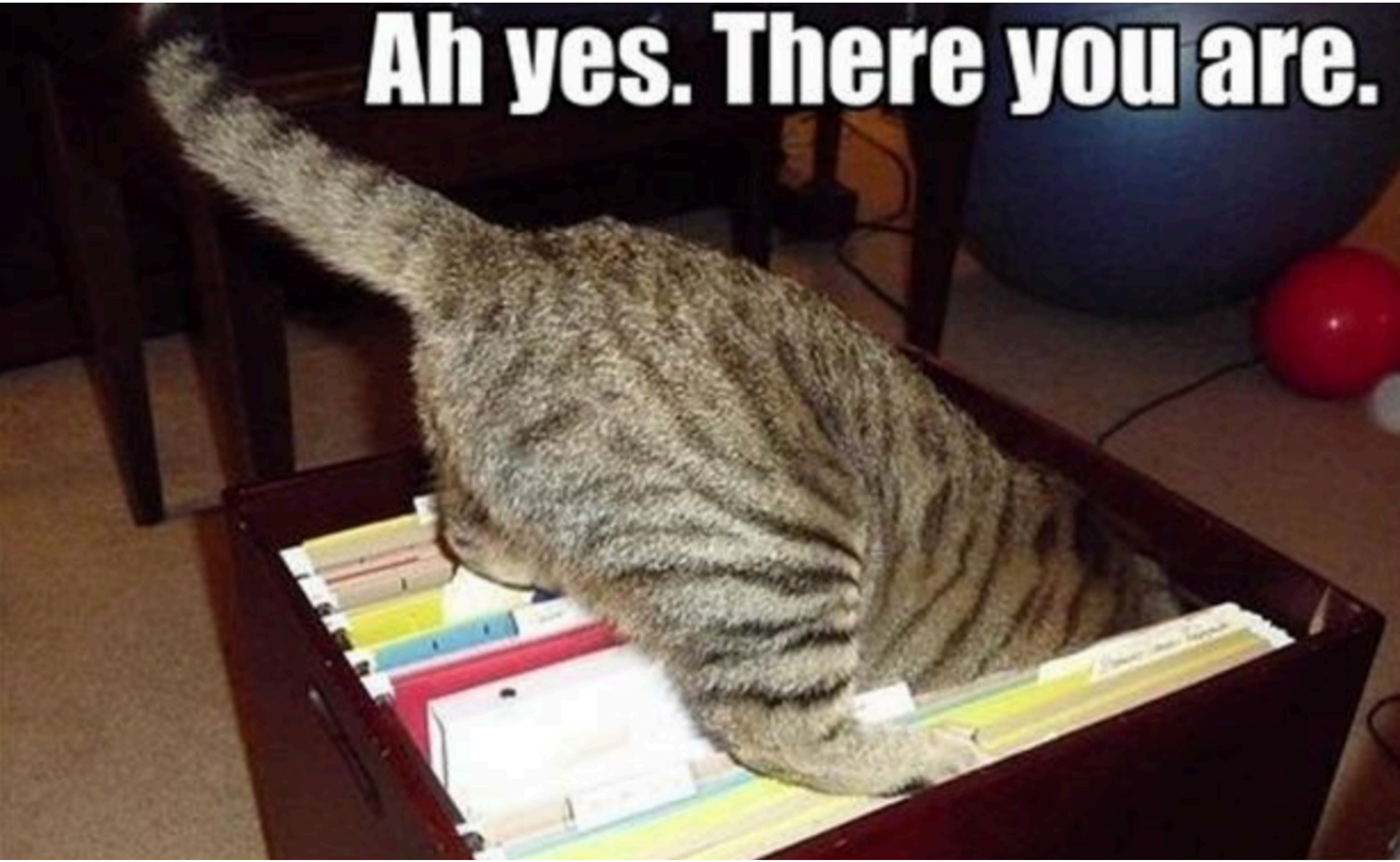
Week 8 Lecture 1: More Files!



