Linked list complete code:

Code:

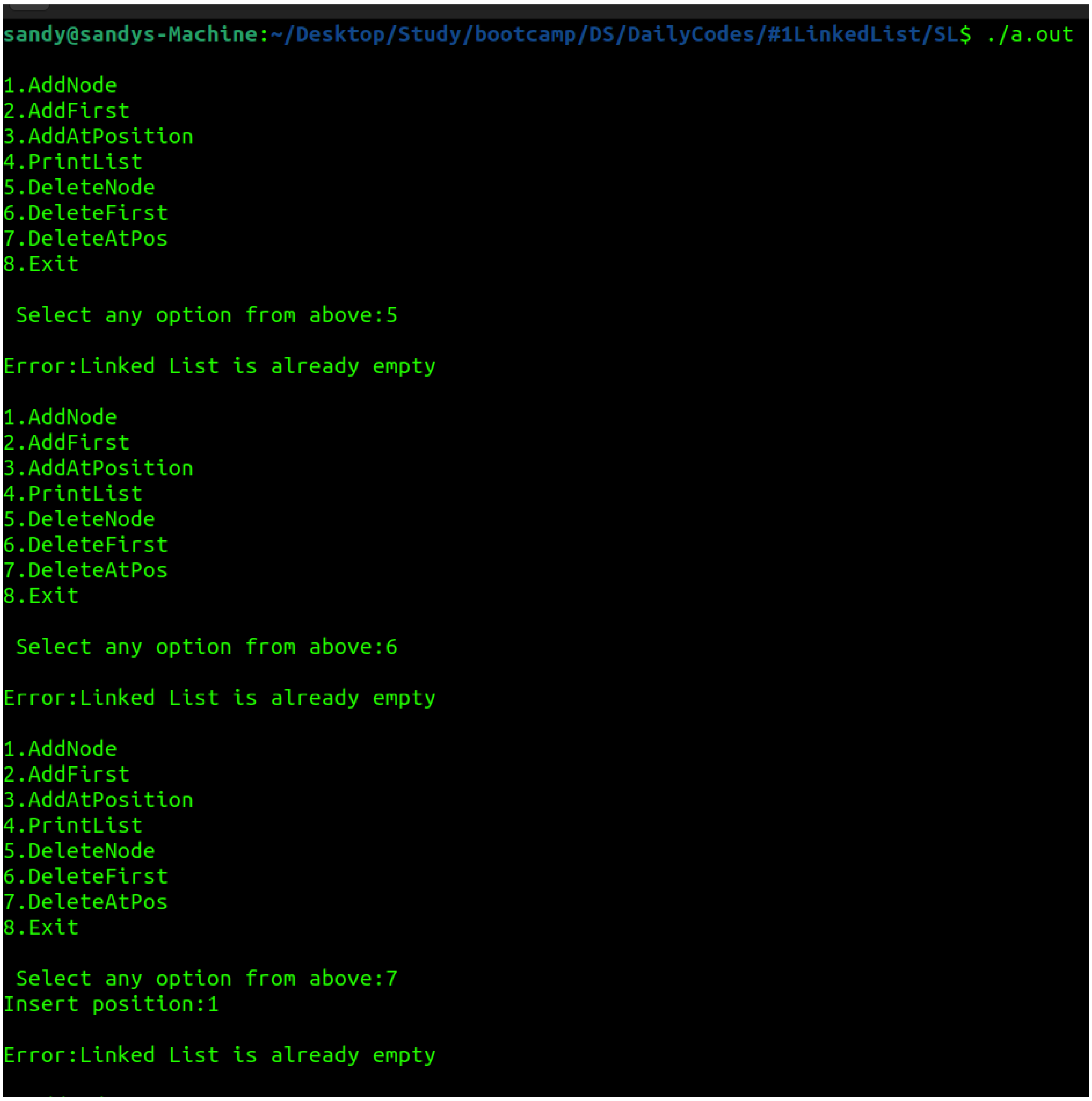
```
#include<stdio.h>
#include<stdlib.h>

typedef struct Demo {

        int data;
        struct Demo *next;

}Demo;

Demo *head = NULL;

int countNodes() {

        Demo *tmp=head;
        int cnt=0;

        while(tmp != NULL) {

                cnt++;
                tmp=tmp->next;
        }

        return cnt;
}

Demo *createNode() {

        Demo *newNode = (Demo*)malloc(sizeof(Demo));

        printf("Enter data:");
        scanf("%d",&newNode->data);

        newNode->next = NULL;

        return newNode;
}

void addNode() {

        Demo *newNode = createNode();

        if(head == NULL)
                head = newNode;
        else {

                Demo *tmp = head;

                while(tmp->next != NULL)
                        tmp = tmp->next;

                tmp->next = newNode;
```

```
		}
	}

	void addFirst() {

		Demo *newNode = createNode();

		if(head != NULL)
			newNode->next = head;

		head = newNode;
	}

	void addAtPos() {

		int pos;
		printf("Insert position:");
		scanf("%d",&pos);

		if(pos == 1)
			addFirst();
		else if(pos > countNodes()+1 || pos < 1) {		//if position is greater than nodes present or 
position is -ve

			/*
			 *	-if there are 2 nodes and i gives pos=3 still this code work
			 *	-error will come only when position is countNode()+1
			 *	-this situation is same as adding node at last
			 */

