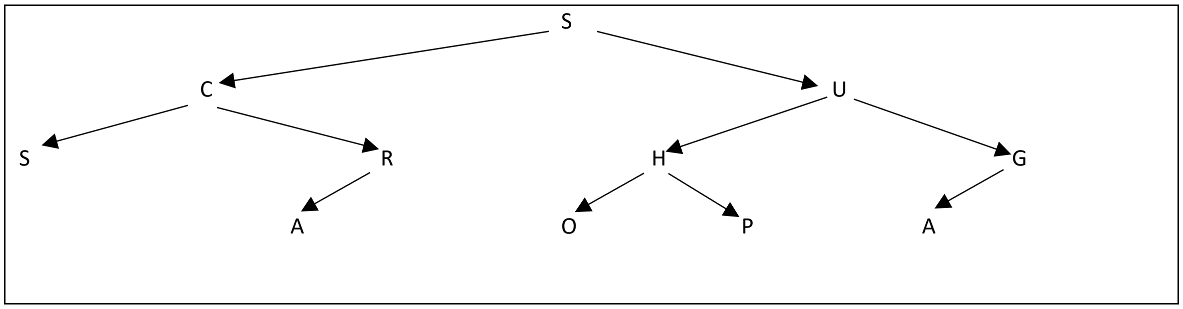
220 Halloween Chocolate Code-off:

Rules:

1. **No cell phones/computers**
2. **You may ask yes/no questions**
3. **You may use your notes**
4. **Competing for:**
    a. **Group with least incorrect problems: 3 e.c. pts on exam**
    b. **2nd least: 2 pts**
    c. **3rd least: 1 pt**
    d. **PLUS CHOCOLATE**
5. **Create one answer sheet with all group member names on it! (to be given to another group for grading**

NOTE: when strings are compared, "apple" is less than "bear" is less than "zebra". Think of what page a word would occur on in the dictionary (approximately). "apple" would occur on page 1, and "zebra" would occur on page 358. 358 is greater than 1, so zebra is greater than apple. (technically it's done based on ascii values of characters, but no one remembers that).
****

`Problem 1: (6 pts, one for each question)`
Given the following Tree:



a. Is it a Tree? ________________________
b. Is it a binary tree?_____________________
c. Is it a binary search tree? _________________
d. Is it balanced? ________________________
e. Is it complete? __________________________
f. Do a post-order traversal _________________________

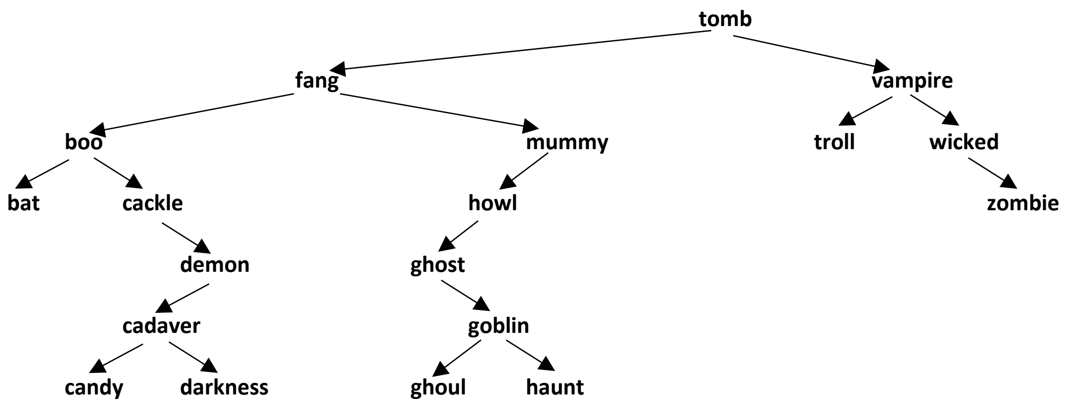
****

`Problem 2:` (1 pt, all or nothing) Assuming each word is inserted only once (meaning that if a word occurs more than once, ignore it from the second time on (meaning only insert 'of' the first time you come across it, and then ignore it)
Create a plain binary search tree out of the following list of words:

