

CS 14 - Summer 2004 - Quiz 1

August 9, 2004

1 Multiple Choice

Fill in the bubble of the single-best answer for each question.

Do NOT make any unrequested marks on your answer sheet. You may use pen or pencil, but if you want to change an answer, you MUST erase completely. Marks that are "X"ed out will be counted as filled.

1. Any sorting algorithm that accesses the list in sequential (not just skipping around) order can be done on a Linked List with an iterator.
 - (a) True
 - (b) False
2. The following is code for which sorting algorithm?

```
void mysterySort(vector<int> nums)
{
    for (unsigned int i = 0; i < nums.size(); i++)
    {
        for (unsigned int j = 0; j < nums.size() - i - 1; j++)
        {
            if (nums[j] > nums[j + 1]) swap(nums[j], nums[j + 1]);
        }
    }
}
```

- (a) Insertion Sort
- (b) Bubble Sort
- (c) Quick Sort
- (d) Merge Sort
- (e) Radix Sort

3. The following is a description of which sorting algorithm?

Pick an element of the list as a "pivot". Move everything less than the pivot to one side of it, and everything greater than to the other side, and recursively sort those two sides.

- (a) Insertion Sort
- (b) Bubble Sort
- (c) Quick Sort
- (d) Merge Sort
- (e) Radix Sort

4. The following code will do what?

```
struct Node
{
    int val;
    Node* left;
    Node* right;
};

Node* insert(Node* root, int val)
{
    if (val < root->val) root->left = insert(root->left, val);
    if (val > root->val) root->right = insert(root->right, val);
    return root;
}

int main()
{
    Node* root = insert(NULL, 10);
    root = insert(root, 5);
    root = insert(root, 3);
    root = insert(root, 12);
}
```

- (a) Build a binary search tree with values 10, 5, 3, and 12, with 12 as the root.
- (b) Build a binary search tree with values 10, 5, 3, and 12, with 10 as the root.
- (c) Go into an infinite recursion.
- (d) Segfault
- (e) Build a linked-list with values 10->5->3->12.

5. The worst-case time to find an element in a binary search tree of n nodes is

- (a) $O(1)$
- (b) $O(\log n)$
- (c) $O(n)$
- (d) $O(n \log n)$
- (e) $O(n^2)$

6. This is the proper code for MergeSort if you have already written Merge:

```
void MergeSort(vector<int> nums, int begin, int end)
{
    int mid = (begin + end) / 2;
    Merge(nums, begin, end);
    MergeSort(nums, begin, mid);
    MergeSort(nums, mid, begin);
}
```

- (a) True
- (b) False

7. Since Radix Sort is $O(n)$ and Bubble Sort is $O(n^2)$, Radix Sort is ALWAYS faster than Bubble Sort.

- (a) True
- (b) False

8. Quick Sort has a worst case runtime of $O(n \log n)$

- (a) True
- (b) False

9. Merge Sort cannot be used efficiently without allocating additional memory.

- (a) True
- (b) False

10. The proper evaluation of this postfix expression is what?

5 4 + 3 2 + *

- (a) 15
- (b) 19
- (c) 25
- (d) 30
- (e) 45
- (f) 60