			printf("\nError:you entered incorrect positioni\n");

		}else {

			Demo *newNode = createNode();
			Demo *tmp = head;

			while(pos-2) {

				tmp=tmp->next;
				pos--;
			}

			newNode->next = tmp->next;
			tmp->next = newNode;
		}
	}

	void printList() {

		Demo *tmp=head;

		printf("--------------------------------------------------------\n");

		while(tmp != NULL) {

			printf("|%d|->",tmp->data);
```

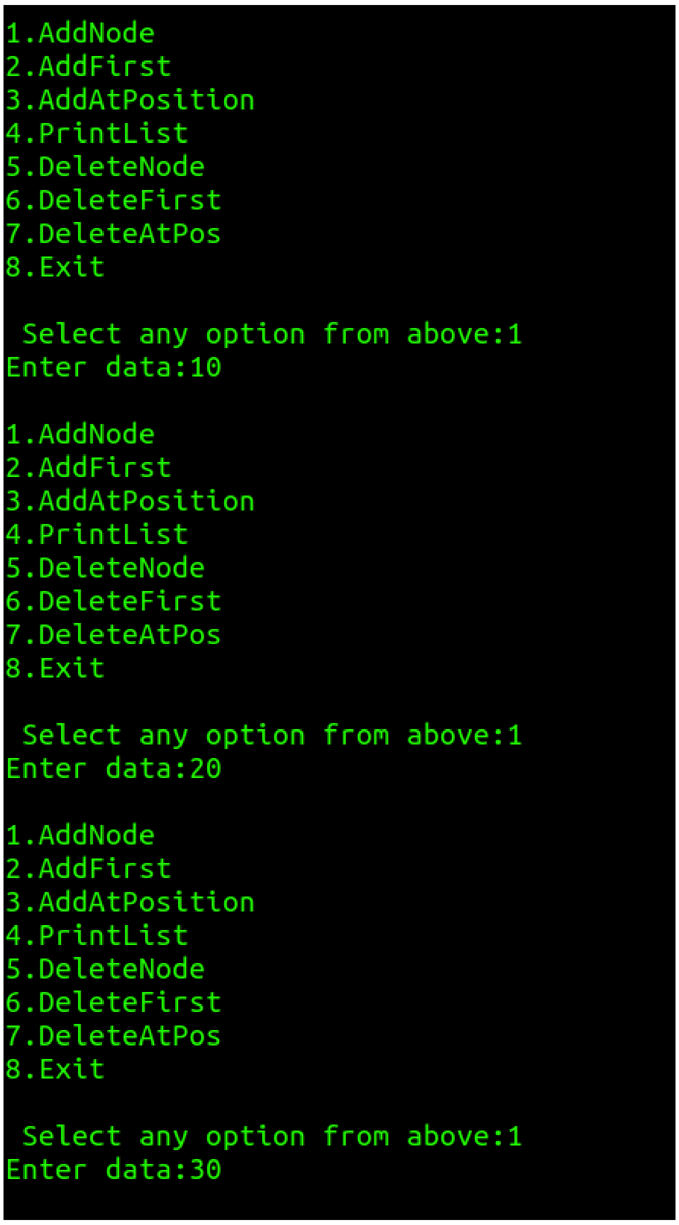
```
                tmp=tmp->next;
        }
        printf("NULL\n");
        printf("-----------------------------------------------------\n");
}

void deleteNode() {

        if(head == NULL) {

                printf("\nError:Linked List is already empty\n");

        }else if(countNodes() == 1) {           //if there is only one node in list
                free(head->next);
                head = NULL;
        }else {

                Demo *tmp =head;

                while(tmp->next->next != NULL)
                tmp = tmp->next;

                free(tmp->next);
                tmp->next =NULL;

        }
}

void deleteFirst() {

        if(head == NULL) {

                printf("\nError:Linked List is already empty\n");
        }else {

                Demo *tmp = head;
                head = tmp->next;
                free(tmp);
        }
}

void deleteAtPos() {

        int pos;
  printf("Insert position:");
  scanf("%d",&pos);

  if(pos == 1) {
       deleteFirst();
        }else if(pos > countNodes() || pos < 1) {

