Q6: What is the purpose of a file stream, just files?
A "file stream" (or "file descriptor" in system calls) is the base interface to EVERYTHING external to RAM. This includes:

- Network Socket
- USB DEVICES
- STANDARD ~~FILE~~ files.
- Standard Streams:
  - CREATED BY DEFAULT BY OS (SHELL)
  - stdin: STDIN_FILENO == 0 (maps to keyboard)
  - stdout: STDOUT_FILENO == 1
  - stderr: STDERR_FILENO == 2 (C SHELL maps to console)

not buffered, ↳ buffered.

Q7: Writing to file streams: `fprintf`
What if the output of the following code snippet?

```
1:  fprintf(stderr, "CS 241: ");
2:  fprintf(stdout, "System ");
3:  fprintf(stderr, "Programming ");
4:  fprintf(stdout, "\n");
```

⇨ Result: CS 241: Programming System

Q8: What is asprintf()?

```
int asprintf(char **strp, const char *fmt, ...)
```

allocate
⇨ `char **strp:`
POINTER TO A STR TO BE ALLOCATE

⇨ `const char *fmt:`

From Friday: Pointer Arithmetic

```
1:  // Count the number of elements in an int-array
2:  // before the number -1 appears in the array:
3:  int count_before(int *array) {
4:    int *ptr = array;
5:
6:
7:    while (*ptr != -1) { ptr++; }
8:
9:
10:   return (ptr - array) / [illegible];
11: }
```

Debug Less: Use assert! DEFENSIVE PROGRAMMING
C provides the library macro assert that be used to find bugs in debugging and completely disappear in production code! Two modes:

- Debug mode (-g flag): STOP PROGRAM & PRINT BACKTRACE
- Production mode (#NDEBUG): DO NOTHING

Best Practice: Always assert pre-conditions and assumptions.

Puzzle 4: Putting today together!

```
1:  // Sum an array of positive numbers, storing
2:  // the result in `result` (by ref)
3:  void mysum(const int *ptr,  int *result) {
4:    assert ( ptr != NULL );
5:    assert ( result != NULL);
6:    *result = malloc(                    );
7:
8:
9:    while ( *ptr ) {
10:
11:
12:     sum += *(ptr++);
13:
14:
15:   }
16:
17:
18:   return sum;
19: }
```

CS 241 | Lecture Handout #3
January 25, 2016

Q1: How do I find out how to use malloc?

```
$ man malloc
```

q to quit

Puzzle 1: How do I find out how to use stat in C?

man stat -S2

↳ section 2

Q2: What are the manual sections?

- Section 2: system calls
- Section 3: C Lib functions
- Section 7: miscellanea.

Q3: How do I allocate and free heap memory in C?

- Allocate: malloc() calloc realloc()
  ↳ brk maps system adm
- Free: free()

Q4: Can I make a pointer *really free* by freeing it twice?

No

Q5: What do we call a pointer that has been free'd?

Dangling ptr

Best Practice: Always set free'd pointers to NULL.

```
1: // ... code ...
2: free(ptr);
3: ptr = 0;  // ptr = NULL;
```

Puzzle 2: Fix a custom string copy function:

```
1:  void mystrcpy(char *dest, const char *src) {
2:
3:
4:    while (*src) {
5:
6:      dest = src;
7:
8:      src++; dest++;
9:
10:
11:
12:
13:
14:   }   & dest = '\0'.
          *dest = *src.
```

Puzzle 2 - Walk Through

Type	Variable	Memory Addr.	Value
const char *	src	0x1000	Snowflake\0
char *	dest	0x2000	*(unknown)*

⇨ **Line 3:** What does (*src) do?
~~check if~~ current value at the

⇨ **Line 4:** What does (dest = src) do? address save in make dest to memory that point src points to src.

⇨ **Line 3..9:** When does the loop exit?

Puzzle 3: Fix a custom string duplication function:

```
1:  char *mystrdup(const char *src) {
2:
3:      (char*) malloc( strlen(src)+1 );
4:    char *p = sizeof(src);
5:
6:
7:    strcpy(src, p);
8:
9:
10:   return p;
11: }
```