Review and Preview

- Files and File I/O

- Files and the Operating System
- Assignment #3

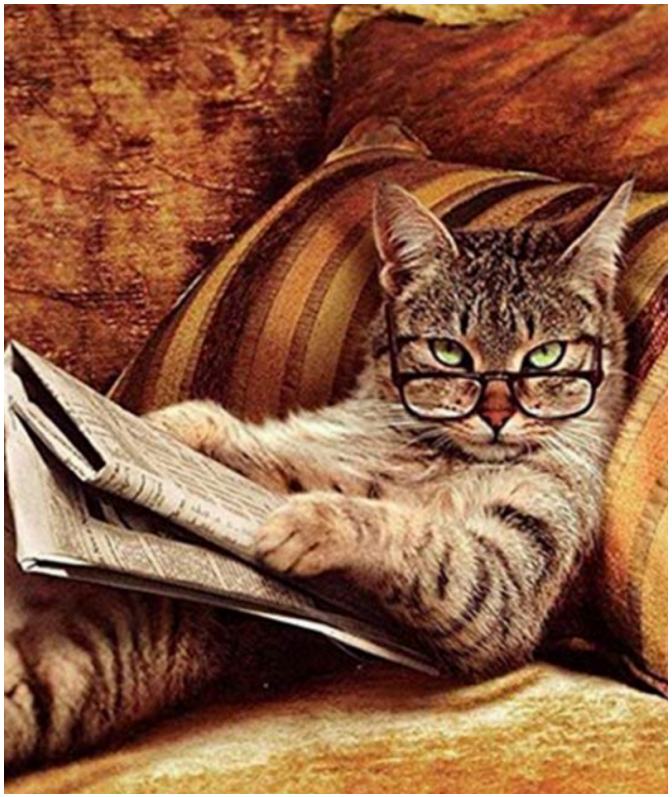
K&R: Chapters 1-5
Sections 4.5, 4.11
Sections 5.1-5.5
Sections 6.1-6.3

Files in UNIX

How does the OS see files?

Information about a File

- The operating system is responsible for the file system and has to maintain information about each file.
- What types of information does it need to know?



UNIX Command of the Day: `stat`

```
$ stat fileBasics.c
File: fileBasics.c
Size: 699         Blocks: 8
                IO Block: 1048576 regular file
Device: 2bh/43d  Inode: 254           Links: 1
Access: (0644/-rw-r--r--)  Uid: ( 1000/
debs)   Gid: ( 1001/    debs)
Access: 2019-10-23 14:35:45.000000000 -0400
Modify: 2019-10-23 14:35:45.000000000 -0400
Change: 2019-10-23 14:35:45.000000000 -0400
 Birth: -
```

-rwxr-xr-x

-rw-r--r--

Remember `ls -l` ?

```
debs@deb-socs:~/CIS1300/Assignments/Assignment2$ ls -l test*
-rwxr-xr-x 1 debs debs 16816 Oct 27 16:18 testChop
-rw-r--r-- 1 debs debs   340 Oct 27 14:31 testChop.c
-rw-r--r-- 1 debs debs  1776 Oct 27 16:18 testChop.o
-rwxr-xr-x 1 debs debs 16840 Oct 27 16:18 testConvert
-rw-r--r-- 1 debs debs   459 Oct 27 14:18 testConvert.c
-rw-r--r-- 1 debs debs  1792 Oct 27 16:18 testConvert.o
-rw-r--r-- 1 debs debs    81 Oct 27 14:30 testfile1.txt
-rw-r--r-- 1 debs debs    34 Oct 27 15:17 testfile2.txt
-rwxr-xr-x 1 debs debs 17136 Oct 27 17:14 testReduceSpace
-rw-r--r-- 1 debs debs   461 Oct 27 14:59 testReduceSpace.c
-rw-r--r-- 1 debs debs  2096 Oct 27 16:18 testReduceSpace.o
-rwxr-xr-x 1 debs debs 16840 Oct 27 16:18 testReplaceDigits
-rw-r--r-- 1 debs debs   349 Oct 27 14:35 testReplaceDigits.c
-rw-r--r-- 1 debs debs  1800 Oct 27 16:18 testReplaceDigits.o
-rwxr-xr-x 1 debs debs 16840 Oct 27 16:18 testReplacePunc
-rw-r--r-- 1 debs debs   347 Oct 27 14:42 testReplacePunc.c
-rw-r--r-- 1 debs debs  1792 Oct 27 16:18 testReplacePunc.o
-rwxr-xr-x 1 debs debs 17184 Oct 27 17:13 testTrim
-rw-r--r-- 1 debs debs   482 Oct 27 16:41 testTrim.c
-rw-r--r-- 1 debs debs  2160 Oct 27 16:42 testTrim.o
```

Owner Group

```
-rwxr-xr-x 1 debs debs 16840 Oct 27 16:18 testConvert
```

The Long Format of `ls`

- File mode: `-rwxr-wr-w`
- Number of links: `1`
- Owner name: **`debs`**
- Group name: **`debs`**
- Number of bytes in the file: `16840`
- Abbreviated month, day-of-month file last modified: `Oct 27`
- Hour, minute file last modified: `16:18`
- Pathname: **`testConvert`**

