1 Administrivia

Announcements

Assignment

Read 1.5–9.

From Last Time

I/O in C.

Outline

1. I/O exercise.

2. Structures and pointers in C.

Coming Up

Memory allocation in C.
2 Exercise

Complete the exercise from last time. You did sketch out the code, didn’t you?

3 Structures

1. Public classes without methods.

2. General structure:

   ```
   struct <struct_identifier>
   {
     <member_declaration>
     [<member_declaration> ...]
   };
   /* Don’t forget the semicolon!!! */
   ```

3. Example:

   ```
   #include <stdio.h>
   /* "struct dimension" becomes a new type. */
   
   struct dimension
   {
     double length;
     double width;
     double height;
   };
   
   /* Prototypes */
   
   void printDimension(struct dimension);
   
   int main()
   {
     struct dimension box1 = { 1.0, 1.0, 1.0 };
   }
   ```
struct dimension box2;

box2.length = 2.0;
box2.width = 4.0;
box2.height = 6.0;

printDimension(box2);

return 0;
}

void printDimension(struct dimension dim)
{
    printf("Length: %g\nWidth: %g\nHeight: %g\n", dim.length,
            dim.width, dim.height);
}

4 Pointers

1. Pointer variables hold the address of another variable.

2. Examples similar to what we’ve already seen:

double data[10];
double *p_data;
int sum;
int *p_sum;

p_data = data;
p_data[3] = 0.0;

p_sum = &sum;
sum = 10;
printf("Sum: %d\n", p_sum);

*p_sum = 12;  /* Dereference the pointer */;

3. What’s going on here?
4. Pointer arithmetic

(a) You can never add two pointers, but you can add a pointer and an integer:

```c
double sum;
double data[10];
double *dp;
int i;

sum = 0.0;
for (i = 0, dp = data; i < 10; i++, dp++)
    sum += *dp;
```

Note that `dp` will be incremented by `sizeof(double)`.

(b) `data[4]` is another way of writing `*(data + 4)`.

c) You can subtract two pointers:

```c
int strlen(char *s)
{
    char *ptr = s;

    while (*ptr != '\0')
        ptr++;

    return ptr - s;
}
```

5. Exercise: Plain vanilla C arrays always start at an index of 0. Using what we just
learned, how could you use an array and a pointer variable to create an array which began at a negative index?