CS 161
Intro to CS I
Errors, Debugging, and Procedural Decomposition
Odds and Ends...

• Exam I – Friday, 1/29
• In-class review on Wed.
• Demo Assignment #2 today!
• Work on Assignment #3!!!
Programming Errors

• Syntax errors
  – Misuse of C++ language
  – How are they caught?

• Logic errors
  – Doesn’t perform task correctly (aka. bugs)
  – How are they caught?

• Runtime errors
  – Stops your program from running
  – How are they caught?
Syntax Error Examples

- Missing main function
- Use of identifier not declared
- Misspelled Words
- Forget a Semicolon
- Forget Required Keyword
- Missing quote, curly brace, and parenthesis
- Use of single quotes instead of double
Logic Error Examples

• Poorly written programs
  – Add instead of subtract (incorrect operation)
  – Using last two digits for date
  – Same error message for different errors
  – Program that never ends
  – Add one to the largest integer (could be syntax)
Runtime Error Examples

• Open a file that doesn’t exist
• Segmentation fault
  – Infinite loop that eats memory
  – Divide by variable that is zero
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

• What can we do to prevent these errors?
  – Overflow
  – Divide by zero
  – Bad input by the user
Demo...
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 << "|        |
    cout << "+--------+\n";
    cout << "|        |
    cout << "+--------+\n";
    cout << "|        |
    return 0;
}

#include <iostream>
using std::cout;

void draw_box(); //Declare function

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

void draw_box() { //Define function
    cout << "+--------+\n";
    cout << "|        |
    cout << "+--------+\n";
    cout << "|        |
}
```
Functions Calling Other Functions

```cpp
#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 << "|           |
```
```
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?
Programming Demo
Programming Demo