Files, Users, and Permissions

- **User**: a user is the owner of the file.
- **Group**: a group can contain multiple users and are identified by the system with a specific gid (group id).
- **World/Other**: any other user who has access to the file.

```
$ ls -l days*
-rwxr-xr-x 1 debs debs 16720 Sep 24 14:49 daysCalculatorA
-rw-r--r--  1 debs debs  2318  Sep 24 14:48 daysCalculatorA.c
```

File Permissions

- **Read**
  - *File*: permission to open and read a file
  - *Directory*: permission to lists its content
- **Write**
  - *File*: permission to modify the contents of a file
  - *Directory*: permission to add, remove and rename files in the directory
- **Execute**
  - *File*: permission to run (execute) an executable file / program

File Permissions: Octal Format

- Values for read, write and execute permissions in octal are:
  - 4 = Read permission
  - 2 = Write permission
  - 1 = Execute permission

Read	Write	Execute	
1	0	0	*4*
0	1	0	*2*
0	0	1	*1*
1	1	0	*6*
1	1	1	*7*

`0644 = Read & Write / Read / Read`

File Permissions: Character Format

- File Type: First character is the type of the file.
- **User** Permission: 2nd - 4th characters specify the read (r), write (w), and execute (x) permission of the **user**.
- **Group** Permission: 5th - 7th characters specify the read (r), write (w), and execute (x) permission of the **group**.
- **World** Permission: 8th - 10th characters specify the read (r), write (w), and execute (x) permission of the **world**.

Regular file `-rw-r--r--`

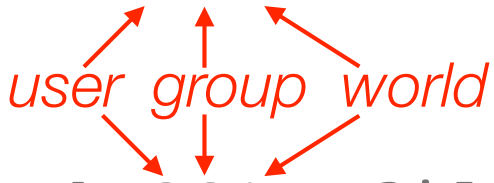
Directory `drw-r--r--`

`t|uuu|ggg|www`

`chmod` - Octal Mode

- `chmod ooo file`

```
$ chmod 744  file    -rwx-r—r—
$ chmod 664  file    -rw-rw-r—

$ chmod 777 file     -rwx-rwx-rwx
```



`chmod` - Symbolic Mode

```
$ chmod ugoa +- rwx
```

- **u** – user , **g** – group, **o** – others , **a** – all
- **+** to add permission , **–** to remove permission
- **r w x** is used for read , write, execute

```
$ chmod u+rwx,g+rw,o+r  filename
```

  - User permissions are read, write, execute
  - Group permissions are read and write
  - Other/World permissions are read

```
debs@deb-socs:~/Programs$ ls -l days*
-rwxr-xr-x 1 debs debs 16720 Sep 24 14:49 daysCalculatorA
-rw-r--r-- 1 debs debs  2318 Sep 24 14:48 daysCalculatorA.c
debs@deb-socs:~/Programs$
debs@deb-socs:~/Programs$ chmod g+w,o+w daysCalculatorA.c
debs@deb-socs:~/Programs$
debs@deb-socs:~/Programs$ ls -l daysCalculatorA.c
-rw-rw-rw- 1 debs debs 2318 Sep 24 14:48 daysCalculatorA.c
debs@deb-socs:~/Programs$
debs@deb-socs:~/Programs$ chmod 777 daysCalculatorA
debs@deb-socs:~/Programs$
debs@deb-socs:~/Programs$ ls -l daysCalculatorA
-rwxrwxrwx 1 debs debs 16720 Sep 24 14:49 daysCalculatorA
```

`chown` - Change file owner

- Changes the ownership of the file.
- You must have permission to do this - either you are the current owner or the superuser.

```
$ chown owner filename
```

`chgrp` - Change group of the file

- Changes the group that the file is in.
- You must have permission to do this - either you are the current owner or the superuser.

```
$ chgrp group filename
```

file - Determines file type

```
$ file testChop.c
```

testChop.c: C source, ASCII text

```
$ file chop.o
```

chop.o: ELF 64-bit LSB relocatable, x86-64, version 1 (SYSV), not stripped

```
$ file testChop
```

