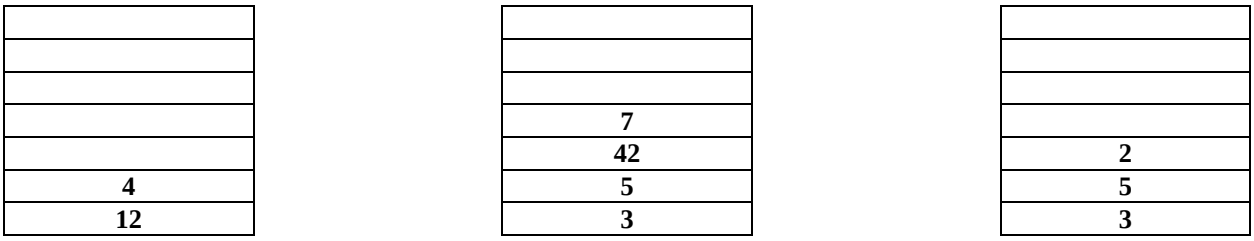
Sample COP 3502H Exam #2 Solutions

1) (10 pts) Evaluate the following postfix expression, showing the state of the operand stack at the three points A, B and C indicated below:

				A				B				C		
12	6	2	-	/	5	42	7	/	4	-	*	+		



A B C

Value of the Expression: **13**

Grading: 2 pts for each stack, 4 pts for the answer. May give partial if a single error caused a propagation error. (Take off 1 pt per actual error.)

2) (10 pts) Circle either True or False about each of the following assertions about queues.

a) A queue is a Last In, First Out (LIFO) abstract data structure. True **False**

b) If a queue is implemented with a regular linked list, with a pointer to the front of the queue only, the enqueue operation would take Θ(n) time for a list with n elements. (Θ indicates proportional to n.) **True** False

c) If a queue is implemented with a regular linked list, with a pointer to the front of the queue only, the dequeue operation would take Θ(n) time for a list with n elements. True **False**

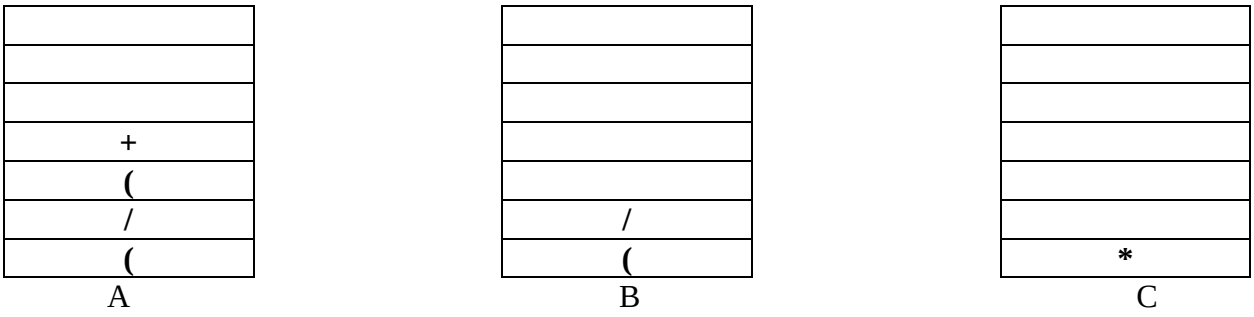
d) A queue must be implemented with a linked list. True **False**

e) A queue allows for access to any of its elements in O(1) time. True **False**

Grading: 2 pts each, all or nothing

3) (10 pts) Convert the following infix expression to postfix, showing the state of the operator stack at the three points A, B and C indicated below:

```
                        A                 B                 C
( ( 42 - 9 ) / ( 2 + 5 - 2 * 2 ) - 2 * 3 ) * ( 3 + 4 )
```



A B C

Equivalent Postfix Expression:

42 9 - 2 5 + 2 2 * - / 2 3 * - 3 4 + *

Grading: 2 pts for each stack, 4 pts for the expression, if a single error propagates, take off 1 pt per error instead

Consider implementing a binary search tree where at each node, in addition to storing a string and pointers to both the left and right child, the height of the node is stored, as well as the total number of nodes in the subtree rooted at the node. The struct defintion for a node in the tree is provided below.

```
#include <string.h>
#define MAX 20

typedef struct bintreenode {
    char word[MAX];
    int height;
    int numNodes;
    struct bintreenode* left;
    struct bintreenode* right;
} bintreenode;
```

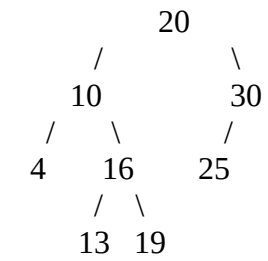
4) (10 pts) Write a **recursive** search function that takes in a word and a pointer to the root of a binary search tree, and returns the height of the first node encountered that stores the word. If no such node exists, your function should return -1.

