Announcements

- HW due Wednesday (in pairs, to me)
- Lab tomorrow - due Friday (no prelab)
- Midterm 1 in 1 week - review on Friday -> bring questions!
HW 3 Recap:
- be careful w/ pointers & templates
- don't alter your private data!

```cpp
Object minimum() const {
    Node* temp = _head;
    Object minsofar;
    while (temp != NULL) {
        if (temp->date < minsofar)
            minsofar = temp->date;
        temp = temp->next;
    }
    return minsofar;
}
```
Can you return `const Object`?
Can't return variable created in function.

But:

```c
SNODE* minsofar = head;
while (not NULL) {  
    if (minsofar->data > temp->data)  
        minsofar = temp;  
    temp = temp->next;  
}
return minsofar->data;  
```
Last time:

Vector running times:

Linear time: $O(n)$

(except for size, empty + other $O(1)$ - time adding)
But: Is it really that bad?

Consider a sequence of push-back operations.

Runtime: \( n \) push-backs, each takes \( \mathcal{O}(n) \) time

\[ \Rightarrow \mathcal{O}(n^2) \]

Really? \[ \sum_{i=1}^{n} i = \mathcal{O}(n^2) \]

But:

When do we actually double?

Only to be linear time when doubling a/later doubling, half empty!
Amortization

Every time we have to rebuild the array we get a bunch of extra spots. Need to formalize this idea:

Amortization: finding average running time per operation over a long series of operation.
Claim: The total time to perform a series of $n$ push-back operations into an initially empty vector is $O(n)$.

proof: Think of a bank account. Each constant time operation costs $1$ to run.

So each non-overflow push costs $1.

Overflow inserts?

$1 + \$n$
Key idea: overcharge the non-overflow pushes
Analysis: array has \(2^i\) elements in it and needs to be doubled

Last double had \(2^{i-1}\), so a \(2^{i+1}\) total of \(2^{i-1}\) new things have been inserted since then.

Each gave \(\$3\) a cost \(\$1\)

\[3 \cdot 2^{i-1}\]

in bank

\[-2^{i-1} = 2 \cdot 2^{i-1}\]

left in bank

"enough" in bank to cover doubling.
Vector class:

What's left: House keeping

~20 other functions

& HW next week