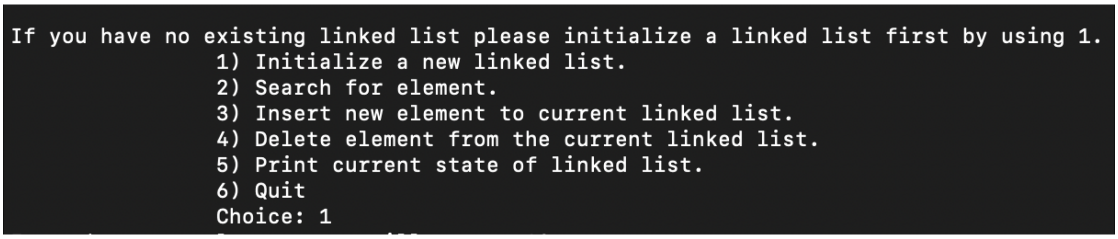
Data Structures HW1

Toygar Tanyel
18011094

March 28, 2022

My solution to the ineffectiveness of random choices from past stage: I jump 1 element while creating new stages. That helps us searching elements like binary search at the end. It's more robust solution than choosing random elements.

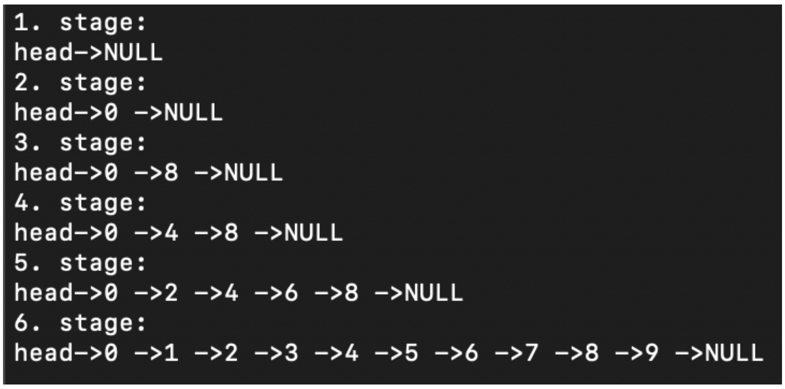
Menu:



```
If you have no existing linked list please initialize a linked list first by using 1.
                1) Initialize a new linked list.
                2) Search for element.
                3) Insert new element to current linked list.
                4) Delete element from the current linked list.
                5) Print current state of linked list.
                6) Quit
                Choice: 1
```

Figure 1 Menu for ease of use.

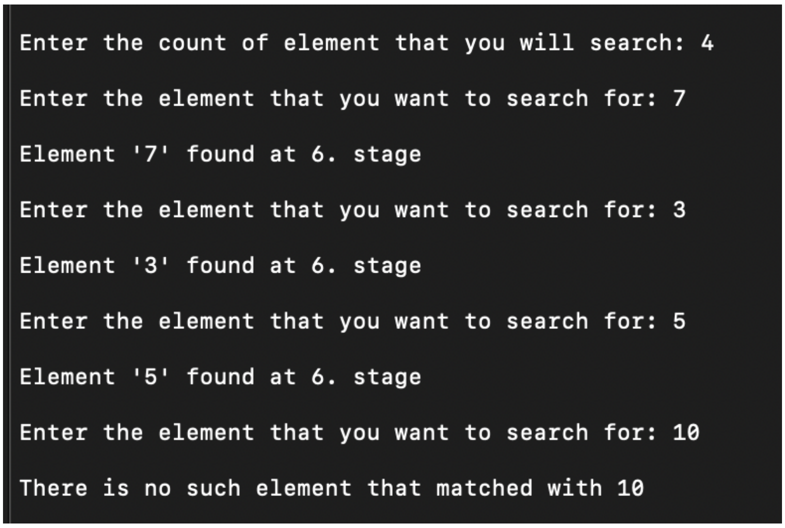
TASK1: Print all stages.



```
1. stage:
head->NULL
2. stage:
head->0 ->NULL
3. stage:
head->0 ->8 ->NULL
4. stage:
head->0 ->4 ->8 ->NULL
5. stage:
head->0 ->2 ->4 ->6 ->8 ->NULL
6. stage:
head->0 ->1 ->2 ->3 ->4 ->5 ->6 ->7 ->8 ->9 ->NULL
```

Figure 2 Initial linked list for [0 1 2 3 4 5 6 7 8 9].

