0

Final List:

head

NULL ← | | 1 | | → NULL

data

Output:

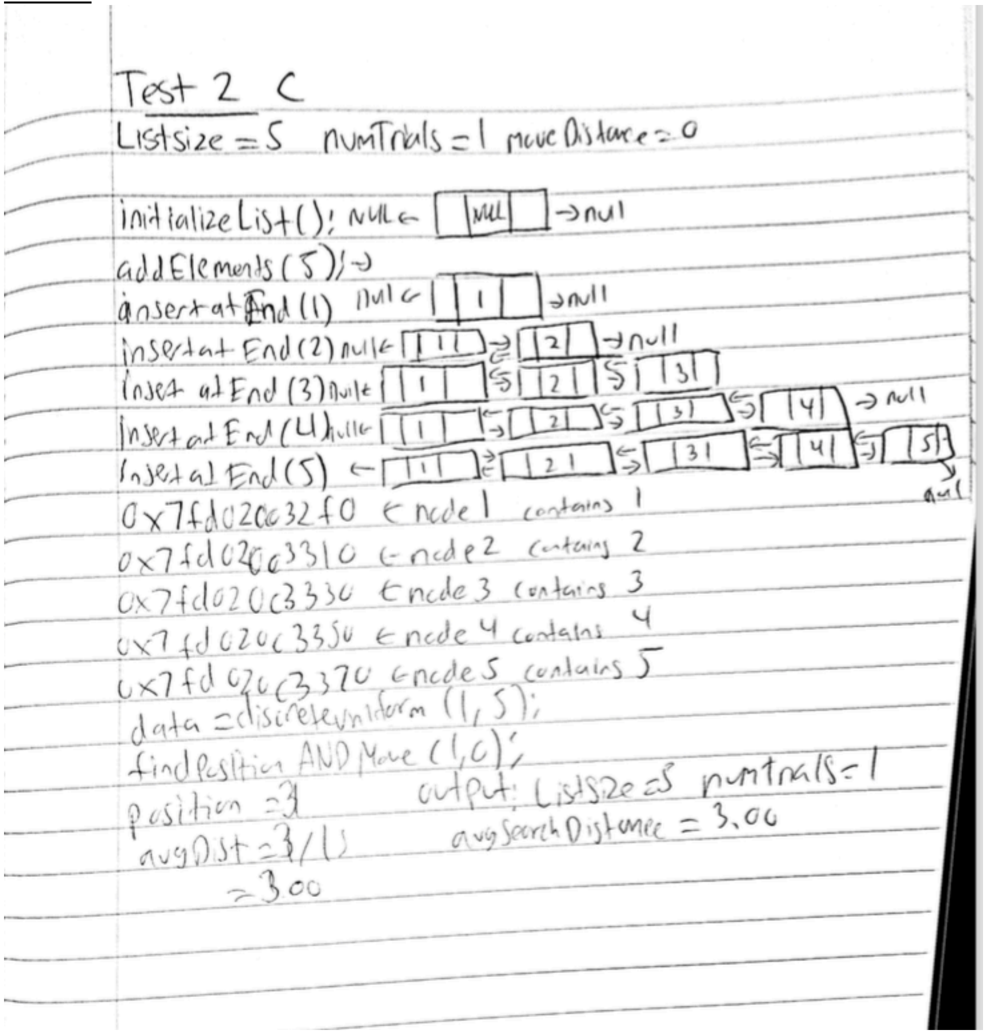
Listsize = 1 numTrials = 1

moveDistance = 0 avgSearchDistance =
1.000

Memory address:

0x7fd020c0320

Test 2:



Test 2 C

Listsize = 5 numTrials = 1 moveDistance = 0

initializeList(); NULL ← | NULL | → null

addElements(5); →

insertatEnd(1) null ← | 1 | → null

insertatEnd(2) null ← | 1 | ⇄ | 2 | → null

insertatEnd(3) null ← | 1 | ⇄ | 2 | ⇄ | 3 |

insertatEnd(4) null ← | 1 | ⇄ | 2 | ⇄ | 3 | ⇄ | 4 | → null

insertatEnd(5) ← | 1 | ⇄ | 2 | ⇄ | 3 | ⇄ | 4 | ⇄ | 5 | → null

0x7fd020c32f0 ← node 1 contains 1

0x7fd020c3310 ← node 2 contains 2

0x7fd020c3330 ← node 3 contains 3

0x7fd020c3350 ← node 4 contains 4

0x7fd020c3370 ← node 5 contains 5

data = discreteuniform(1, 5);