                printf("\nError:you entered incorrect position\n");

        }else if(pos == countNodes()) {         //if pos is last node

                deleteNode();
        }else {
                Demo *tmp1 = head;
                Demo *tmp2 = NULL;

                while(pos-2) {
                        tmp1 = tmp1->next;
                        pos--;
                }

                tmp2 = tmp1 -> next;
                tmp1->next = tmp2->next;
                free(tmp2);
        }

}

void main() {

        int ch;

        while(1) {

                printf("\n1.AddNode\n");
                printf("2.AddFirst\n");
                printf("3.AddAtPosition\n");
                printf("4.PrintList\n");
                printf("5.DeleteNode\n");
                printf("6.DeleteFirst\n");
                printf("7.DeleteAtPos\n");
                printf("8.Exit\n");

                printf("\n Select any option from above:");
                scanf("%d",&ch);


                switch(ch) {

                        case 1:
                                addNode();
                                break;
                        case 2:
                                addFirst();
                                break;
                        case 3:
                                addAtPos();
                                break;
                        case 4:
                                printList();
                                break;
                        case 5:
                                deleteNode();
                                break;
                        case 6:
                                deleteFirst();
                                break;
                        case 7:
                                deleteAtPos();
```

```
					break;
				case 8:
					exit(0);
					break;
			}
		}

	}
```

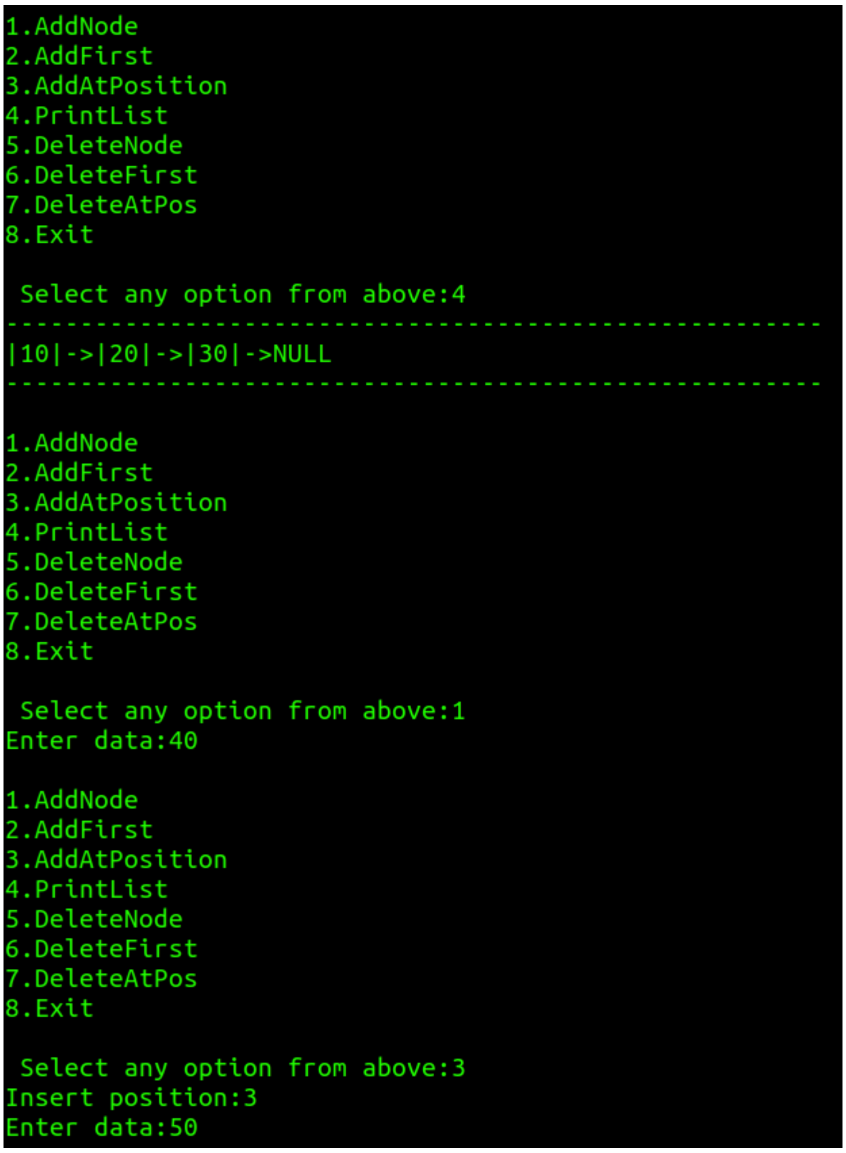
Output:



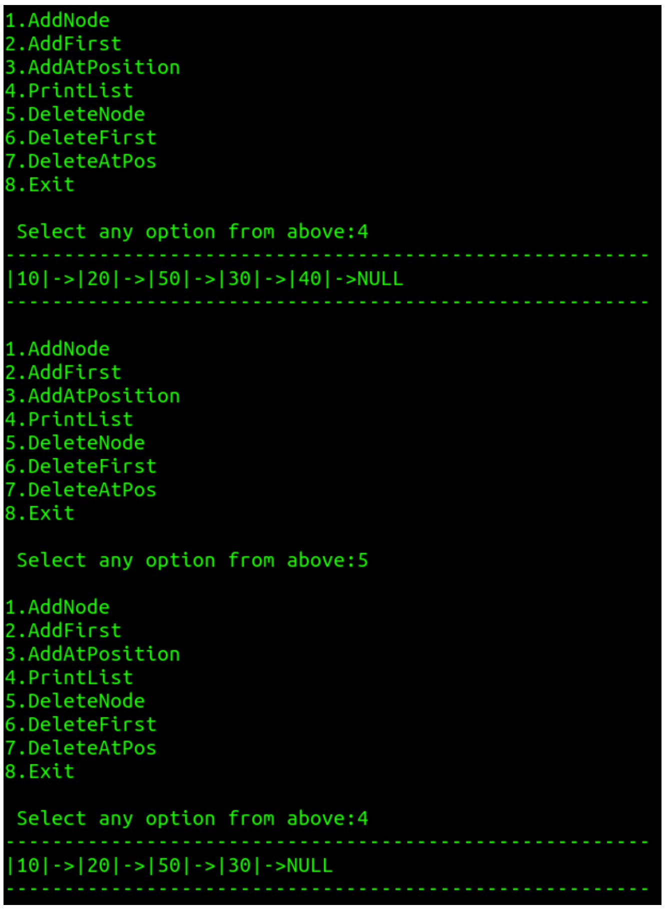
```
sandy@sandys-Machine:~/Desktop/Study/bootcamp/DS/DailyCodes/#1LinkedList/SL$ ./a.out

1.AddNode
2.AddFirst
3.AddAtPosition
4.PrintList
5.DeleteNode
6.DeleteFirst
7.DeleteAtPos
8.Exit

 Select any option from above:5

Error:Linked List is already empty

1.AddNode
2.AddFirst
3.AddAtPosition
4.PrintList
5.DeleteNode
6.DeleteFirst
7.DeleteAtPos
8.Exit

 Select any option from above:6

Error:Linked List is already empty

1.AddNode
2.AddFirst
3.AddAtPosition
4.PrintList
5.DeleteNode
6.DeleteFirst
7.DeleteAtPos
8.Exit

 Select any option from above:7
Insert position:1

Error:Linked List is already empty
```



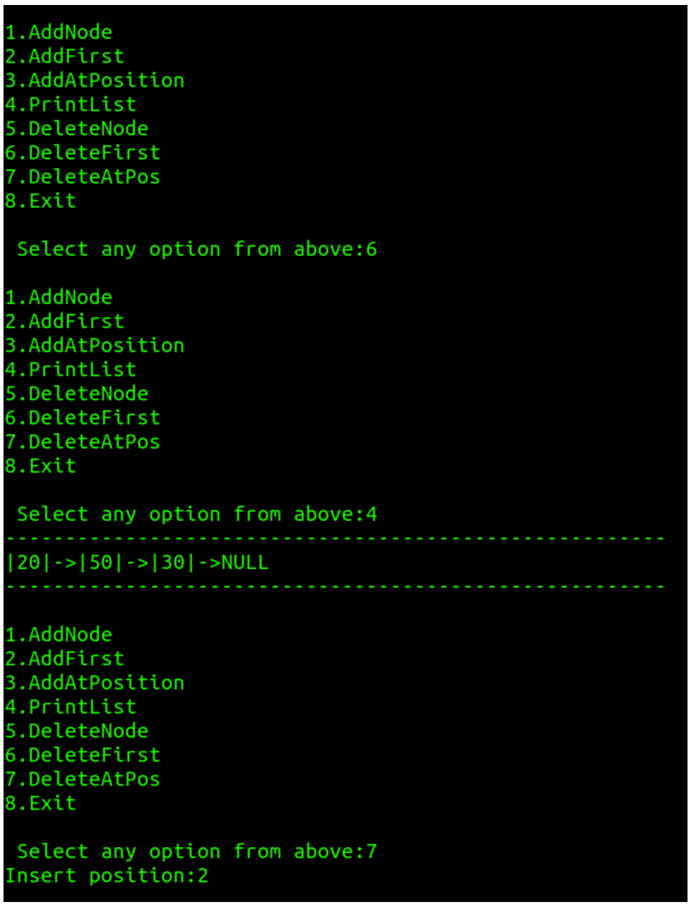
```
1.AddNode
2.AddFirst
3.AddAtPosition
4.PrintList
5.DeleteNode
6.DeleteFirst
7.DeleteAtPos
8.Exit

 Select any option from above:1
Enter data:10

1.AddNode
2.AddFirst
3.AddAtPosition
4.PrintList
5.DeleteNode
6.DeleteFirst
7.DeleteAtPos
8.Exit

 Select any option from above:1
