Ajay Mittur
1BM18CS005

```
void insertElement(key) {
    Node *curr = head
    Node *update[maxLvl+1]
    for(int i = level → 0, i--) {
        while(curr->forward[i]->key < key)
            curr = curr->forward[i]
        update[i] = curr;
    }

    if(!(curr || curr->key != key)) {
        int rlevel = randomLevel();
        if (rlevel > level) {
            for (i = level+1 → rlevel, i++){
                update[i] = head;
            level = rlevel;
        }
        Node *n = createNode(key, rlevel);
        for ( i = 0 → rlevel; i++) {
            n->forward[i] = update[i]->forward[i];
            update[i]->forward[i] = n
        }
    }
```

Ajay Mittur
1BM18CS00L

```
void delete(key) {
    Node *curr = head, *update[maxlvl+1];
    for (i = level -> 0, i--) {
        while(curr -> forward[i] < curr -> forward[i]->key)
            curr = curr -> forward[i];
        update[i] = curr;
    }
    curr = curr -> forward[0];
    if (curr != NULL && curr -> key == key)
        for (i = 0 -> level; i++) {
            if(update[i] -> forward[i] != curr)
                break;
            update[i] -> forward[i] = curr -> forward[i];
        }
    while(level) {
        level--;
    }
}

Node* search(key){
    Node* curr = head;
    for(i = level -> 0; i--) {
        while(curr -> forward[i] && curr -> forward[i]->key < key)
            curr = curr -> forward[i];
    }
    curr = curr -> forward[0];
    return curr;
}
```