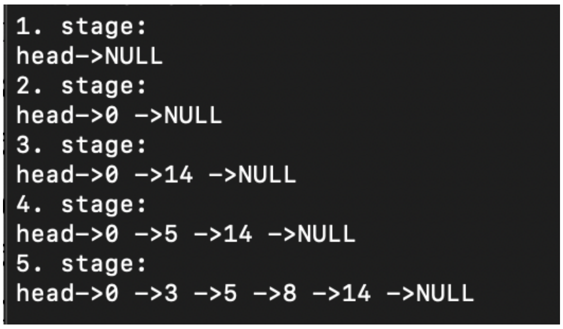
TASK2: Search for an element, if it found print its stage, else give a message like element not found. Give [0 1 2 3 4 5 6 7 8 9] as elements, and search for 7, 3, 5, 10



```
Enter the count of element that you will search: 4

Enter the element that you want to search for: 7

Element '7' found at 6. stage

Enter the element that you want to search for: 3

Element '3' found at 6. stage

Enter the element that you want to search for: 5

Element '5' found at 6. stage

Enter the element that you want to search for: 10

There is no such element that matched with 10
```

Figure 3 Search for 7, 3, 5, 10. Task 6.a

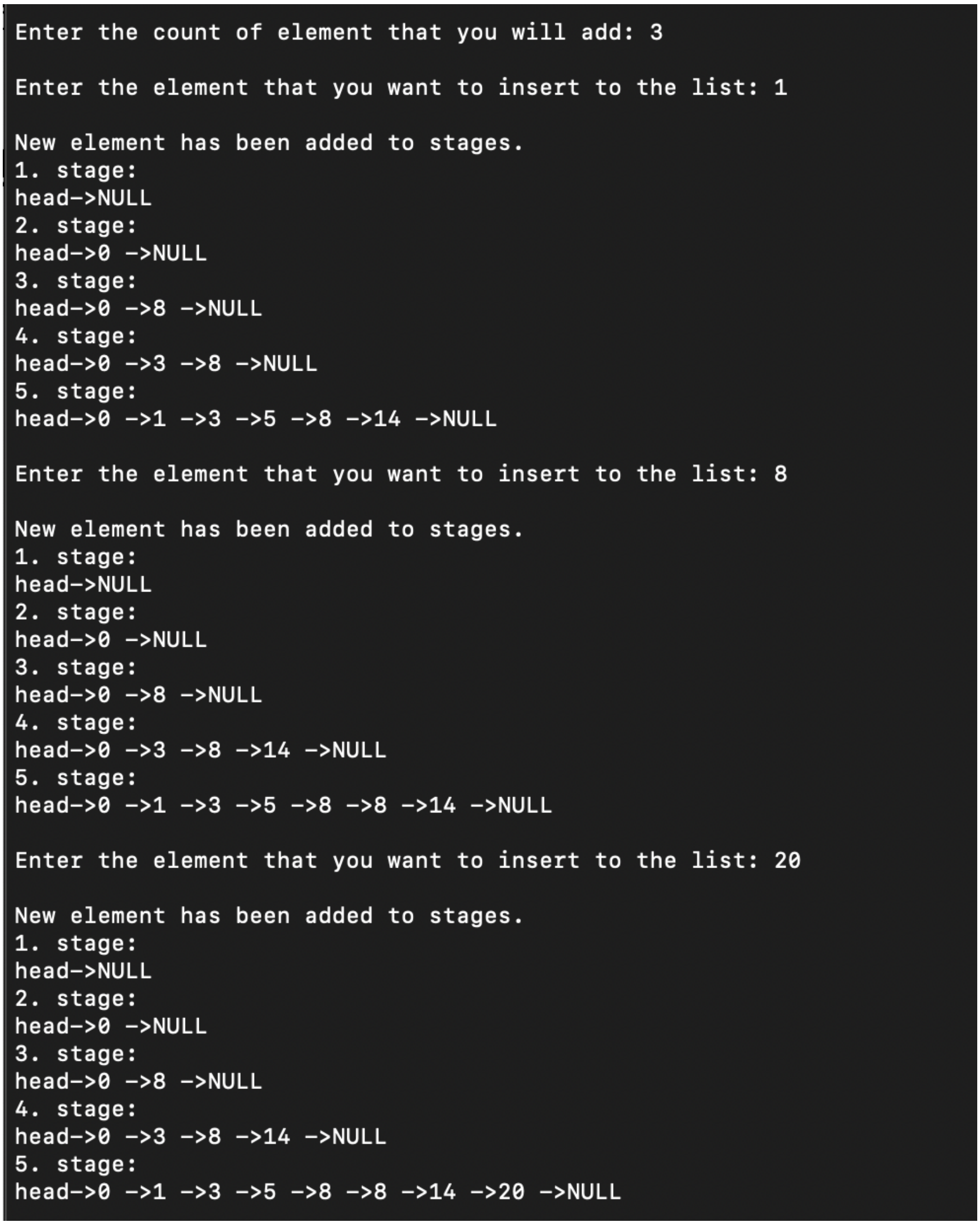
INITIALIZATION FOR TASK3:



```
1. stage:
head->NULL
2. stage:
head->0 ->NULL
3. stage:
head->0 ->14 ->NULL
4. stage:
head->0 ->5 ->14 ->NULL
5. stage:
head->0 ->3 ->5 ->8 ->14 ->NULL
```

