

Project 1, CSci 330, Fall 2006

Due: Sep 8, 2:10pm. Value: 60 pts.

Suppose we represent a binary tree of nonnegative integers, where each node has the node value in the first four bytes, the left node's address in the next four bytes, and the right node's address in the next four bytes. When there is no left node (or right node), we place 0 in the relevant bytes. **Note:** We are not working with binary search trees, and so values can appear in any order within the tree.

You will compare the below two techniques for computing the maximum value in such a tree.

Recursive approach

```
int max(node *n) {
    if(n == NULL) {
        return -1;
    } else {
        a = max(n->left)
        b = max(n->right)
        ret = n->value
        if(a > ret) ret = a;
        if(b > ret) ret = b;
        return ret;
    }
}
```

Iterative approach

```
int max(node *root) {
    if(root == NULL) return -1;
    n = root;
    ret = -1;
    while(1) {
        if(n->value > ret) ret = n->value;
        if(n->left != NULL) {
            if(n->right != NULL) push(n->right);
            n = n->left;
        } else if(n->right != NULL) {
            n = n->right;
        } else if(!stackIsEmpty()) {
            n = pop();
        } else {
            return ret;
        }
    }
}
```

Write both in MIPS assembly language; your solutions should be optimized while staying within the rough recursive/iterative approach above. (You can feel free to modify the above, particularly if it gives you greater efficiency; you just need to follow the rough concept of using recursion or using a loop.) As with Assignment 1, you can download a testing framework from the course Web page.

Also, write a short analysis of these two techniques' relative merits. That is, why would you ever opt for the first approach? Or the second approach? Be quantitative in your analysis, referring to what you can glean from your MIPS implementations. Although length will not be a factor in my evaluation, it is certainly possible to do an excellent analysis with two pages of single-spaced text.

You should write the analysis as if it were being submitted as a technical paper. In particular, it should not be written as if it were an assignment ("I was assigned to..."), and it should make sense to a knowledgeable computer scientist without having to read this assignment handout or your MIPS implementations.

Your grade will be based on the following components.

Code correctness (25%) As on Assignment 1.

Code efficiency (10%) As on Assignment 1.

Code documentation (15%) As on Assignment 1.

Analysis content (30%) It should be as complete and as insightful as possible, demonstrating a substantive analysis of the MIPS assembly code.

Analysis style (20%) The analysis should be engaging, well-composed, and well-written.

Submit your code solution by e-mail to burch@grendel.hendrix.edu. While you are free to include your analysis as an attachment also, I will only grade your analysis if submitted on paper.