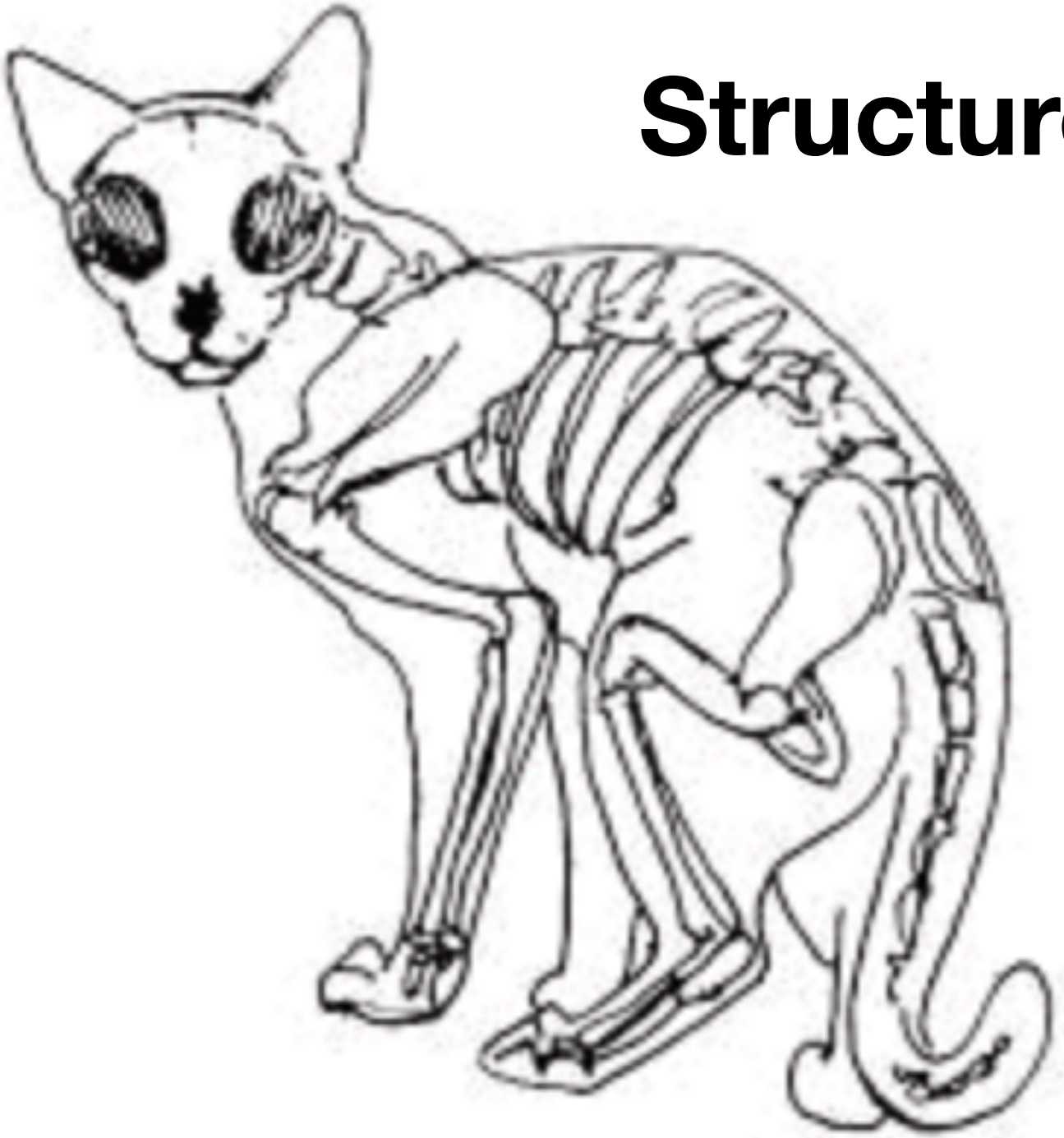
testChop: ELF 64-bit LSB pie executable, x86-64, version 1 (SYSV), dynamically linked, interpreter /lib64/ld-linux-x86-64.so.2, for GNU/Linux 3.2.0, BuildID[sha1]=2db09aa8addd6f6b97cd91f7f0c3f2f9f6b3854e, not stripped

`touch` - changing a file's modification date

- `touch` lets you change the date on a file. It can also be used to create a blank file.

```
$ ls -l chop*

-rw-r--r-- 1 debs debs  105 Oct 27 14:12 chop.c

-rw-r--r-- 1 debs debs 1424 Oct 27 16:18 chop.o

$ touch chop.c

$ ls -l chop*

-rw-r--r-- 1 debs debs  105 Oct 28 16:22 chop.c

-rw-r--r-- 1 debs debs 1424 Oct 27 16:18 chop.o
```

Structures in C



Structures

- A **structure** is a *user-defined* datatype in C.