"eye of newt, toe of frog, wool of bat, tongue of dog, adder's fork, and blind worm's sting, lizard's leg, and owlet's wing"

****

Problem 3: (3 pts, all or nothing) Given the following plain old binary search tree, remove fang and show the resulting tree:



****
Problem 4: 2 pts, 1 for each line) the function below assumes you have removed a node from the binary search tree and now are trying to find the proper replacement node in the right subtree. Fill in the blanks so that the appropriate substituting node from the right subtree is returned (Note that this function ONLY finds and returns the appropriate replacing node – it does not implement the replace or anything else):

```
Node *replacefind(Node *n)
    if (________________ ) == NULL {
        Return n;
    }
    else  {
        ______________________;
        return replacefind( n);
    }
}
```

****
Problem 5 (1 pt) What is printed out in main?

```
int fa(int k[]) {
    int a = 0;
    for (int i = 0; i < 4; i++) {
        a += k[i];
    }
    return a;
}

int main() {
    int x[8] = {3,2,7,1,5,4,3,9};
    int y = fa(&x[3]);
    cout << y << endl;
}
```

****
Problem 6 (5 pts, all or nothing) Create an AVL Tree out of the following Phrase. Show the tree after each word has been inserted, including the height of each node and the balance of each node. Note, by definition, a leaf (or a node that has just been inserted) has a height of 1 and a balance of 0. When a tree is updated, the nodes affected in the rotation are updated by taking the height of the LEFT child minus the height of the RIGHT child. Once that height is updated, the parent is updated, then grandparent, etc., until the height of a node does not change. At that point we no longer have to continue updating heights and balances.

“darkness falls across the land midnight hour is close at hand”

****

Problem 7: (1 pt) Given the following linked list:
3->32->18->30->9->12

If you run the following function, then print out the linked list, what do you get?

```
void SLL::f3() {
    SNode *tmp;
    SNode *tmp2 = first;
    while (tmp2->next != NULL) {
        tmp = tmp2->next;
        tmp2->next = tmp2->next->next;
        tmp->next = first;
        first = tmp;
    }
    last = tmp2;
}
```

****

Problem 8: (1 pt) Assume that the Node's data type is string, and you have the following linked list:
pumpkin->morbid->scream->tomb->skull

What is printed in printit()?

```
SNode *SLL::f1(SNode *tmp, SNode *tmp2) {
    if (tmp == NULL) {
        return tmp2;
    }
    else {
        if (tmp->s > tmp2->s) {
                    f1(tmp->next,tmp);
              }
              else {
                    f1(tmp->next,tmp2);
        }
    }
}
void SLL::printit() {
    SNode *x = f1(first->next,first);
    cout << x->s << endl;
}
```

****
Problem 9: (1 pt) Assume:

```
class SNode {
    friend class SLL;
    char c;  // as opposed to int data;
    SNode *next;
};
```

Given the following linked list,

a->c->b->r->t->y->o->p-v->t->

If you run the following function, then print out the linked list (characters), what do you get?

```
void SLL::func4() {
    SNode *tmp = first->next;
    first = first->next;
    while ((tmp->next != NULL)&&(tmp->next->next != NULL)) {
        tmp->next = tmp->next->next;
        tmp = tmp->next;
    }
    if (tmp->next != NULL) {
        tmp->next = NULL;
    }
    last = tmp;
}
```

****
Problem 10: (1 pt)
Given the following linked list:
35->10->35->47->9->26->42->35->17->24->8->

If you run the following function, then print out the linked list, what do you get?

```
void SLL::sj() {
    SNode *tmp = first;
    for (int i = 0; i < size/2; i++) {
        tmp = tmp->next;
    }
    last->next = first;
    first = tmp->next;
    tmp->next = NULL;
}
```

****
Problem 11: (1 pt)
Assume:

```
class DNode {
    friend class DLL;
    char c;  // as opposed to int data;
    DNode *next;
    DNode *prev;
};
```

