CSC209: Software Tools and Systems Programming

Week 3: Arrays and Pointers pt. 1[1]

Kianoosh Abbasi

[1] Slides are mostly taken from Andi Bergen's in summer 2021.

PCRS: Arrays

Declaring arrays:

```
int student_ids[400];
```

Writing to or reading from array elements:

```
student_ids[0] = 1001111111;
student_ids[399] = 1002222222;
int x = student_ids[0];
```

Questions about these?

PCRS: Pointers and the & Operator

A pointer is a variable that contains the memory address of another variable

1. Assume `x` is an integer and `px` is a pointer
2. Then, `px = &x` stores the *address* of `x` in `px`

The `&` operator expects its operand to be a variable or array element, so constructs such as `&(x+1)` are illegal

PCRS: Pass-by-value and Pass-by-reference

```
int x = 10;
increment_int(x); // Cannot change x
increment2_int(&x); // Can change x
```

`increment()` takes an *integer* parameter, whereas `increment2()` takes an *address of an integer*

PCRS: Pointers and the * Operator

*The * operator interprets its operand as an address, and fetches the memory contents at that address*

1. Assume that `y` is an integer and `px` is a pointer
2. The statement `y = *px` assigns to `y` the integer that `px` points to

The * operator is said to *dereference* its operand

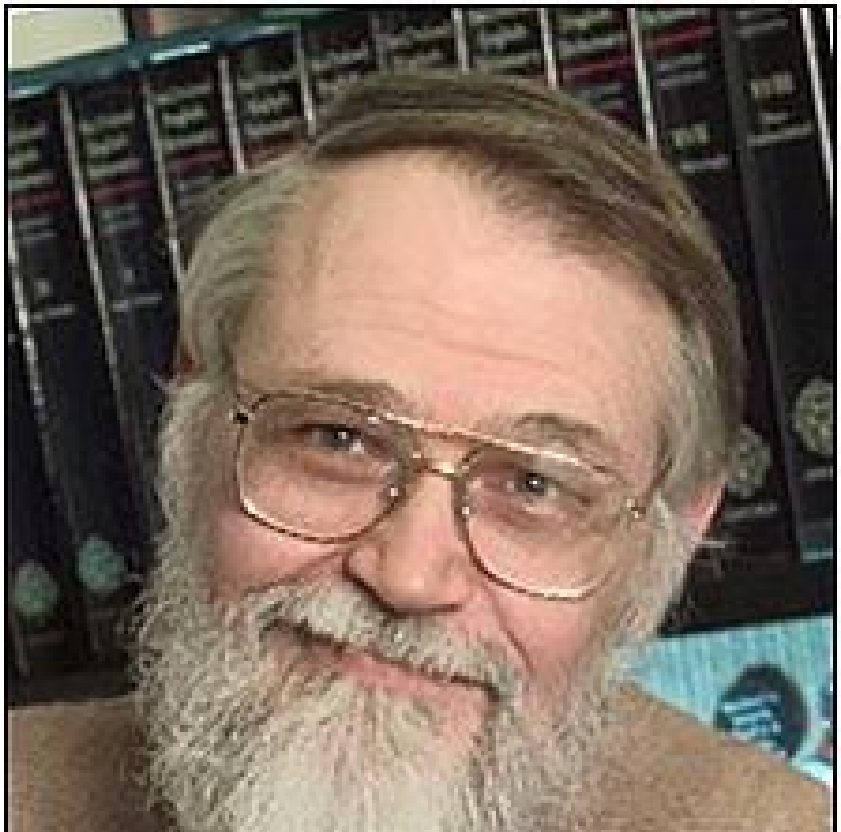
PCRS: Declaring Pointers

Pointer declarations are intended as a mnemonic, so:

```
int *px;
```

means that `*px` is equivalent to a variable of type `int`. Thus, `px` is a pointer containing the address of an `int`.

Why? Because K&R.(See p. 90)



Notes on Pointers and Addresses

- A pointer is not an array
  - But it can contain the address of an array
- An array is not a pointer
  - But the compiler interprets the name of an array as the address of its zeroth element, so the following statements are equivalent

```
int *x = &a[0];
int *y = a; // Assuming "a" is an array.
```

A Common Error With Pointers

What does this do:

```
int *x;
*x = 10;
```

What about this:

```
int *x;
printf("%d", *x);
```

Never dereference uninitialized (or `NULL`) pointers!