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!!

Hints - 16 char variables space
char grid00 = "grid00"
if (x == 0 and y == 0)
    grid00 = 'X'
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
```cpp
#include <iostream>
#include <cstdlib>
#include <ctime>
#include <string>  // c++ string
using namespace std;
#define GUESSES 3

int main() {  
    int rnum, unum;
    string s;
    //bool error=true;

    // only do this once and once only
    srand(time(NULL));
    rnum=rand()%11;  // in the range 0-10
    cout << rnum << endl;
    for(int x=0; x<GUESSES;x++){
        //error=true;  // we assume they are an idiot first
        cout << "Enter a number 0-10: ";
        cin >> s;
        // if all the characters in the string are 0-9, then good
        do {
            for(int i=0; i<s.length(); i++){
                if(!(s.at(i)=='0' && s.at(i)!='9')){
                    cout << "idiot! enter 0-10: ";
                    cin >> s;
                    i=-1;
                }
            }
        } while (i>=0);
    }
    return 0;
}
```
/if all the characters in the string are 0-9, then good

```cpp
do {
    for(int i=0; i<s.length(); i++) {
        if(!(s.at(i) >= '0' && s.at(i) <= '9')){
            cout << "idiot! enter 0-10: ";
            cin >> s;
            i=-1;
        }
    }
}
unum=atoi(s.c_str()); //when we have a good int, let's change s to int
/*do {
    if(unum>10 || unum<0) {
        cout << "you idiot! enter 0-10! ";
        cin >> unum;
    } else
        error=false;
}while(error==true);*/
if(unum>10) {
    cout << "you idiot! enter 0-10! ";
    cin >> s;
} else
    break; //break works for loops and switch
while(1);
cout << unum << 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

#include <iostream>
using std::cout;

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

Void function does not return info to the call!
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?
```cpp
#include <iostream>
#include <cstdlib>
#include <ctime>
#include <string>  // c++ string
using namespace std;
#define GUESSES 3

bool is_pos_int(string s) {
    for(int i=0; i<s.length(); i++) {
        if(!s.at(i)=='0' && s.at(i)!='9')
            return false;
    }
    return true;
}

int main() {
    int rnum, unum;
    string s;

    // only do this once and once only
    srand(time(NULL));
    rnum=rand()%11;  // in the range 0-10
    cout << rnum << endl;
    for(int x=0; x<GUESSES; x++) {
        cout << "Enter a number 0-10: ";
        cin >> s;
        // if all the characters in the string are 0-9, then good
        while(!is_pos_int(s) || (unum=atoi(s.c_str())) > 10) {
            cout << "you idiot";
            cin >> s;
        }
        cout << unum << endl;
    }
    return 0;
}
```