CS 162
Intro to CS II

STL/Templates
Odds and Ends

• Turn in your extra credit from Expo
• Keep working on Assignment 4
• Assignment 4 questions
Why Function Templates?

//at least C++ has overload
void swap(int &, int &);
void swap(char &, char &);
...
void swap(int &a, int &b) {
    int temp = a;
    a = b;
    b = temp;
}
void swap(char &a, char &b) {
    char temp = a;
    a = b;
    b = temp;
}
Function Template...

//Have to have this header

template<class T>
void swap(T &, T &);

...

template<class T>
void swap(T &a, T &b){
    T temp = a;
    a=b;
    b=temp;
}
When can you get into trouble?
//Have to have this header

template<class T>
void func(T *, T *, int);
...

template<class T>
void func(T a[], T b[], int size){
    //a is already a reference/static array
    //what if we wanted to swap values in arrays
}

OSU Oregon State University
Why make a class templated?

• What functionality have we written so many times, regardless of type?
Class Templates

//Have to have this header

template<class T>
class vector {

public:
    vector();
    ~vector();
    void push_back(T);

private:
    T *v;
};
//Have to have this header
template<class T>
vector<T>::vector(){
    v=NULL;
}
template<class T>
vector<T>::~vector(){
    delete [] v;
}
template<class T>
void vector<T>::push_back(T element){
    ...
}
Using pre-defined vector class...

- What is size() vs. capacity()
- What is resize() vs. reserve()?
- How does push_back() work?
```cpp
#include <iostream>
#include <vector>
using namespace std;

int main() {
    vector<int> v;

    for(int i=0; i<5; i++) {
        v.push_back(10);
        cout << "size: " << v.size() << endl;
        cout << "capacity: " << v.capacity() << endl;
    }

    for(int i=0; i<5; i++) {
        v.pop_back();
        cout << "size: " << v.size() << endl;
        cout << "capacity: " << v.capacity() << endl;
    }

    //This will seg fault if you don't resize, push_back, assign, or insert
    cout << v[0] << endl;

    return 0;
}
```