Administrivia

- Reading Assignments on Web
- HW3 Announced on Wed.

Tip of the Day

- Try stuff out!
  - Run lecture examples
  - Tweak them to try new things
  - Make example programs
    - For things you don’t understand
  - Try “interesting” exercises in book
    - If you don’t know where to begin

Remember, you’re learning to teach yourself

Last Time

- C Memory Model
- Simple C Programs
- Introduction to Pointers

Today

- C-Program Structure
  - Declarations and Scope
  - Argument Passing
- Syntax
  - Booleans and arrays
  - Confusing Cases
- Printf

Storage Duration and Scope

- Define or declare everything before using
- Scope
  - Global variables: outside function
  - Local variables: inside function
- Storage class (lifetime)
  - Globals exist for duration of program
  - Locals exist while their block is active
  - Static locals retain value between invocations
- Examples in declare.c
Structure of a C Program

// First include all header files (more later)
#include <stdio.h>

// Declare global variables (avoid if you can!)
int myVar_g;

// Function must be declared before it is used
void my_function(int a, int b);

int main() {
    my_function(2,3);
    return 0;
}

void my_function(int a, int b) { ... }

Argument Passing

• All Arguments Passed by Value
  – Function receives copy of argument
  – Chances to copy won’t affect original

• You can still modify an argument
  – Using pointers (see pointerArg.c)
  – References introduced by C++

• Examples in pointerArg.c

Syntax: Data Types

• Variables
  – Same as Java (int, float, double, etc.)
  – bool is a recent addition to C
    • Requires #include <stdbool.h>
    • Examples in bool.c

• Arrays
  – Size: constant expression only
  – Funny rules for use (more later)
  – Examples in arrays.c

Syntax: Control Constructs

• Very similar to Java
  – while, if, for, switch
  – break, continue

• Some ugly additions
  – goto : jump to any location
  – Hard to reason about, but handy sometimes
  – More general than Java’s labeled break

Confusion: Left vs. Right Exprs.

• In Assignment (left = right;)
  – Left is evaluated to location (address)
  – Right is evaluated to value
  – Values include numbers and pointers

• Key difference
  – Left: variable gives location
  – Right: evaluated from data in variables
  – Note: same as Java

Left vs. Right

Examples

int X =3;
int *pX;
int Y;
int *pY;
pX = &X;
pY = &Y;
*pY = *pX;

X
pX
Y
pY

0x1000
0x1004
0x1008
0x100b

3
XXXX
XXXX
XXXX
Left vs. Right

Examples

```
int X = 3;
int *pX;
int Y;
int *pY;
pX = &X;
pY = pX;
pY = &Y;
*pY = *pX;
```

Left vs. Right

Examples

```
int X = 3;
int *pX;
int Y;
int *pY;
pX = &X;
pY = pX;
pY = &Y;
*pY = *pX;
```

NULL Pointers

- Pointer to NULL means "nothing"
- NULL defined in `<stdio.h>`
  - Actual value is zero
- Dereferencing causes crash
- Examples in `pointer.c`

Confusion: Dangling Pointers

- Pointer initialized to address
- Storage for that address is reclaimed
  - Lifetime of variable ended
  - Explicitly de-allocated (we'll see this later)
- The pointer is left "dangling"
  - Points to undefined location
  - Will crash if you're lucky!
  - Usually subtle and silent bugs.
- Examples in `dangling.c`
Confusion: C99 Standard

- Revisions to C language in 1999
- Not fully supported in our gcc
  - Some require additional flag `-std=c99`
  - E.g. declaring variables in for (see arrays.c)
- Some just not supported
  - E.g. variable-length arrays
- Know the capabilities of your compiler
- Programming in C assumes C99!

Confusion: Other

- Variable declarations are funky
  - Can have multiple on one line
    - `int x, y; or int x=0, y; or int x, y=0;`
  - Pointers have confusing syntax
    - `int *x, y; or *y` is type `int` yet `y` is type `int`
    - `not int *y;`
- Implicit declarations
  - Function assumed to return int if not declared
  - Can result in "linker error"

Printf

- `printf("format", v1, v2, ...);`
- Most Common Formats
  - `%d` : int
  - `%f` : float, double
  - `%c` : char
  - `%s` : char* (strings)
  - `%x` : hexadecimal
- Examples in `format.c`

Summary

- C-Program Structure
  - Declarations and Scope
  - Argument Passing
- Syntax
  - Booleans and arrays
  - Confusing Cases
- `Printf`

Reading

- Programming in C
  - Skim Chapters 4-6
    - Only if you need to refresh your memory of Java
  - Chapter 8: Functions
  - Chapter 11:
    - pp235-240: Pointers in expressions
    - pp254-259: Pointers and Functions
    - pp272: Operations on Pointers
    - pp274-276: Pointers and Memory Addresses

Next Time

- Arrays
- Strings
- Command Line Arguments