CSCI 2270
Data Structures and Algorithms
Lecture 11

Elizabeth White
elizabeth.white@colorado.edu
Office hours: ECCS 128
Wed 1-2pm
Thurs 2-3pm

Admin

HW1 due on Monday, 11:55 pm: Singly Linked List
This will be *interview graded.*

Good colloquium talk next week, 3:30-4:30 Thursday, ECCR 265

Exam review: posted questions for written part
  slides and quiz questions, labs, homework
  insertion sort example

Programming exam: in lab next Tuesday/Thursday.
  2 problems will be posed; you will answer one
Written exam: in class next Friday

Not clearing a linked list

Another way… THIS SLIDE WAS WRONG (thanks to the 2 students who argued with me after class!)

```
void clear_list(node*& head_ptr)
{
        if (head_ptr == nullptr) return;
        if (head_ptr->next == nullptr)
        {
                delete head_ptr; head_ptr = nullptr; return;
        }
        clear_list(head_ptr->next);
}
```

Actually clearing a linked list

Another way…

```
void clear_list(node*& head_ptr)
{
        if (head_ptr == nullptr) return;
        clear_list(head_ptr->next);
        delete head_ptr;
        head_ptr = nullptr;
        return;
}

// check by printing head_ptr->data before deleting
```

Recursively printing a linked list

```
void print_list2(const node*& source_ptr)
{
        if (source_ptr == nullptr) return;
        else
        {
                cout << source_ptr->data << endl;
                const node* cursor = source_ptr->next;
                print_list2(cursor);
        }
}
```

Recursively printing a linked list

```
void print_list2(const node*& source_ptr)
{
        if (source_ptr == nullptr) return;
        else
        {
                const node* cursor = source_ptr->next;
                print_list2(cursor);
                cout << source_ptr->data << endl;
        }
}
```

Recursion

Tail recursion is best, when recursive call is last computation

This code is not tail recursive: multiplication happens after recursive call

```
unsigned int factorial_slow(unsigned int n)
{
        if (n == 0)
                return 1;
        return n * factorial_slow(n - 1);
}
```

Recursion

Tail recursive version (one function is recursive, one is not)

```
unsigned int factorial1(unsigned int n,
                        unsigned int accumulator)
{
    if (n == 0)
        return accumulator;
    return factorial1(n - 1, n * accumulator);
}
unsigned int factorial_fast(unsigned int n)
{
    return factorial1(n, 1);
}
```

Practice questions for exam…

Write me a function that takes 2 unsigned ints, m and n, and creates a 2-dimensional array of integers on the heap with m rows and n columns.

Write me a function that destroys the above 2-dimensional array.

Write me a function that tells me if 2 sorted int_arrays have exactly the same numbers.

Write me a function that tells me if 2 unsorted int_arrays have exactly the same numbers. Assume you can’t sort the arrays. (Why is this so much harder?)

Practice…

Write me a function to convert a decimal number to a string of binary digits. This requires working out how many digits you will need, and if you use the algorithm on my slides, you'll need to write the digits into the string backwards. Return the string.

Write me a function to convert a string representing a binary number to a decimal number. Return this number as an int.

Write me a function void minimum(bool& empty, int& answer) to find the smallest number in an unsorted int_array's data if the array's not empty, and return 0 if it is. Return answer (the minimum number) and empty (true if array's empty) by reference.

Practice…

Write me a function that takes an int_array as input, and then makes a new array 10 times larger, with the contents of the original data array repeated 10 times. Replace the original int_array's data with this new array, but don't leak memory.

Write me a recursive function to copy a singly linked list (from the end node backwards.)

Write me a recursive function to count the number of times a target integer appears in a singly linked list.

Practice…

~~Write me a recursive function to take a singly linked list and return a new list with the same elements as the old one, but in reverse order.~~ NOT INTERESTING.