- It allows a combination of data of different types unlike an array that contains only data of the same type.

```
struct tag {

    member variable

    member variable

};

struct tag strName;
```

The *tag* names this structure and can be used subsequently as a shorthand for the part of the declaration inside the braces.

The *member variables* can be of any type including other structures!

Declaration of a variable of type *tag*

Structure Example

```
struct student {
    char firstName[30];
    char lastName[50];
    unsigned int idNum;
    unsigned int semesterLevel;
};
struct student singleStudent;
```

Create the new “type"

Declare a variable of type “student”

```
singleStudent.idNum = 16377809;
singleStudent.semesterLevel = 1;
strcpy (singleStudent.firstName, “Charles”);
strcpy (singleStudent.lastName, “Brown”);
```

Assign values to parts of singleStudent

Using Structures in Assignment 3

- Popular Names (United States Social Security)

  https://www.ssa.gov/OACT/babynames/index.html

- All names are from Social Security card applications for births that occurred in the United States after 1879.

  - Note that many people born before 1937 never applied for a Social Security card, so their names are not included in the data.

- All data are from a 100% sample of records on Social Security card applications as of March 2019.

Social Security

SEARCH MENU LANGUAGES SIGN IN/UP

Top names of the 1880s

Popular Baby names

Select another decade?

Decade Go

Top names in last 100 years

Top 5 names in each year

Popular Names by State

The following table shows the 200 most popular given names for male and female babies born during the 1880s. For each rank and sex, the table shows the name and the number of occurrences of that name. The 200 most popular names were taken from a universe that includes 1,177,162 male births and 1,399,569 female births

Popular names of the period 1880s

	Males		Females	
Rank	**Name**	**Number**	**Name**	**Number**
1	John	89,950	Mary	91,668
2	William	84,881	Anna	38,159
3	James	54,056	Emma	25,404
4	George	47,651	Elizabeth	25,006
5	Charles	46,656	Margaret	21,799
6	Frank	30,967	Minnie	21,724
7	Joseph	26,292	Ida	18,283

https://www.ssa.gov/OACT/babynames/decades/names1880s.html

Names by Decade: 1880 to 2010

Rank	Male Name	#	Female Name	#
1	John	89,950	Mary	91,668
2	William	84,881	Anna	38,159
3	James	54,056	Emma	25,404
4	George	47,651	Elizabeth	25,006
5	Charles	46,656	Margaret	21,799
6	Frank	30,967	Minnie	21,724
7	Joseph	26,292	Ida	18,283
8	Henry	24,139	Bertha	18,263
9	Robert	24,074	Clara	17,717
10	Thomas	23,750	Alice	17,142

1880Names.txt

```
#define MAXLENGTH 20
struct popNames {
    int year;
    int rank[200];
    char maleName[200][MAXLENGTH];
    int maleNumber[200];
    char femaleName[200][MAXLENGTH];
    int femaleNumber[200];
};
struct popNames popular;
```

```
if ( (f1 = fopen(argv[1], "r")) != NULL ) {
   index = 0;
   while ( fgets(string, 100, f1) != NULL ) {
      sscanf (string, "%d %s %s %s %s",
          &popular.rank[index],
          popular.maleName[index], maleSNumber,
          popular.femaleName[index], femaleSNumber);

      removeCommas ( maleSNumber );
      popular.maleNumber[index] = atoi(maleSNumber);

      removeCommas ( femaleSNumber );
      popular.femaleNumber[index] = atoi(femaleSNumber);
      index++;
    }
}
```

1	John	89,950	Mary	91,668
%d	%s	?	%s	?

Is there an issue with using fscanf to read integers that have a comma in them?

How easy is it to remove the commas before converting the character string into an integer?

```
for ( i=0; i<10; i++ ) {
    printf ("%d (%s %d) (%s %d)\n",
    popular.rank[i],
    popular.maleName[i], popular.maleNumber[i],
    popular.femaleName[i], popular.femaleNumber[i]);
}
```

```
./example 1880Names.txt
1 (John 89950) (Mary 91668)
2 (William 84881) (Anna 38159)
3 (James 54056) (Emma 25404)
4 (George 47651) (Elizabeth 25006)
5 (Charles 46656) (Margaret 21799)
6 (Frank 30967) (Minnie 21724)
7 (Joseph 26292) (Ida 18283)
8 (Henry 24139) (Bertha 18263)
9 (Robert 24074) (Clara 17717)
10 (Thomas 23750) (Alice 17142)
```