CSC209: Software Tools and Systems Programming

Week 6: Structs and File I/O [1]

Kianoosh Abbasi

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

PCRS: Structures Recap

A `struct` is a collection of *members*:

```c
struct student {
    char name[21];
    int student_num;
};
```

- Can be dynamically or statically allocated
- Can declare arrays of structs, pointers to structs. . .
- What is `sizeof(struct student)`?
  - This is a “trick” question

PCRS: Streams and Files Recap

- `printf` and `scanf` operate on `stdout` and `stdin`, respectively
- `stdin`, `stdout`, and `stderr` are the standard I/O streams
- But streams can be associated with an open file
- Use `fopen` and `fclose` to open and close files

PCRS: Stream I/O Recap

- Functions are available to read/write on **any** stream:
  - Characters/strings: `fprintf`, `fscanf`, and `fgets`
  - Binary data: `fread`, `fwrite`
- Use `fseek` and `rewind` to *reposition* a stream
- Use `fflush` to *flush* a stream
- The man pages for all of these functions are very detailed and informative

Testing your understandingn

On the following slides and code snippets, assume . . .

- file pointers are set up correctly
- the memory region `buf` is defined with plenty of space

Testing your understanding (1/3).

- Assume you are reading a file that is 2MB large and this code is in a loop.
- After this call completes for the very first time, how many bytes were read?

```
fread(buf, 11, 19, fp);
```

Possible answers: 11, 19, 209 (consult the man pages)

Testing your understanding (1/3). *Answer*

- Assume you are reading a file that is 2MB large and this code is in a loop.
- After this call completes for the very first time, how many bytes were read?

```
fread(buf, 11, 19, fp);
```

Possible answers: 11, 19, 209 (consult the man pages) ==> 209 bytes

```
fread(void *ptr, size_t size, size_t nitems,
      FILE *stream);
```

...reads `nitems` objects, each `size` bytes long,...

Testing your understanding (2/3).

```
int r;
while ((r = fread(buf, 2, 4, fp_in)))
  fwrite(buf, 2, r, fp_out);
```

Q: Does this copy the contents from the file pointed to by `fp_in` to `fp_out`? (yes/no)

Testing your understanding (2/3). *Answer*

```
int r;
while ((r = fread(buf, 2, 4, fp_in)))
  fwrite(buf, 2, r, fp_out);
```

Q: Does this copy the contents from the file pointed to by `fp_in` to `fp_out`? (yes/no)

No, files with odd byte counts will not be copied correctly.

Testing your understanding (3/3).

```
int r;
while ((r = fread(buf, 1, 4, fp_in)))
  fwrite(buf, 1, r, fp_out);
```

Q: Does this copy the contents from the file pointed to by `fp_in` to `fp_out`? (yes/no)

Testing your understanding (3/3). *Answer*

```
int r;
while ((r = fread(buf, 1, 4, fp_in)))
  fwrite(buf, 1, r, fp_out);
```

Q: Does this copy the contents from the file pointed to by `fp_in` to `fp_out`? (yes/no) Yes

Extra Slides

Reference on Function Declaration vs. Definition

- Function Declarations
- Function Definitions

Struct Padding

Running the sample code below, you can observe how the compiler adds padding bytes to structs to make them align with word boundaries.

The GNU C Reference Manual explains why this is done. See what happens when this behaviour is disabled by compiling with `-fpack-struct`.

```c
#include <stdio.h>

struct test {
    char b;
    int a;
};

int main() {
    struct test a[10];
    printf("Address of first element: %p\n", &a[0]);
    printf("Address of second element: %p\n", &a[1]);
    printf("Address of third element: %p\n\n", &a[2]);
    printf("Address of char in struct: %p\n", &(a[0].b));
    printf("Address of int in struct: %p\n\n", &(a[0].a));
    printf("Size of struct array: %ld\n", sizeof(a));
    return 0;
}
```