CS360 Lab #1

Due 11:59PM, Tuesday, August 27, 1996

This assignment concentrates on using the `fields.o` set of procedures. See the lecture notes in the directory `/bl(green/homes/plank/cs360/notes/Fields` for instruction on how to use the `fields.o` procedures.

`mycat`, `myhead`, `mywc`, and `myfill`

Implement `mycat`, `myhead`, `mywc`, and `myfill`, which are defined with the following man page. Use the `fields.o` set of procedures to read the input.

The first three of these programs are very simple (my versions are 53, 59, and 60 lines respectively). With `mycat` and `myhead`, you only need to access the `line` and `text1` fields of the `IS` struct. With `mywc`, you only need to access the `text1` and `NF` fields (and the `line` field if you want).

`Myfill` is more complex. Here, you'll need to access the `fields` part of the `IS` struct, and do a little more thinking. Still, the final program does not have to be too large (mine is 89 lines).

Error Conditions

Your program should flag an error if the user specifies an invalid command line argument, or a bogus one (like "`myhead --9`", "`myhead -jim`", "`myfill -c 0`" or "`myfill -c -rj`"). Error statements should be printed to stderr, and the program should exit with "`exit(1)`".

Your program should do something reasonable if you encounter problemmatic input – such as a word with more than 72 characters. You should explain what your program does in such cases.

Working Programs

Working versions of `mycat`, `myhead`, `mywc`, and `myfill` are in the directory `/bl(green/homes/plank/cs360/bin`. If you have a question about what your program should do, first see what these programs do.

Useful Utilities

If you don't know them already, you should read the man pages on `strcpy`, `strlen`, `atoi`, `isdigit` and `sscanf`. These can all be helpful to you.

```
MYCAT(1)                 USER COMMANDS                  MYCAT(1)

NAME
     mycat - concatenate and display

SYNOPSIS
     mycat [ -n ] [ filename...  ]

DESCRIPTION
     mycat reads each filename in sequence and displays it  on  the
     standard output.  Thus:

          UNIX> mycat file1

     displays the contents of file1 on the standard output, and

          UNIX> mycat file1 file2 > file3

     concatenates the first two files and places  the  result  on
     the third.

     If no filename argument is given, or if the argument '-'  is
     given,  mycat  reads from the standard input.

OPTIONS
     -n   Precede each line of output with its line number.   Take
          care to make the output look as in  cat -n.
```

```
MYHEAD(1)                 USER COMMANDS                 MYHEAD(1)

NAME
     myhead - display first few lines of specified files

SYNOPSIS
     myhead [ -n ] [ filename...]

DESCRIPTION
     myhead copies the first n lines of each filename to the  stan-
     dard  output.   If  no  filename is given, myhead copies lines
     from the standard input.  The  default  value  of  n  is  10
     lines.

     When more than one file is specified, the start of each file
     looks like:

          ==>filename<==

EXAMPLES
     The following example:
         UNIX> myhead -4 /blugreen/homes/plank/cs360/notes/Dlists/h*

     produces:
         ==> /blugreen/homes/plank/cs360/notes/Dlists/hostread.c <==
         #include <stdio.h>
         #include <string.h>    /* This contains the definition ...
                                   the man page */


         ==> /blugreen/homes/plank/cs360/notes/Dlists/hosts <==
            128.169.202.200            cocker.vet.utk.edu
            128.169.201.208         berkelium.chem.utk.edu
            128.169.201.152            cherokee.gg.utk.edu
              192.31.99.193              slipgate1.utk.edu

     The following example:
         UNIX> myhead -8 /blugreen/homes/plank/cs360/notes/Dlists/hosts

     produces:
            128.169.202.200            cocker.vet.utk.edu
            128.169.201.208         berkelium.chem.utk.edu
            128.169.201.152            cherokee.gg.utk.edu
              192.31.99.193              slipgate1.utk.edu
              192.249.11.79            csmacw14.cs.utk.edu
               128.169.94.12           cygnusx1.cs.utk.edu
               128.169.93.24               pepsi.cs.utk.edu
               128.169.92.48             wham-o.cs.utk.edu
```

```
MYWC(1)                    USER COMMANDS                    MYWC(1)

NAME
     mywc - display a count of lines, words and characters

SYNOPSIS
     mywc [ filename ...  ]

DESCRIPTION
     mywc counts lines, words, and characters in filenames,  or  in
     the  standard input if no filename appears.  It also keeps a
     total count for all named files.  A  word  is  a  string  of
     characters delimited by SPACE, TAB, or NEWLINE characters (i.e.
     a word is as defined in fields.o)

OPTIONS
     When filenames are specified  on  the  command  line,  their
     names will be printed along with the counts.

EXAMPLE
          UNIX> cd /blugreen/homes/plank/cs360/notes/Fields
          UNIX> mywc *
                   39      79     647 badword.c
                   41      86     733 goodword.c
                  258     941    7343 lecture
                   21      56     484 makefile
                   30      59     451 printwords.c
                    3      18      90 rex.in
                  392    1239    9748 total
          UNIX> mywc < lecture
                  258     941    7343
          UNIX>
```

```
MYFILL(1)                  USER COMMANDS                  MYFILL(1)

NAME
     myfill -- primitive text formatting

SYNOPSIS
     myfill [ -rj ] [ -c num ]

DESCRIPTION

Myfill reads its input on standard input, strips out excess white
space, and fills each paragraph to be 72 characters, maximum.  Thus,
the following input:

----------------------------------------------------------------------
Dear Abby:   Almost all          pharmacies use ''Rx''
        as an identifying
symbol, so I have been asking pharmacists the origin of this
symbol.
----------------------------------------------------------------------

would be converted to:

----------------------------------------------------------------------
Dear Abby: Almost all pharmacies use ''Rx'' as an identifying symbol,
so I have been asking pharmacists the origin of this symbol.
----------------------------------------------------------------------

If the input has entirely blank lines (or lines with only white
space), then myfill should retain them.  For example,
suppose the input is the following:

----------------------------------------------------------------------
''June, why didn't you tell me
if you didn't like my cooking?''

''Rex, I never said your cooking
was bad -- just that it was lacking in
variety.  Don't take it so personally.''


''I'm not taking it personally!''
----------------------------------------------------------------------

Then the output should be:

----------------------------------------------------------------------
''June, why didn't you tell me if you didn't like my cooking?''

''Rex, I never said your cooking was bad -- just that it was lacking in
variety.  Don't take it so personally.''


''I'm not taking it personally!''
----------------------------------------------------------------------

Myfill should take two optional command line arguments:

-rj: This means to right justify the text: Put one space between
     words, and put enough spaces in the beginning of each line to
     make sure that the last word on each line ends on the 72nd
     character.

-c num: This means to fill to num columns instead of 72. This should
        work with the other command line arguments.

The order of the command line arguments does not matter (i.e.
either -rj or -c can come first).
```