Office Hour #7

```
// Problem 1
// Prints all permutations of sentences from string arrays in linked list

struct Node {
    string alt[N];
    Node* next;
};

void permute(Node* head) {
    permuteAux(*head, "");
}

void permuteAux(Node * p, string prefix) {
    if (p == nullptr)
        cout << prefix << endl;

    for (int i = 0; i < n; i++)
        permuteAux(p->next, prefix + " " + p->alt[i]);
}
```

```
// Returns index of largest item in linked list

struct Node {
    int val;
    Node* next;
};

int positionOfBiggest(Node* head) {
    // if empty linked list, position is -1
    int maxSoFar = -1;
    biggestPositionAux(head, maxSoFar);
}

// int magicGetBiggest(Node* p, int& maxSoFar)

// returns index
int biggestPositionAux(Node* p, int & maxSoFar) {
    if (p == nullptr) {
        return -1;
    }

    if (p->next == nullptr) {
        maxSoFar = p->val;
        return 0;
    }
    int temp;
    int pos = biggestPositionAux(p->next, temp) + 1;
    if (p->val > temp) {
        maxSoFar = p->val;
        return 0;
    }
```

```
    else {
        maxSoFar = temp;
        return pos;
    }
}

int main () {
    // ... create new linked list
    positionOfBiggest(Node* head;)
}
```

Things to Keep in Mind:

- Recursive function always thinks its at the top node (node 0, element 0, etc.)
- Always update reference value passed in helper function
-