Week 02-2

CSC209 Fall 2023

Dr. Demetres (dee) Kostas

September 21, 2023

Announcements

- Lab 2 tomorrow
  - I said on Tuesday
    - there would be only three rooms
  - I was mistaken
    - there **should be** a TA
    - in *all* of the rooms for this week
      - we will adjust a bit next week
- A1 next Wednesday

Arrays and pointers

Arrays

- sequence of multiple variables
  - of the same type
- can also be understood
  - as a single sequence variable
- the array is in sequence
  - in the actual memory
- Indexed using square brackets

Pointers

- a type that stores memory addresses
- when declaring them
  - the type is followed by a *
    - `int *x;`
- the star is also used `*x = 10`
  - to look up the value
    - stored in the address
      - a.k.a. value
    - of the pointer

Getting the address of variables

- any variable
  - including pointers
- has a (starting) memory address
  - containing the variables value
- putting an `&`
  - e.g. x = &y;
    - gives you the address
      - where the value of y is **stored**

Arrays vs. pointers

- they are fundamentally connected
- since the following is true
  - (for any array)

```
// generic array, could be int a[10];
type a[SIZE];
a == &(a[0])
```

Indexing

- with arrays
  - we get the n'th index
  - by saying `a[n]`
    - this also means
    - the value of n'th memory address
- we can also index pointers
  - the following holds true!

```
int a[10];
int *p = &(a[2]);
p[1] == a[3];
```

Pointer Arithmetic

- a very powerful tool
- why do we give the type
  - for pointers, rather than some
    - specifically pointer type?
- because when we do math
  - using the pointer values
    - (using the address stored)
- the actual addresses change
  - appropriate to the type

Example

```
int a[10];
int *p = &(a[0]);
// What is the result of?
p = p + 2;
```

Example - answer

- suppose the value of &(a[0])
  - is 0x200
- what happens with `p+2`?
  - well, it moves up two spaces
    - the size of integers!
- If the sizeof an int is 4
  - then `p + 2`
    - = 0x200 + 2(4) = 0x208
- **This is equivalent to `&(a[2])`!!!**

`calls_and_pointers.pdf`

Some A1 notes

Opening an image file

- this was left ambiguous
  - the handout mentions
    - a "valid file pointer"
- look into the function `fopen`
  - the following code might help

```
FILE *fp = fopen(argv[1], "r");
```

`fscanf.c`