Administrative Stuff

- Quiz 1 and solution set posted
- Assignment 2 due
- Assignment 3 posted soon
  - non-programming
  - either hard or electronic copies are fine
Last Time

- pointers
  - more on pointers
  - little bit of pointer arithmetic
  - pointers and arrays
  - passing pointers to functions
- more on bitwise operators
- big endian vs little endian
Fake Quiz

Suppose we have

```c
int x=0xABCD0123;
```
Fake Quiz

Suppose we have

\[
\text{int } x=0\text{ABCD0123};
\]

- \(x\) is stored starting at address 500
- We’re on a little-endian machine
- ints are 4-bytes

How is \(x\) stored in memory?
Fake Quiz

Suppose we have

```
int x=0xABCD0123;
```

- What would we write to declare a pointer that can point to \( x \)?
- How do we make it *point to* \( x \)?
- What would be stored in this pointer after we’ve written this?
- How can we print the value of \( x \) through the pointer?
Suppose we have

```
int x=0xABCD0123;
```

- What would we write to declare a pointer that can point to `x`?
- How do we make it *point to* `x`?
- What would be stored in this pointer after we’ve written this?
- How can we print the value of `x` through the pointer?
- What would be stored in the pointer variable if we added 1 to it (*i.e.*, did a `p++`)?
Fake Quiz

Suppose we have

```c
int x = 0x12345678;
```

- int variables store ints
- float variables store floats
- What do pointer variables store?
Suppose we have

```c
int x=0xABCD0123;
```

- `int` variables store `ints`
- `float` variables store `floats`
- What do pointer variables store? **Answer:** Addresses
Is this ok?

```c
int x=0xABCD0123;

char *p;
p=&x;
```
Legal? What does it do?

```c
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char **argv) {
    int x = 0xABCD0123;
    char *p;
    int i;

    p = (char *)&x;
    for (i = 0; i < sizeof(int); i++, p++)
        printf("%p: %x\n", p, *p);

    return EXIT_SUCCESS;
}
```
Another version of the same

... but don’t worry about the differences from the previous version.

```c
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char **argv) {
    int x = 0xABCD0123;
    unsigned char *p;
    int i;

    p=(unsigned char*)&x;
    for (i=0; i<sizeof(int); i++, p++)
        printf("%p: %.2x\n", p, *p);

    return EXIT_SUCCESS;
}
```