Enter data:20

1.AddNode
2.AddFirst
3.AddAtPosition
4.PrintList
5.DeleteNode
6.DeleteFirst
7.DeleteAtPos
8.Exit

 Select any option from above:1
Enter data:30
```



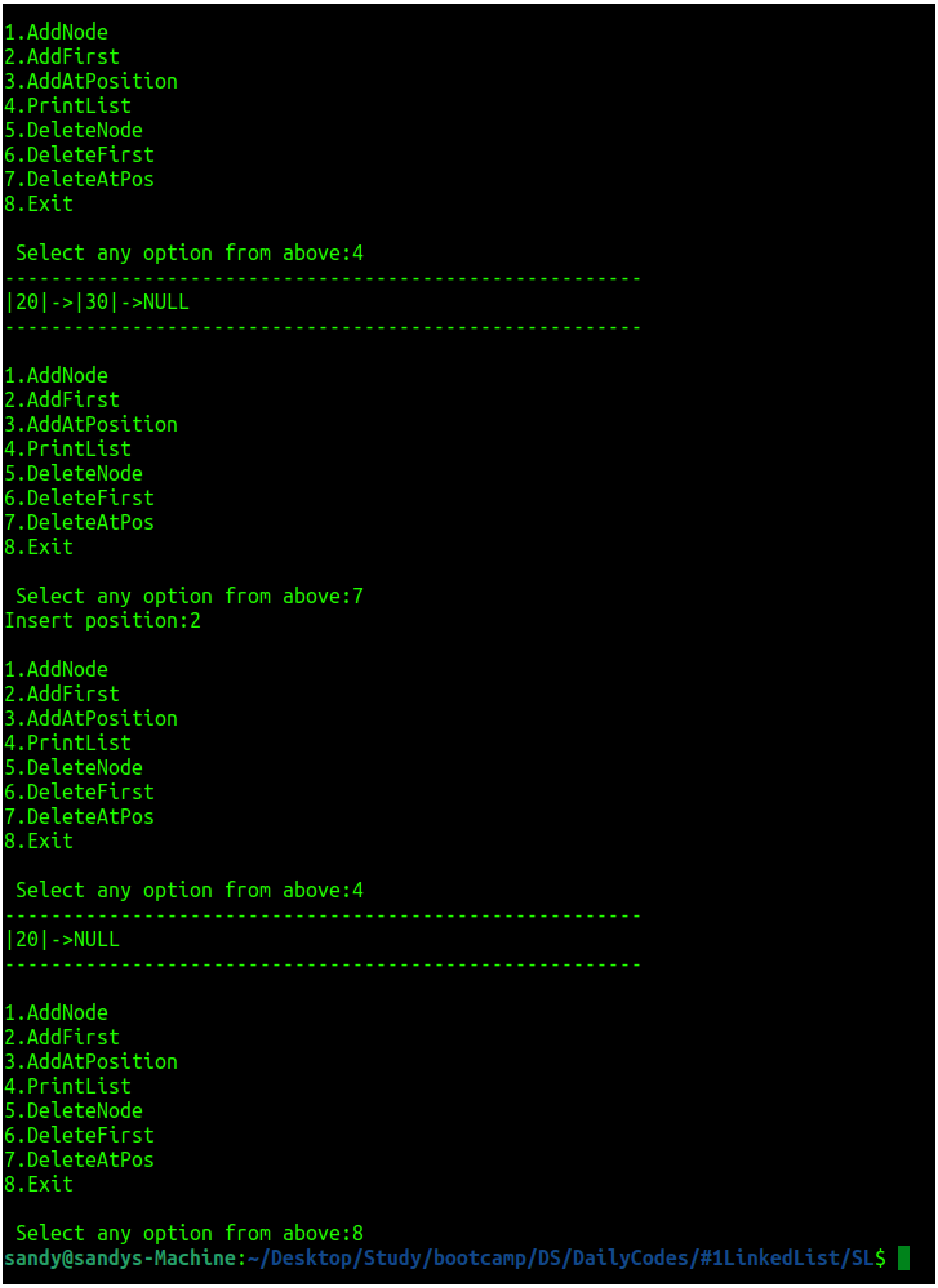
```
1.AddNode
2.AddFirst
3.AddAtPosition
4.PrintList
5.DeleteNode
6.DeleteFirst
7.DeleteAtPos
8.Exit

 Select any option from above:4
-------------------------------------------------------
|10|->|20|->|30|->NULL
-------------------------------------------------------

1.AddNode
2.AddFirst
3.AddAtPosition
4.PrintList
5.DeleteNode
6.DeleteFirst
7.DeleteAtPos
8.Exit

 Select any option from above:1
Enter data:40

1.AddNode
2.AddFirst
3.AddAtPosition
4.PrintList
5.DeleteNode
6.DeleteFirst
7.DeleteAtPos
8.Exit

 Select any option from above:3
Insert position:3
Enter data:50
```



```
1.AddNode
2.AddFirst
3.AddAtPosition
4.PrintList
5.DeleteNode
6.DeleteFirst
7.DeleteAtPos
8.Exit

 Select any option from above:4
-------------------------------------------------------
|10|->|20|->|50|->|30|->|40|->NULL
-------------------------------------------------------

1.AddNode
2.AddFirst
3.AddAtPosition
4.PrintList
5.DeleteNode
6.DeleteFirst
7.DeleteAtPos
8.Exit

 Select any option from above:5

1.AddNode
2.AddFirst
3.AddAtPosition
4.PrintList
5.DeleteNode
6.DeleteFirst
7.DeleteAtPos
8.Exit

 Select any option from above:4
-------------------------------------------------------
|10|->|20|->|50|->|30|->NULL
-------------------------------------------------------
