Lab 11 - In Lab Assignment

Due end of the lab session

Acknowledge your collaborators or source of solutions, if any. **Submission by the end of the LAB is required.** Please type your answers, handwritten submission will not be accepted. Do all of the following.

1. **If a linked list contains three nodes with values "him", "her", and "its", and hp is a pointer to the list head, what is the effect of the following statements? Assume the data component of node type pro_node_t is pronoun, the link component is nextp, and np and mp are pointer variables.**

```
np = hp->nextp;
strcpy(np->pronoun, "she");
```

This code replaces "her" with "she".
It is because np is pointer to next node from head of the linked list. The current value of pronoun in that node is "her".

When the program uses this line:
`strcpy(np->pronoun, "she");`

So, this line copies the string "she" to chunk of memory pointed by np.
Hence, this code replaces "her" with "she

2. **When an element is deleted from a linked list, it is automatically returned to the heap. True or false? If false, please explain.**

When the element is deleted from the linked list. We only have lost the pointer to that element. So, the memory is not returned to heap automatically unless we call the free() function. We need to use free function to deallocate the memory.

free() function can deallocates the memory allocated by calloc, malloc or realloc function.

So, these are the step to delete node and return the memory from linked list:

1. find previous node of node to be deleted.
2. Change the next of the previous node.
3. Free memory for the node to be deleted.

If we forget to call the free() function, our program may suffer from memory leak issues.

3. **Often computers allow you to type characters ahead of the program's use of them. Should a stack or a queue be used to store these characters?**

When the computer allows us to type character ahead, all those characters must be stored in some data structures so that it can be used in the program later.
So, queue can be best solution because it is first in first out data structure.
Honestly, I think almost any data structure can be used for such requirement depending upon how the program use those store values.

For e.g., Pythons use tuple to store those command line argument. Java use string array to use those command line arguments.

But most natural and easy data structure would be queue.

Honestly, we could also use stack if we are making array inside the program to store all the data of stack and then use it in a program. But this approach would be long and tedious.