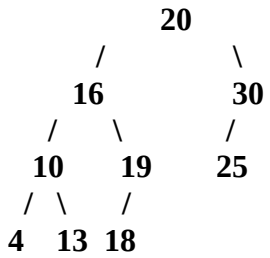
```
int search(bintreenode* root, char myword[]) {

    // 2 pts
    if (root == NULL) return -1;

    // 3 pts
    if (strcmp(root->word, myword) == 0) return root->height;

    // 3 pts
    if (strcmp(myword, root->word) < 0)
        return search(root->left, myword);

    // 2 pts
    else
        return search(root->right, myword);









}
```

5) (15 pts) Implement a **recursive** insert function for this tree. If the word being inserted is less than **or equal** to a word in a node in the tree, it should be inserted in the left subtree of that node. Otherwise, it should be inserted in the right subtree. Use the partially filled out function provided below. (Note: assume stdlib.h and string.h are included.)

```
int max(int a, int b);

bintreenode* insert(bintreenode* root, char newword[]) {

    if (root == NULL) {
        bintreenode* tmp = malloc(sizeof(bintreenode));

        strcpy(tmp->word, newword) ;          // 2 pts, 1pt for =

        tmp->height = 0 ;                     // 1 pt

        tmp->numNodes = 1 ;                   // 1 pt
        tmp->left = NULL;
        tmp->right = NULL;
        return tmp;
    }

    if (strcmp(newword, root->word ) <= 0)    // 1 pt

        root->left = insert(root->left, newword); // 3 pts
    else

        root->right = insert(root->right, newword);// 3 pts

    root->numNodes = root->numNodes + 1;       // 2 pts
    int leftH = root->left == NULL ? -1 : root->left->height;
    int rightH = root->right == NULL ? -1 : root->right->height;
    root->height = 1 + max(leftH, rightH);     // 2 pts
}

int max(int a, int b) {
    return a > b ? a : b;
}

/*** Partial List of String Functions
strlen(s) - returns the length of string s
strcpy(s,t) - copies the contents of string t into string s.
strcmp(s,t) - returns a negative integer if s comes before t,
              lexicographically, 0 if s equals t, and a positive
              integer if s comes after t lexicographically.
***/
```

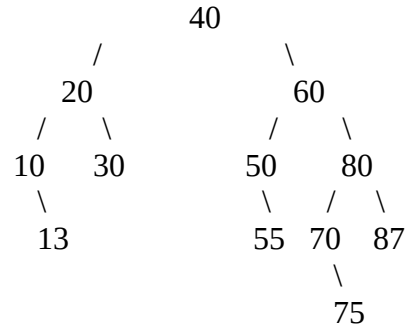
6) (6 pts) Show the final result of inserting the value 18 into the AVL tree shown below. Draw a box around your final answer.



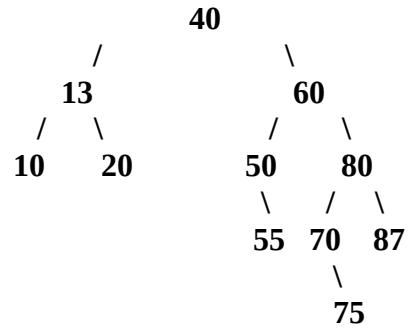


Grading - 2 pts out of 6 if rotation is done on (20,10,16).
3 pts if rotation done on (10,16,19) , 3 pts for 4, 13 and 18.

7) (8 pts) Show the final result of deleting the value 30 from the AVL tree shown below. Draw a box around the intermediate answer after the first rebalance and a separate box around your final answer (after the second rebalance). Label both boxes clearly

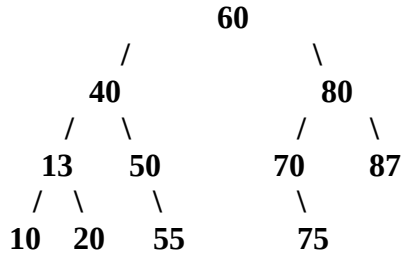


After first rebalance:



Grading: 4 pts total - 1 pt for 40 and right being unchanged, 1 pt for each of 10, 13, 20.

Final answer (after 2nd rebalance):



8) (1 pt) The John C. Hitt Library is named after what UCF President? **John C. Hitt**