Assignment 3

In this assignment, you will create a simplified version of the `cat` program using the code examples for copying from lecture 3. There are two parts. You must complete both parts for an A grade.

`cat` reads the contents of a file and writes it to standard output (`stdout`). You already know one way to write to `stdout`. That is to use `printf`.

Problem 1.

Look at the source code for the program `better_copy.c`. You can use any of the code in this or other program examples.

copy the file to another file named `cat1.c`. In this new file, make the following changes:

1. Change the documentation at the top of the file to be appropriate for this new file.
2. Look at the program and remove all references to `outfile`. You won't need to read the name from the command line, open, or close this file.
3. Remove all references to `fdwt`.
4. In the loop that writes to `fdwt`, replace the if statement with `printf("%s\n", buffer);` to print the buffer to `stdout`.
5. Compile and run the resulting program. fix any bugs.
6. Run this program using the files found in the `corpora` folder or any other file you want to `cat` to the terminal.

Problem 2

The file descriptors for `stdin` and `stdout` are already open when your program starts running. You don't need to open or close them while your program runs, so they are convenient for doing file I/O (input/output). In this problem, you will see how we can use `stdout` instead of a specified output file.

1. Copy the file `better_copy.c` to a file called `cat2.c`. Open the new file and fix up the header documentation.
2. Remove the open and close of rdwt and references to `outfile`.
3. In the read/write loop, replace the `fdwt` file descriptor with the special constant `STDOUT_FILENO`.
4. Compile the program, fix bugs, and run it as in problem 1.
5. You can copy to any other file name using **redirection** of `stdout` at the terminal. You can do this by typing:

```
cat2 ../corpora/carroll-alice.txt > alice-copy.txt
```

This will copy the contents of *alice in wonderland* to new file in your program's directory. The `>` forces `stdout` to point at a new file.