```



```
1.AddNode
2.AddFirst
3.AddAtPosition
4.PrintList
5.DeleteNode
6.DeleteFirst
7.DeleteAtPos
8.Exit

 Select any option from above:6

1.AddNode
2.AddFirst
3.AddAtPosition
4.PrintList
5.DeleteNode
6.DeleteFirst
7.DeleteAtPos
8.Exit

 Select any option from above:4
-------------------------------------------------------
|20|->|50|->|30|->NULL
-------------------------------------------------------

1.AddNode
2.AddFirst
3.AddAtPosition
4.PrintList
5.DeleteNode
6.DeleteFirst
7.DeleteAtPos
8.Exit

 Select any option from above:7
Insert position:2
```



```
1.AddNode
2.AddFirst
3.AddAtPosition
4.PrintList
5.DeleteNode
6.DeleteFirst
7.DeleteAtPos
8.Exit

 Select any option from above:4
--------------------------------------------------------
|20|->|30|->NULL
--------------------------------------------------------

1.AddNode
2.AddFirst
3.AddAtPosition
4.PrintList
5.DeleteNode
6.DeleteFirst
7.DeleteAtPos
8.Exit

 Select any option from above:7
Insert position:2

1.AddNode
2.AddFirst
3.AddAtPosition
4.PrintList
5.DeleteNode
6.DeleteFirst
7.DeleteAtPos
8.Exit

 Select any option from above:4
--------------------------------------------------------
|20|->NULL
--------------------------------------------------------

1.AddNode
2.AddFirst
3.AddAtPosition
4.PrintList
5.DeleteNode
6.DeleteFirst
7.DeleteAtPos
8.Exit

 Select any option from above:8
sandy@sandys-Machine:~/Desktop/Study/bootcamp/DS/DailyCodes/#1LinkedList/SL$ 
```