CSE 351
Section 1: Intro to C
Housekeeping

• TAs
  – Nick Burgan-Illig
    • nmbi@uw.edu
    • Office Hours: Th 11:30-12:30, CSE 216
  – Cortney Corbin
    • ccorbin@cs.washington.edu
    • Office Hours: TBD
  – Chee Wei Tang
    • acetang@cs.washington.edu
    • Office Hours: Tu 3:30-4:30, CSE 218

• Sections once a week
  – Supplement class material
  – Ask questions on homework/labs

• Other avenues for help
  – Discussion boards, direct email, office hours
  – With 3 TAs, shouldn’t be difficult to find help
Introductions

• Who am I?
• Who are you?
Today

• Overview of C
  – Mainly discuss a few differences from Java
  – Not a real tutorial, not enough time
  – See the C book for a good introduction

• Overview of debugging C programs

• Introduction to pointers in C
Intro to C: Why C?

• It’s ubiquitous
  – 2\textsuperscript{nd} most popular language today
• Modern languages are still implemented in C
  – Java, Python, Perl, PHP, Ruby
• So are operating systems
• Affords great performance and more control
  – “With great freedom comes great responsibility”
/* HelloWorld.java */

class HelloWorld {
    public static void main(String[] args) {
        System.out.println("Hello, "+"world!");
    }
}

Intro to C: Hello World in Java
Intro to C: Hello World

/* hello.c */
#include <stdio.h>

int main(int argc, char *argv[])
{
    printf("Hello, world!\n");
    return 0;
}

Intro to C: Hello World

```c
/* hello.c */
#include <stdio.h>

int main(int argc, char *argv[])
{
    printf("Hello, world!
");
    return 0;
}
```

Preamble of file includes headers, provides function declarations, useful comments, etc.

Common headers, see refs:
- stdio.h
- stdlib.h
- stdint.h
- unistd.h
- string.h
Intro to C: Hello World

/* hello.c */
#include <stdio.h>

int main(int argc, char *argv[])
{
    printf("Hello, world!\n");
    return 0;
}
Intro to C: Compiling

- Previous program in hello.c
- To compile and run:
  
  ```bash
  $ gcc hello.c -o hello -Wall
  $ ./hello
  Hello, world!
  ```

- Options:
  
  - `o` What to name the output file
  - `Wall` Print all warnings
Intro to C: C and Java

• C is a weakly typed language
  – int, float, long int, double, etc.

• Syntax similar to Java
  – if/then/else, do/while, for, switch/case

• printf/scanf for console I/O

• open/read/write/close for file I/O
Intro to C: Differences from Java

• No classes! No objects!
  – Class-like things though; check out structs
  – Data only, no methods

• No garbage collection! Not managed!
  – Must remember to allocate/deallocate on your own
  – No built-in bounds checking

• No exceptions!
  – Need to do your own error checking/handling

• No virtual machine!
  – Must recompile the code for different architectures
  – Compiles to “real” op codes (as opposed to virtual)
Intro to C: References

• The C Programming Language
  – Written by the authors of the language
  – Concise and precise
  – Excellent collection of practice problems

• Linux `man` pages
  – Useful for looking up how to use a particular function, e.g.:
    
    ```bash
    $ man printf
    ```
Intro to C: Debugging

• You write a program, try to run it, and it crashes. What now?
Intro to C: Debugging

• One option: “printf debugging”
  – Add printf statements to the code to see where/why it crashes

• Another idea: run it through a debugger
  – Monitor accesses to variables, see where the program crashes, verify loop invariants, etc.

• Depends on the situation; one may be easier than the other
Intro to C: `printf` Debugging

- `printf` allows you to print formatted strings
- Arguments include a *format string*, and data to display
- Format string is a literal string, containing special placeholders indicating how to display the data
- Ex:
  - `printf("Sum: %d + %d = %d\n", 1, 2, 1+2)`
    - `%d` displays an integer
    - Produces “Sum: 1 + 2 = 3”
- Seen “`man printf`” or the C book for more
/* Buggy program */
#include <stdio.h>

int main(int argc, char* argv[]){
    int a = 5, *b = &a;
    printf("%d %d\n", a, *b);
    a ^= a; b = *b ^ a;
    printf("%d %d\n", a, *b);
    return 0;
}
Intro to C: Debugging with GDB

• Use `–ggdb` to compile with debugging symbols
  
  
  ```
  $ gcc –o foo –Wall –ggdb foo.c
  ```

• Invoke with `gdb`:
  
  ```
  $ gdb ./foo
  ```

• Important commands:
  
  – run
  – break <line# / symbol>
  – step
  – continue
  – info <locals / frame / register>
  – print, x
  – backtrace
  – help
Intro to C: Debugging with GDB

/* Buggy program */
#include <stdio.h>

int main(int argc, char* argv[]){
    int a = 5, *b = &a;
    printf("%d %d\n", a, *b);
    a ^= a; b = *b ^ a;
    printf("%d %d\n", a, *b);
    return 0;
}
Intro to C: Taste of Pointers

• Variables in C have types
  – int, long, double, float, char, etc.

• A pointer is just another type
  – Pointers store addresses of other variables
  – int is an integer, but int* is a pointer to an int
  – Same for float and float*, char and char*, etc.

• “NULL pointers” are pointers containing 0 (zero)
Intro to C: Taste of Pointers

• & is the address-of operator
  – Returns the address of a variable
• * is the value-of operator
  – Retrieves the value stored at the address in a pointer
  – “Dereferencing”
  – NULL pointers cannot be dereferenced

• Ex:

  ```c
  int a = 5; int *ap;
  ap = &a; *ap = 10;
  printf("%d %d\n", a, *ap);
  ```
Intro to C: Debugging with GDB

/* Buggy program */
#include <stdio.h>

int main(int argc, char* argv[]){
    int a = 5, *b = &a;
    printf("%d %d\n", a, *b);
    a ^= a; b = *b ^ a;
    printf("%d %d\n", a, *b);
    return 0;
}
Intro to C: Debugging with GDB

/* Buggy program */
#include <stdio.h>

int main(int argc, char* argv[]){
    int a = 5, *b = &a;
    printf(“%d %d\n”, a, *b);
    a ^= a; b = *b ^ a;
    printf(“%d %d\n”, a, *b);
    return 0;
}

b becomes NULL, so dereferencing causes a crash
Intro to C: Taste of Pointers

• Why are pointers useful?
Intro to C: Taste of Pointers

• Why are pointers useful?
• Some ideas:
  – Linked data structures
  – Passing by reference
  – Avoid copying large blocks of data
  – Any others?
• Don’t need to know this stuff now; just wanted to whet your appetite!
HW0

• Has anyone started yet?
• Any questions?