CSE 3010 – Data Structures & Algorithms

Lecture #18-19-20

What will be covered today

- Implementation of a list using singly linked list
- Implementation of a list using doubly linked list
- Practice exercises using linked list

Using singly linked list – Some operations on a list

1. Create a new node
2. Create an empty list
3. Get size of the list
4. Display the contents of the list
5. Add a node in the list
6. Delete a key (first occurrence) from the list
7. Delete a key from a given position
8. Find if a key exists in the list
9. Fetch a key given the position in the list
10. Get the position occupied by a key in the list
11. Delete all occurrences of a key from the list
12. Get the predecessor of a key in the list
13. Get the successor of a key in the list

Deleting a node in singly linked list

```
ListNode* deleteNode(LinkedList* list, ITEM key) {
        ListNode *curr, *prev, *deletedNode;
        prev = curr = list->head;
        deletedNode = NULL;

        if (curr != NULL) {
                if (curr->key == key) {
                        deletedNode = curr;
                        list->head = list->head->next;
                }
                else {
                        while (curr != NULL) {
                                if(curr->key == key) {
                                        prev->next = curr->next;
                                        deletedNode = curr;
                                        curr = curr->next;
                                }
                                else {
                                        prev = curr;
                                        curr = curr->next;
                                }
                        }
                }
        }
        return deletedNode;
}
```

Using doubly linked list

- Node contains two pointers
  - To the previous node
  - To the next node
- Pointer to the previous node of the head node is NULL
- Pointer to the next node of the last node is NULL
- Insertions and deletions require adjustment of previous and next pointers

Using doubly linked list – Some operations on a list

1. Create a new node
2. Create an empty list
3. Get size of the list
4. Display the contents of the list
5. Add a node in the list
6. Delete a key (first occurrence) from the list
7. Delete a key from a given position
8. Find if a key exists in the list
9. Fetch a key given the position in the list
10. Get the position occupied by a key in the list
11. Delete all occurrences of a key from the list
12. Get the predecessor of a key in the list
13. Get the successor of a key in the list

Practice exercises using linked list

1. Implement a set using linked list keeping the properties of a set. Additional operations to be implemented on a set are union, intersection and difference.
2. Implement a tuple (as defined in the Python programming language).
3. Implement merge function that takes two lists as arguments and merges the two files with the second file appended to the first file.
4. Implement split function that takes a list as an argument and a position in the list, and splits the list into two lists with the first list having nodes from the start until the node that equals the position, and the second list having the rest of the nodes.