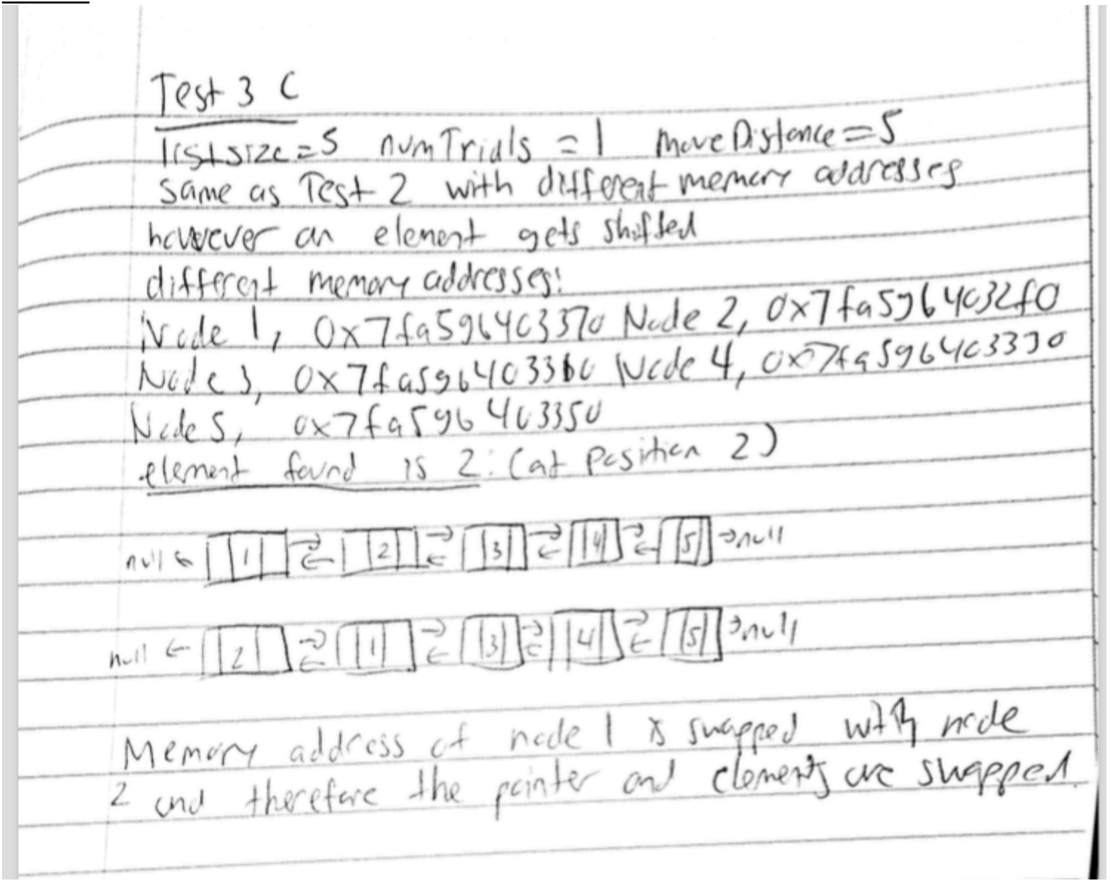
findPosition AND Move(1, 0);

position = 3 output: ListSize = 5 numtrials = 1

avgDist = 3/1 avgSearchDistance = 3.00

= 3.00

Test 3:



Test 3 C

listsize = 5 numTrials = 1 moveDistance = 5
same as Test 2 with different memory addresses
however an element gets shifted
different memory addresses:
Node 1, 0x7fa5964033?0 Node 2, 0x7fa5964032f0
Node 3, 0x7fa5964033b0 Node 4, 0x7fa596403330
Node 5, 0x7fa596403350
element found is 2: (at position 2)

$$\begin{tikzcd}
\text{null} & \boxed{1} \arrow[l] \arrow[r, shift left] & \boxed{2} \arrow[l, shift left] \arrow[r, shift left] & \boxed{3} \arrow[l, shift left] \arrow[r, shift left] & \boxed{4} \arrow[l, shift left] \arrow[r, shift left] & \boxed{5} \arrow[l, shift left] \arrow[r] & \text{null}
\end{tikzcd}$$

$$\begin{tikzcd}
\text{null} & \boxed{2} \arrow[l] \arrow[r, shift left] & \boxed{1} \arrow[l, shift left] \arrow[r, shift left] & \boxed{3} \arrow[l, shift left] \arrow[r, shift left] & \boxed{4} \arrow[l, shift left] \arrow[r, shift left] & \boxed{5} \arrow[l, shift left] \arrow[r] & \text{null}
\end{tikzcd}$$

Memory address of node 1 is swapped with node 2 and therefore the pointer and elements are swapped.

**Average search depth for a list of 10 elements using various values of moveDistance in the range of 1 to 10.**

Using a list with 10 elements, 10 as the number of trials and values of moveDistance from 1 to 7, here are the values for average search distance:

| Move Distance | Average Search Distance |
|---|---|
| 1 | 7.2 |
| 2 | 6.4 |
| 3 | 4.7 |
| 4 | 5.7 |
| 5 | 5.6 |
| 6 | 6.8 |
| 7 | 5.3 |

**Code to prove above:**

```
10 elements
  Listsize=10 numTrials=10 moveDistance=1  avgSearchDistance=7.200000
10 elements
  Listsize=10 numTrials=10 moveDistance=2  avgSearchDistance=6.400000
10 elements
  Listsize=10 numTrials=10 moveDistance=3  avgSearchDistance=4.700000
10 elements
  Listsize=10 numTrials=10 moveDistance=4  avgSearchDistance=5.700000
10 elements
  Listsize=10 numTrials=10 moveDistance=5  avgSearchDistance=5.500000
10 elements
  Listsize=10 numTrials=10 moveDistance=6  avgSearchDistance=6.800000
10 elements
  Listsize=10 numTrials=10 moveDistance=7  avgSearchDistance=5.300000
```