Look for email with problem I grade tonight.

One conflict exam left, so tests will be.

Check point is next Monday.

Problem A is due next Thurs.

Announcements

CS180 - Lecture 13
Last time - linked lists

head

value

pointer to next element

LAX - MSP - ATL - STL

ptr = NULL
Node (const Object &e = Object(), Node *n = NULL):
    // Constructor
    next (n), element (e), next next (n)

    // ptr to next node
    next 
    // value of this node
    element

Our Struct for a Node
Disadvantage: - takes extra space

Advantage: - arbitrary size

A new stack.
A version of a stack which uses

Linked Stack
Code: for LinkedList

```
int size;
Node *top;
```

Private data:

Why?

"protected" (instead of public/private)

Our node structure will be included as

protected is essentially same as private.
bool empty() const { return size == 0; }

int size() const { return size; }

void push(int x) {
    if (size < capacity) {
        arr[size] = x;
        size++;
    }
}

void pop() {
    size--;
}

int top() const {
    return arr[size - 1];
}

class Stack {
private:
    int capacity;
    int size;
    int* arr;
public:
    Stack(int c) : capacity(c), size(0) { arr = new int[c]; }
    ~Stack() { delete[] arr; }

    void push(int x) {
        if (size < capacity) {
            arr[size] = x;
            size++;
        }
    }

    void pop() {
        size--;
    }

    int top() const {
        return arr[size - 1];
    }

    bool empty() const { return size == 0; }

    int size() const { return size; }

    int* arr;
};

// Functions (Easy Ones)
3. return fp->element;
   throw std::runtime_error("Stack empty");

   if (empyfy())
     cost + Object::top(c) const
     return fp->element;
   throw std::runtime_error("Stack empty");

   Object::top(c) const
   return fp->element;

   a
   another way: (c)
   return fp->element;
   throw std::runtime_error("Stack empty");

   if (empyfy())
     cost + Object::top(c) const
   return fp->element;

   What is our choice here?
   Top function (2 versions)
Push + Pop

```cpp
void push(const Object& e) {
    Node* new_node = new Node(e, top);
    top = new_node;
    size++;
}

void pop() {
    if (empty())
        throw std::runtime_error("throw error - see top");
    Node* old = top;
    top = top->next;
    delete old;
    size--;
}
```
- Assignment speeches
- Copy constructor
- Destructor

What else do we need to worry about?

"Please keep in mind"