Assume you have the following linked list:
al->chr->pu->se->l->

If you run the following function, then print out the linked list, what do you get?

```
void DLL::f3() {
    DNode *tmp3= first->next;
    DNode *tmp2;
    while (tmp3 != NULL) {
        DNode *tmp = tmp3;
        tmp2 = tmp->prev;
        while (tmp2 != NULL && tmp2->c < tmp->c) {
            tmp2 = tmp2->prev;
        }
        tmp3 = tmp3->next;
        if (tmp->prev != tmp2) {
            if (tmp->next == NULL ){
                last = tmp->prev;
            }
            if (tmp->next != NULL) {
                tmp->prev->next = tmp->next;
                tmp->next->prev = tmp->prev;
            }
            else {
                tmp->prev->next = NULL;
            }

            if (tmp2 != NULL) {
                tmp->next = tmp2->next;
                tmp->prev = tmp2;
                tmp2->next = tmp;
                tmp->next->prev = tmp;
            }
            else {
                tmp->next = first;
                first->prev = tmp;
                first = tmp;
                first->prev = NULL;
            }
        }
    }
}
```

****

Problem 12: (1 pts)

Assume:

```
class SNode {
    friend class SLL;
    int data;
    char c;  // as opposed to int data;
    SNode *next;
};
```

Given the following list:

op->bi->tr->s->a->ho->pec

If you run the following function, then print out the list characters, what do you get?

```
void SLL::MakeIt() {
    last->next = first;
       SNode *tmp = first;

    int ct = 0;
    int s2 = 0;
    SNode *tmp2;
    while (size > 0) {
        if (ct ==2) {
            if (s2 == 0) {
                first = tmp->next;
                tmp->next = tmp->next->next;
                first->next = NULL;
                tmp2 = first;
            }
            else {
                tmp2->next = tmp->next;
                tmp->next = tmp->next->next;
                tmp2->next->next = NULL;
                tmp2 = tmp2->next;
            }
            s2++;
            ct = 0;
            size--;
        }
        ct++;
        tmp = tmp->next;
    }
    last = tmp2;
}
```

```
**************************************************************************
```
Problem 13: (1 pt)

```
Assume you have two linked lists as follows:
list1 is b->g->e->z->h->d->o->u->v->l->
list2 is s->g->r->h->t->o->p->u->x->l->

And
SLL *n = new SLL();
SNode *tmp = list1->first;
SNode *tmp2 = list2->first;

If you run the following function with:
is(tmp,tmp2,n);
then print out the new list (n), what do you get?


void SLL::is(SNode *tmp, SNode *tmp2, SLL *n) {
    if (tmp2 == NULL) {
        first = n->first;
        size = n->size;
        last = n->last;
        return;
    }
    else if (tmp == NULL) {
        return(is(first,tmp2->next,n));
    }
    else if (tmp->c == tmp2->c) {
        if (n->size == 0) {
            n->first = new SNode(0);
            n->first->c = tmp->c;
            n->last = n->first;
        }
        else {
            n->last->next = new SNode(0);
            n->last->next->c = tmp->c;
            n->last = n->last->next;
        }
        n->size++;
        n->last->next = NULL;
        return(is(first, tmp2->next, n));
    }
    else {
        return(is(tmp->next, tmp2, n));
    }
}
```

****
Problem 14: (3 pts) Assume the linked list is as follows:

g->a->m->h->o->y->n->e->r->

and in main, the function func2 is called as follows:

```
list->func2(3,4);
list->printSLL();
```

func2 is defined below. What would be printed out?

```
void SLL::func2(int sk, int tsk) {
    SNode *tmp = first;
    SNode *tmp2 = tmp;
    tmp= tmp->next;
    last->next = tmp;
    first = tmp2;
    tmp2->next = NULL;
    int ct = 1;
    int ct2 = 1;
    while (ct2 < tsk ) {
        while (ct < sk) {
            tmp = tmp->next;
            ct ++;
        }
        tmp2->next = tmp->next;
        tmp->next = tmp->next->next;
        tmp2 = tmp2->next;
        tmp2->next = NULL;
        ct = 0;
        ct2++;
    }
    size = tsk;
    last = tmp2;
}
```