Figure 4 Initialization for [0 3 5 8 14].

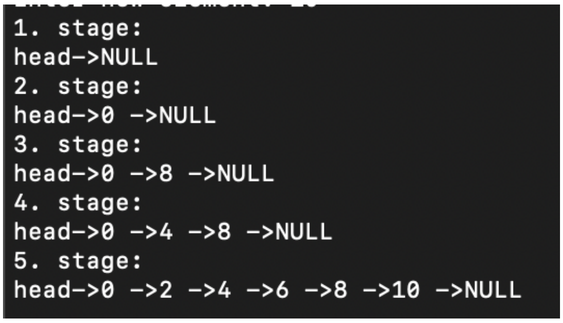
TASK3: Add new element to the linked list and update all stages. Give [0 3 5 8 14] as elements, add 1, 8 , 20 and print.



```
Enter the count of element that you will add: 3

Enter the element that you want to insert to the list: 1

New element has been added to stages.
1. stage:
head->NULL
2. stage:
head->0 ->NULL
3. stage:
head->0 ->8 ->NULL
4. stage:
head->0 ->3 ->8 ->NULL
5. stage:
head->0 ->1 ->3 ->5 ->8 ->14 ->NULL

Enter the element that you want to insert to the list: 8

New element has been added to stages.
1. stage:
head->NULL
2. stage:
head->0 ->NULL
3. stage:
head->0 ->8 ->NULL
4. stage:
head->0 ->3 ->8 ->14 ->NULL
5. stage:
head->0 ->1 ->3 ->5 ->8 ->8 ->14 ->NULL

Enter the element that you want to insert to the list: 20

New element has been added to stages.
1. stage:
head->NULL
2. stage:
head->0 ->NULL
3. stage:
head->0 ->8 ->NULL
4. stage:
head->0 ->3 ->8 ->14 ->NULL
5. stage:
head->0 ->1 ->3 ->5 ->8 ->8 ->14 ->20 ->NULL
```

