CS 161
Intro to CS I

Loops/Begin Functions
Odds and Ends...

• Test Friday
• Study Sessions: Wed 6-7:30 and Thur 5-7:30
• Last week to demo Assignment 2!!!

• Read: conditionals (if/else and switch), loops (for, while, do-while), and begin functions
bool bad_input;
//cin ignores all leading whitespace,
//then reads characters of its expected
//type until it gets to a character not
//of its type OR whitespace without
//consuming the char not of its type or whitespace

do {
    bad_input=false;  //assume good input each time, until given otherwise
    //can use to see if a cin failed and then clear the failbit and ignore bad
    /*if(cin.fail()){
        cin.clear();  //clear the failbit
        cin.ignore(256, '\n');  //ignore/disregard our the input
    }*/
    cout << "enter an int: ";
    cin >> s;  //what happens when you enter a non-integer or newline/enter?
    cout << "value of x: " << s << endl;
    for(int i=0; i<s.size(); i++){}  //for all the chars in s
        if(!(s.at(i)='0' && s.at(i)='9'))  //check each to see if 0-9 digit
            bad_input=true;
}
while(bad_input);

x=atoi(s.c_str());  //now you know the string is good, so change ascii to int
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 << "+--------+
    cout << "|           |
    cout << "+--------+
    cout << "+--------+
    cout << "|           |
    cout << "+--------+
    return 0;
}

void draw_box() {  //Define function
    cout << "+--------+
    cout << "|           |
    cout << "+--------+
    cout << "+--------+
    cout << "|           |
    cout << "+--------+
```

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

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

void draw_box() {  //Define function
    cout << "+--------+
    cout << "|           |
    cout << "+--------+
    cout << "+--------+
    cout << "|           |
    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 << "+--------+
```

```cpp
    std::cout << "|           |
```

```cpp
    std::cout << "|           |
```

```cpp
    std::cout << "|           |
```

```cpp
    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

3

software engineering & large scale projects
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?

x = rand();

\[ \text{gave you back info} \]

srand(time(...));

\[ \text{gave it an argument \& didn't give back info} \]
void Functions

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

• Can we write a **void good_pos_int()**?
• Is it more useful to return a value?
```cpp
#include <iostream>
#include <string> // c++ strings
#include <cstdlib> // contains atoi()

using namespace std;

// return true if it is bad and false if it is good int
bool check_positive_int(string s) {
    for(int i=0; i<s.size(); i++) { // for all the chars in s
        if (!(s.at(i) >= '0' && s.at(i) <= '9')) // check each to see if 0-9 digit
            return true;
    }
    return false;
}

int main() {
    int x = 20;
    string s; // c++ style string
    float f;
    char c;
    bool bad_input = false;

    // cin ignores all leading whitespace,
    // then reads characters of its expected
    // type until it gets to a character not
    // of its type OR whitespace without
    // consuming the char not of its type or whitespace
    do {
        cout << "enter an int: ";
        cin >> s; // what happens when you enter a non-integer or newline/enter?
        cout << "value of x: " << s << endl;
    } while (check_positive_int(s));

    x = atoi(s.c_str()); // now you know the string is good, so change ascii to int
    cout << "enter a float: ";
```