CS 161
Intro to CS I

Decomposition/Functions
Chap. 3.1 – 3.2
Debugging Errors

• Syntax:
  – READ compiler errors (pay attention to line #)
  – Use google to search for error

• Logic/Runtime
  – Use std::cout to find where the code is breaking
    • Print variable values
    • Print indicator messages
  – Trace through the code
  – Comment out code
Error Handling

• Prevent Divide/Mod by Zero

```cpp
#include <iostream>

using namespace std;

int main () {
    int num, denom;
    cout << "Enter numerator: ";
    cin >> num;
    cout << "Enter denominator: ";
    cin >> denom;

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

    cout << "Division: " << num/denom << endl;
    return 0;
}
```
Decomposition

• Divide Problem (task) Into Subtasks
  – Procedural Decomposition
  – Examples: cooking, cleaning, etc.

• Incremental Programming
  – Iterative Enhancement (Stepwise Refinement)

• Examples: Replicating Code
Procedural Decomposition

• Functions
  – int main() {
  
  – User defined
    void draw_box() {

• Function Call
  – draw_box();
Procedural Decomposition

```cpp
#include <iostream>
using std::cout;

int main() {
    cout << "+-------+\n";
    cout << "|     |\n";
    cout << "+-------+\n";
    cout << "|     |\n";
    cout << "+-------+\n";
    cout << "|     |\n";
    cout << "+-------+\n";
    return 0;
}

void draw_box(); //Declare function

int main() {
    draw_box(); //Use function
    draw_box();
    return 0;
}

void draw_box() { //Define function
    cout << "+-------+\n";
    cout << "|     |\n";
    cout << "+-------+\n";
}
```
#include <iostream>

void draw_box();
void draw_top_bottom();
void draw_sides();
int main() {
    draw_box();
    return 0;
}

void draw_box() {
    draw_top_bottom();
    draw_sides();
    draw_top_bottom();
}

void draw_top_bottom() {
    std::cout << "+--------+\n";
}

void draw_sides() {
    std::cout << "|           |\n";
}
Functions

• What is a function?
  – Block of code to perform action/subroutine

• When have we seen functions already?
  – Predefined

• What is the purpose?
  – Reduce
  – Reuse
  – Readability
Generalization

• Does a function make a task more specific or more general?
  – Justification
  – Examples
Predefined Functions

- `sqrt()`
- `pow()`
- `abs()`
- `rand()`
- `srand()`
- What is the difference b/w `srand()` and others?
void Functions

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

• Can we write a `void check_denominator()`?
• Is it more useful to return a value?
denom not visible in zero_check()
Even though it works, DO NOT USE GLOBAL VARIABLES!!!
Uh-Oh, What happened...

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

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

int main () {
    int num, denom;
    cout << "Enter numerator: ";
    cin >> num;
    cout << "Enter denominator: ";
    cin >> denom;
    zero_check(denom);
    cout << "Division: " << num/denom << endl;
    return 0;
}
```

Reading/Assignments

• Read Chap. 3.2 – 4.1
• Assignment #3 Posted