CS 161
Intro to CS I

Begin Functions
Odds and Ends...

• Design Critique due!!!
• Exam I – Friday, 10/21
• Demo Assignment 2 this week!!!
void Functions

• Doesn’t return a value
• Still has arguments/parameters

Get into groups of 4-5

• Write a void check_positive_int ()
  – What does the parameter look like?
  – What does the function call look like?
  – Is it more useful to return a value?
```cpp
#include <iostream>
#include <string>  // location of C++ string object
#include <cstdlib> // location of atoi() / ASCII to integer

using namespace std;

// check to make sure you have a good positive int
void check_pos_int(string);

int main() {
    string msgg="hello"; // can assign a string literal to string object
    cout << "Enter a number: ";
    cin >> msgg;

    check_pos_int(msgg);

    // if all the characters are digits in string, then...
    cout << atoi(msgg.c_str()) << endl; // change it to an integer

    return 0;
}

void check_pos_int(string msgg){
    int i; // declare it outside the for because we use if after the for
    for(i=0; i<msgg.length(); i++) {
        // if a character is not within the range of '0' - '9' on ASCII chart,
        // then it isn't a positive integer
        if(!(msgg.at(i) >= '0' && msgg.at(i) <= '9')) {
            cout << "bad input" << endl;
            exit(0); // get out of the whole program
        }
    }
}
```
Scope (Visibility)

• Part of program in which a declaration is valid
• Local variable
  – Declared inside a function only accessible inside function
• Localizing variables
  – Declaring variable in innermost scope
C++ Pass by Value

void swap(int, int);
int main() {
    int a=5, b=10;
    swap(a, b);
    cout << "a: " << a << "b: " << b;
}
void swap(int x, int y) {
    int temp = x;
    x = y;
    y = temp;
}
• What if we didn’t have temp?
More about break, exit, and return

• **break** – used with switch and loops, breaking out of the closest associated case or loop (for, while, or do while). This statement can only occur in a loop or case, otherwise the compiler yells!

• **return** – leave the current function, which exits the program when in the main() function. You can put this anywhere inside any function, otherwise the compiler yells!

• **exit()** – exit the entire program, no matter where this is encountered. You can put this anywhere inside any function, as long as you include `<cstdlib>`, otherwise the compiler yells!
Programming Demo

• Would it be more helpful to return a value from check_positive_int()?
More About Functions

• Do not use global variables!

• Function Headers
  – Description, Parameters, and Return Value
  – Postconditions
    • What is this?
  – Preconditions
    • What is this?
Even though it works, 
DO NOT USE GLOBAL VARIABLES!!!

```cpp
#include <iostream>
using namespace std;

int num, denom;  //Make global variables, not good!

void zero_check() {
    while(denom == 0) {
        cout << "Can't divide by zero!!!" << endl;
        cout << "Enter denominator: ";
        cin >> denom;
    }
}

int main () {
    cout << "Enter numerator: ";
    cin >> num;
    cout << "Enter denominator: ";
    cin >> denom;
    zero_check();  //Call function w/o args
    cout << "Division: " << num/denom << endl;
    return 0;
}
```
```cpp
#include <iostream>

using std::cout;
using std::endl;

int pwr(int, int n=1);  // Example of default args

int main() {
    int base=2, expn=8;
    cout << "The power function: " << pwr(base, expn) << endl;
    cout << "The power function: " << pwr(base) << endl;
    return 0;
}

int pwr(int x, int n) {
    int num=1;
    for(int i=0; i < n; i++) {
        num*=x;
    }
    return num;
}
```

"test.cpp" 25L, 388C written 1,19 All
C++ Function Overloading

• Multiple functions w/ same name
• Arguments determine function
• Default Args can be done w/ overloading
• Example: pow()
```cpp
#include <iostream>

using std::cout;
using std::endl;

//int pwr(int, int n=1); //Example of default args
double pwr(double, double); //Example of overloading

int main() {

    int base=2, expn=8;
    double dbase=2.2, dexp=8.0;

    //cout << "The power function: " << pwr(base, expn) << endl;
    //cout << "The power function: " << pwr(base) << endl;
    cout << "The power function: " << pwr(dbase) << endl;
    //cout << "Pow with int and double: " << pwr(2, 2.2) << endl;
    return 0;
}

double pwr(double x, double n = 1) {
    double num=1.0;
```
Programming Demo...