Odds and Ends...

• Exam I – Friday, 1/29
• Exercise #4 posted
• Keep working on Assignment #3!!!

• Monday lab people don’t panic
• Don’t miss Demo, and be patient waiting
• READ, READ, READ!!!!
• Ask TA pointed questions
• Just THINK! KISS😊
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
#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, num;
    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) || (num=atoi(s.c_str())) > 10)){
            cout << "you idiot";
            cin >> s;
        }
        cout << num << endl;
    }
    return 0;
}
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
Illegal access outside loops

```cpp
for(x = 0; x < 10; x++) {
    int y = 10;
    cout << "The value of x * y is: " << x*y << endl;
}
cout << "The value of y is: " << y << endl; /*y outside scope*/
```

- How do we fix this?
- What about if/else blocks?
Illegal access in functions

```c
int main () {
    int x=2, y=3;
    compute_sum();
    sum = x+y;  //error: sum hasn’t been declared
    return 0;
}
void compute_sum() {
    int sum = x+y;  //error: x and y outside scope
}
```
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!