

Slide 1

Administrivia

- Homework 7 originally due today. Extend deadline to Wednesday.

Slide 2

Minute Essay From Last Lecture

- Many people use graphing calculators. (I was surprised that people still find a separate gadget useful!)
- Also many mentions of programs used to “make plots” (previous question).

Pointers Revisited

Slide 3

- Every time you call `scanf`, you pass it at least one parameter of the form `&x`. What does that mean? Also, when you look at `man` pages for some functions, they show function declarations with parameters of the form `type *`. What does that mean?
- To explain, we need one more kind of variable — *pointers*. A pointer, as its name suggests, points to something — namely, a location in memory. Typically a pointer “points to” a variable.

Pointers in C

Slide 4

- Many programming languages provide something like pointers. Unlike some more-recent languages, C allows you to have both pointer variables and non-pointer variables.
- To a first approximation, C pointers are just memory addresses — i.e., numbers — but they are declared to point to variables (or data) of a particular type. Examples:

```
int * pointer_to_int;
double * pointer_to_double;
```
- Can display value of pointer using `printf` with `%p`. Sometimes interesting in exploring how variables are laid out in memory (implementation-dependent).

Pointers in C — Operators

Slide 5

- `&` gets a pointer to something in memory. So for example you could write

```
int x;  
int * x_ptr = &x;
```

- `*` “dereferences” a pointer. So for example you could change `x` above by writing

```
*x_ptr = 10;
```

- Special value `NULL` means the pointer “doesn’t point to anything”. Dereferencing a null pointer usually produces an error, as does dereferencing an uninitialized pointer variable.

Pass By Reference, Sort Of — Review(?)

Slide 6

- Functions can only explicitly return a single value — a significant limitation. Pointers provide a way to get around that: By passing a pointer to something, rather than the thing itself, can in effect have a function return multiple things.
- To make this work, declare the function’s parameters as pointers, and pass addresses of variables rather than variables. (This is how `scanf` does what it does, and why you need the `&`.)
- (The “sort of” in the slide title is because this is not true pass by reference as in, e.g., C++, but the effect is the same.)
- (We did an example of this a while back — sample program `divide.c`.)

Pointers and Arrays in C

Slide 7

- C treats pointers and arrays as interchangeable in most respects. (This is why it works that many functions whose parameters are supposed to be strings — arrays of characters — declare them as pointers. Look again at `man` page for `printf`, e.g.)
- About the only difference is behavior of `sizeof` operator — for locally-declared array you get size *in bytes*, for array parameter or pointer you get pointer size.

Pointer Arithmetic in C

Slide 8

- C also permits doing some arithmetic operations on pointers, though only the ones that are “sensible”.
- Adding an integer n to a pointer that points to *type* advances it n times the size of *type*. Subtracting an integer from a pointer works similarly. (Strictly speaking, though, you should only do this within an array.)
- Subtracting one pointer from another gives an integer result. (This can be particularly useful in working with strings.)
- Comparing pointers with relational operators works, though strictly speaking you should probably only use less-than and greater-than operators on pointers into the same array.

Pointer Arithmetic in C, Continued

- Example: If `a` is an array of `ints`, `a[2]` and `*(a+2)` are equivalent.
- So we could write loops over arrays using pointers. Once upon a time that was sometimes more efficient. With current compilers, probably not so, so use whatever is most readable.
- (Example.)

Slide 9

Character Strings in C — Preview

- We'll talk more about text data soon, but for now a summary version:
- Text strings are represented as arrays of characters. Can vary in length; end of string indicated by a special character.
- Text in double quotes (e.g., in a call to `printf`) defines a string constant — so somewhere in memory there is an array of those characters.
- (So now the arguments to `fopen`, etc., make more sense?)

Slide 10

Minute Essay

- None — quiz.

Slide 11