

SI 413: Debugging: Being the detective in a murder mystery where you are also the murderer.

Professor Keith Sullivan

Control Flow

- ▶ Determines order of execution
- ▶ Eight primary categories:
 1. Sequencing
 2. Selection
 3. Iteration
 4. Procedural Abstractions
 5. Recursion
 6. Concurrency
 7. Exception Handling
 8. Non-determinacy

Pointers and Arrays

```
int n;  
int a*;  
int b[10];
```

```
a = b;  
n = a[3];  
n = *(a+3);
```

[] is syntactic sugar for pointer arithmetic. In general, $E1[E2] = *((E1)+(E2))$. Which implies $A[3] = \dots$?

Can do other pointer arithmetic operations: comparison, subtraction

Pointers and arrays are not the same. In particular, declaration. Why?

Stack Smashing in C

```
int get_acct_num(FILE *s)
{
    char buf[100];
    char *p = buf;
    do {
        *p = getc(s);
    } while (*p++ != '\n');
    *p = '\0';
    return atoi(buf);
}
```

Assignment

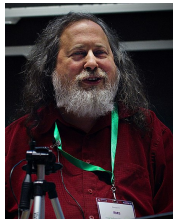
- ▶ l-values, r-values
- ▶ Value model: variable is a named **container** for a variable
- ▶ Reference model: variable is a named **reference to** a value

Open Source Software

- ▶ Intellectual property laws prevent the modification and sharing of creative work
 - ▶ Treats creative output as property, comparable to private property
- ▶ Free and open source software (FOSS) uses IP laws for the inverse purpose: sharing and collaboration

History

- ▶ 1980: US changes IP laws to classify computer code as literary work
 - ▶ Impetus for the GNU Project by Richard Stallman
- ▶ 1985: Free Software Foundation (FSF) is founded
 - ▶ First open source licenses
- ▶ 1998: Open Source Initiative (OSI) is founded
 - ▶ "Given enough eyeballs, all bugs are shallow"
 - ▶ Brought open-source to corporate developers



FSF FREE SOFTWARE
FOUNDATION



Open Source Licenses

- ▶ Permissive
 - ▶ Academic license
 - ▶ Use, modify, and redistribute software
 - ▶ No obligation to distribute source code
 - ▶ Can be used in proprietary projects
 - ▶ Only stipulation is to cite authors
 - ▶ BSD, MIT License, Apache License
- ▶ Copyleft
 - ▶ Require derivative works to distribute source code
 - ▶ Prevents proprietary software from not contributing to the broader community
 - ▶ Attracted commercial developers
 - ▶ GNU General Public License (GPL), Lesser General Public License (LGPL), Mozilla Public License (MPL), Eclipse Public License (EPL)

Beerware

```
/*  
* _____  
* "THE BEER-WARE LICENSE" (Revision 42):  
* <phk@FreeBSD.ORG> wrote this file. As long as  
* you retain this notice you can do whatever you  
* want with this stuff. If we meet some day, and  
* you think this stuff is worth it, you can buy  
* me a beer in return. Poul-Henning Kamp  
* _____  
*/
```

Unstructured Programming

- ▶ Sequencing (how to do in Scheme?)
- ▶ Program counter
- ▶ Goto statements (branching)
 - ▶ In Fortran 90 and C++ for backwards compatibility
 - ▶ Most other languages do not have them

Structured Programming

- ▶ Abandoning goto → revolution known as structured programming
 - ▶ Hot trend in the 1970s
- ▶ Emphasizes top down design, modularization, structured types, descriptive naming, extensive commenting
- ▶ Algol 60 pioneers familiar control-flow constructs
- ▶ Replace goto with return, break, continue, exceptions
- ▶ Assembly still uses branches, so structured code speed does not suffer

Errors

- ▶ Non-local goto: return from a surrounding routine
 - ▶ Deallocation of appropriate stack frames
 - ▶ Register restoration
- ▶ Deeply nested code lacks context for meaningful error messages
- ▶ Most straightforward handling of exceptions: auxiliary Boolean variables

```
status = my_prog(args)  
if status == OK then ...
```

- ▶ Better answer: **exception handlers**

Iterators

- ▶ How to have a single function loop over all members of a data structure?
- ▶ Iterators: data structure that allows moving over a collection
 - ▶ Get next element
 - ▶ Know when at the end of the collection
- ▶ C++: abstraction of the pointer type (pointer arithmetic)
- ▶ Java: Iterator interface
- ▶ Python: Iterator type
- ▶ For-each loops

Generics

- ▶ Same chunk of code, multiple data types
- ▶ What if no type checking?
- ▶ C++:

```
template<class T>
T min(T a, T b) {
    if (a < b) return a;
    else return b;
}
```

- ▶ Java:

```
static <T extends Comparable<T>> T min(T a, T b)
    if (a.compareTo(b) < 0) return a;
    else return b;
}
```