Static Variables
Function Scope

- “Normal” local variables go out of scope and are deallocated when a function terminates.
- Static variables remain and retain their value.

```c
void f( )
{
    static <variable_type> <variable_name>
    etc
}
```
Rules for Static Variables

1. A static variable declaration is **only executed once**, the first time the function is executed.

2. A static variable is **initialized only once** (since this is part of the declaration process) and will be initialized to 0 unless the programmer designates otherwise.

3. Subsequent invocations of the function in which a static variable resides will retain the last value of that variable.
void my_function() {
    static short count = 1;
    cout << "this function has been called "
        << count << (count == 1 ? "time" : "times");
    count++;
    return;
}

int main() {
    for (int i = 0; i < 5; i++)
        my_function();
    return 0;
}
void is_it_larger (const float val)
{
    static float largest = val;
    if (largest < val)               // new val is larger
        largest = val;
    cout << "The largest value sent to this function "
         << "so far is " << largest << endl;
    return;
}

int main()
{
    for (int i = 0; i < 7; i++)
        is_it_larger(i);
    return 0;
}
#include <ctime>       // access to time clock

long my_rand()
{
    static long seed = time(NULL);
    seed = (104729 + seed * 7919) % 15485863;
    return abs(seed);
}

int main()
{
    for (int i = 5; i > 0; i--)
        cout << my_rand() << endl;
    return 0;
}
End of Session