Figure 5 Task 6.b

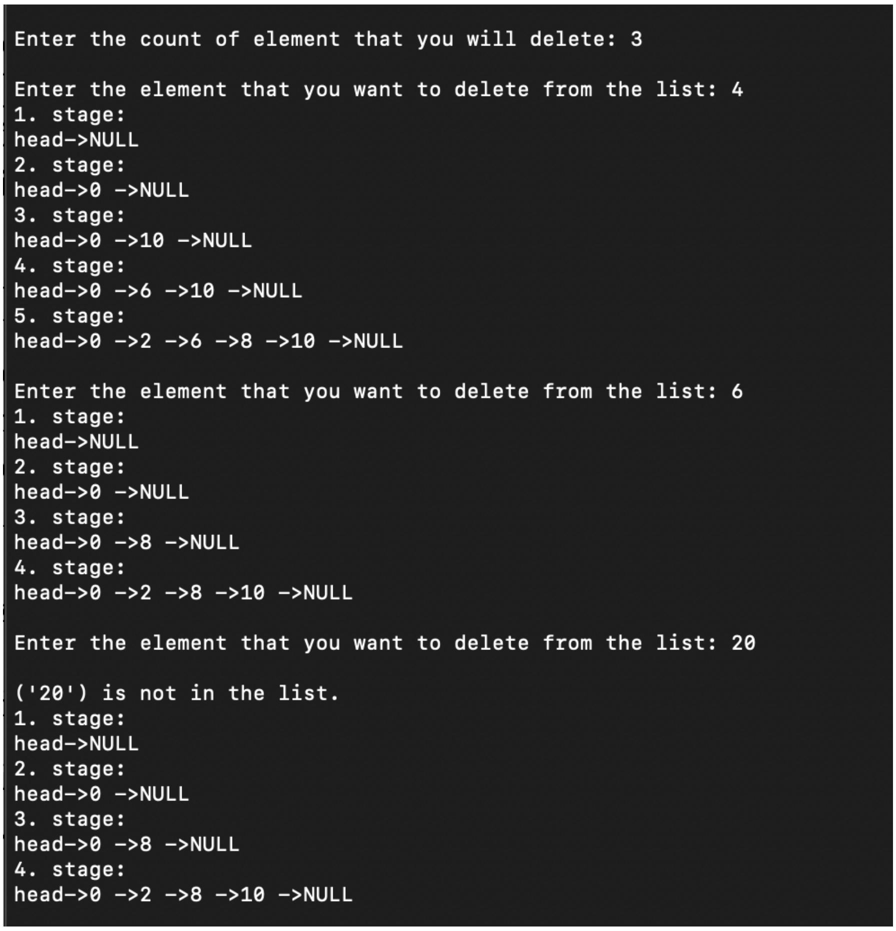
INITIALIZE FOR TASK4:



```
1. stage:
head->NULL
2. stage:
head->0 ->NULL
3. stage:
head->0 ->8 ->NULL
4. stage:
head->0 ->4 ->8 ->NULL
5. stage:
head->0 ->2 ->4 ->6 ->8 ->10 ->NULL
```

Figure 6 Initialization for deletion task

TASK4: Delete an element from linked list. Give [0 2 4 6 8 10] as elements, delete 4, 6, 20 and print.



```
Enter the count of element that you will delete: 3

Enter the element that you want to delete from the list: 4
1. stage:
head->NULL
2. stage:
head->0 ->NULL
3. stage:
head->0 ->10 ->NULL
4. stage:
head->0 ->6 ->10 ->NULL
5. stage:
head->0 ->2 ->6 ->8 ->10 ->NULL

Enter the element that you want to delete from the list: 6
1. stage:
head->NULL
2. stage:
head->0 ->NULL
3. stage:
head->0 ->8 ->NULL
4. stage:
head->0 ->2 ->8 ->10 ->NULL

Enter the element that you want to delete from the list: 20

('20') is not in the list.
1. stage:
head->NULL
2. stage:
head->0 ->NULL
3. stage:
head->0 ->8 ->NULL
4. stage:
head->0 ->2 ->8 ->10 ->NULL
```

Figure 7 New stages after deletion transactions.