CS 537
Section 1
Programming in Unix and C

Michael Swift

Project notes

- First project is individual
  - The rest are for groups
    - How large a group?
    - How should we assign credit?
- Project blog for
  - posting questions / answers
    - questions to instruct537-2@cs.wisc.edu
  - Sharing test code
    - Do **not** share project code
  - It **is moderated**

Facilities

- Department linux machines:
  - 1350: emperor
  - 1366: royal
  - 1358: collaborative programming (8 machines with large monitors)
- Unix Orientation classes in room 1325
  - Today at 4 pm
  - Tomorrow at 4 pm
  - Monday at 4 pm

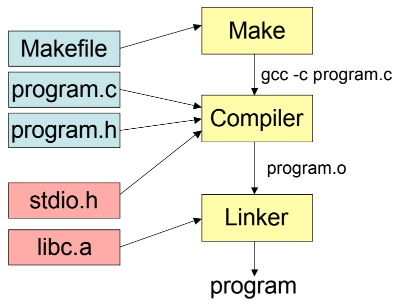
Why C

- All modern operating systems are written in C
- Why?
  - Control
  - Predictable code
  - Expressive
  - Optimizable
  - Powerful pre-processor

Issues with C

- Little hand-holding for programmer
  - Manual memory management
  - Small standard library
  - No native support for threads and concurrency
  - Weak type checking

Using C and Unix



C language

```
#include <stdio.h>
int main(int argc, char * argv[])
{
  printf(“Hello, world\n”);
  return(0);
}
```

Issues with C

- Memory allocation
  - malloc(), free()
- Pointer arithmetic and arrays
- Preprocessor

Example

More advanced topics

- Compiler errors and warnings
  - gcc -Wall foo.c
- Optimization for faster and smaller code
  - gcc -O foo.c
  - gcc -O2 foo.c
- Separate compilation
  - gcc -c foo.c
  - gcc -c bar.c
  - gcc -o foobar foo.o bar.o

Makefiles

- Specify the commands to compile code
  - in a file named “Makefile”
- Example:

```
foo.o: foo.c
        gcc -c -O -Wall foo.c
bar.o: bar.c
        gcc -c -O -Wall bar.c
foobar: foo.o bar.o
        gcc -o foobar foo.o bar.o
default: foobar
```

- General format:

```
target: prereq1 prereq2
<tab>   command1
<tab>   command2
```

Documentation

- Unix/Linux man pages
  - example: “man close”

```
CLOSE(3)                BSD Library Functions Manual                FCLOSE(3)

NAME
     fclose -- close a stream

LIBRARY
     Standard C Library (libc, -lc)

SYNOPSIS
     #include <stdio.h>

     int
     fclose(FILE *stream);

DESCRIPTION
     The fclose() function dissociates the named stream from its underlying
     …

RETURN VALUES
     Upon successful completion 0 is returned.  Otherwise, EOF is returned and
     …
```

Man pages

- Documentation is divided into sections
  1. Programs, commands
  2. System calls
  3. Subroutine libraries
  4. Hardware
  5. Config files
  6. Games
  7. Miscellaneous
  8. System administration
- man returns the result from the lowest-numbered section
- apropos searches for commands with a word

Debugging

- Compile with debugging using "-g"
  - gcc -g -o foo.o foo.c
- Run your program with gdb

```
gdb foobar
GNU gdb 6.3
<copyright omitted>
(gdb) break main
breakpoint 1 at 0x80483b0: in file foo.c, line 5
(gdb) run
Starting program: /afs/cs.wisc.edu/…/foobar
Breakpoint 1, main (argc=1, argv=0xbfe27804) at foo.c:5
5      if (argc > 1) {
(gdb) print argc
$1 = 1
(gdb)
```