

**\*\* OPEN BOOK -- OPEN NOTES \*\***

(1) (18 points, 6 points each) Assume we are reading someone else's code which has some missing lines and they have a function that begins: `int writeout(char * fname, char * outline)`, with a comment that says the function is supposed to open the named file `fname`, write out a line, `outline`, to the end of it without destroying any of the lines that already exist, and close the file, returning -1 if there is an error and 0 otherwise. After this, there is a declaration:

```
FILE *x; int status1;
```

(a) A comment for a single line of code says to open the file in the proper mode, setting `x` to the value returned. The code line is missing. Fill it in.

(b) The next comment for a single line of code says to put out the line to this file using `fputs`, and set the integer variable `status1` to the returned value. Do this.

(c) Finally there are several lines missing, and the comment says to close the file and return 0 if there was no error returned in `x` or `status1`, else to return -1. Do this. (Note that you should really check the value for `x` right after `fopen`, before trying to execute `fput`.)

(2) (12 points) (a) (8 points) We are given a program with the following declaration:

```
int x; double f; char intval[] = "1234"; char floatval[] = "1234.567"; int status1;
```

Write two `sscanf` calls (setting `status1` in each case to receive the returned value) to fill in `x` from `intval` (`atoi` function equivalent) and `f` from `floatval` (`atof` equivalent).

(b) (4 points) Give the test of status1 you would use after each sscanf above to indicate that the function was successful.

### First Practice Quiz 3 Solutions

```
(1) (a) x = fopen(fname, "a")           /* returns stream, NULL == 0      */
(b) status1 = fputs(outline, x);        /* returns non-neg if OK, EOF if error*/
(c) fclose(x);                          /* EOF if error                    */
    if(x == NULL || status1 == EOF)
        return -1;
    else return 0;

(2) (a)
    status1 = sscanf(intval, "%d", &x);   /* note importance of "&" in "&x"  */
    status1 = sscanf(floatval, "%f", &f); /*                                */

(b) if (status1 == 1)                    /* if test is status1 != EOF, slightly wrong */
    then <successful>                     /* any way you want to indicate this is fine */
```

**\*\* OPEN BOOK -- OPEN NOTES \*\***

(1) (30 points, pts as marked) You are asked to write a program called "copy.c", which will be compiled to the executable "copy" as follows: `gcc copy.c -o copy`. The executed copy program will accept two arguments (but no options) with a command form: `copy fnamefrom fnameeto`. The program will open the file `fnamefrom` in the proper mode to read lines from it, then open the file `fnameeto` in the proper mode to destroy any contents that previously existed and write the lines from `fnamefrom` to it until the file is copied. We will do this by stages.

(a) (3 pts) The program `copy.c` will consist of a single main function that provides for exactly two arguments. Write the introductory line of a main function to accept the arguments.

Next, assume you have the following declarations:

```
char fnamefr[20], fnameeto[20], line[1000]; int i, status1; FILE *x, *y;
```

Assume `fnamefr` and `fnameeto` are large enough to contain the names of the files in the two arguments of main and the array `line[ ]` is large enough to contain any line from the files.

(b) (8 pts) Count the number of arguments and return 1 if there are not exactly two (DON'T put out error msgs or anything -- just return 1). Then copy the arguments into the `fnamefr` and `fnameeto` arrays. USE THE FUNCTION `strncpy()`. All this should require 3 to 4 lines.

(c) (3 pts) Open the `fnamefr` file in the proper mode, setting `x` to the value returned. Return 2 if there is an error. (Use a single if statement where the `fopen` is evaluated inside the test.)

(d) (3 pts) Open the `fnameeto` file in the proper mode, setting `y` to value returned. Return 3 for error. (Use a single if statement as in part (c)>)

(e) (8 pts) Perform a while loop with a loop test containing an fgets function to read in line[ ] from x and an fputs function to write out line[ ] to y. The loop should terminate when fgets indicates there are no more lines to read in from x (assume this is not an error) and return 4 if fputs gives an error.

(f) (3 pts) Close both files; return 5 if there is an error closing either file; else return 0.

(g) (2 pts) There are two #include statements you should have at the top of this program. Write them here.

### Second Practice Quiz 3 Solutions

```
#include <stdio.h>
#include <string.h>

main(int argc, char *argv[])
{
    char fnamefr[20], fnameto[20], line[1000]; int i, status1; FILE *x, *y;

    if (argc != 3) return 1;
    strncpy(fnamefr, argv[1], 20);
    strncpy(fnameto, argv[2], 20);

    if((x = fopen(fnamefr, "r")) == NULL) return 2;
    if((y = fopen(fnameto, "w")) == NULL) return 3;

    while(fgets(line, 1000, x))
        if(fputs(line, y) == EOF) return 4;

    if(fclose(x) == EOF || fclose(y) == EOF)
        return 5;
    return 0;
}
```

