RETAB   USERS GUIDE

Developed by: Paul Klink
Documentation covers release: 1.03
Release Status:ShareWare   (See end of doc.)

OVERVIEW

    Retab is a task that allows you to reformat ascii text files
so that the tab characters represent a different number of
characters. This is best explained by explaining the reason why
I wrote this program. It all started like this ....

    I have a C compiler so I write C programs. When one writes
C source files one indents various lines of text in the source
file to make it easier to understand. For example:

```
printf("Print this\n");
i = 0;                          /*initialise i*/
while (i < 10)
{
  printf("i now equals %d\n", i);
  i = i + 1;                    /*increment i*/
}
printf("Counting finished\n");
```

Ignore all the gobbledygook and just note how 2 of the lines are
indented and 2 of them have comments after them. One can indent
them by either typing in the extra spaces before the line or one
can indent them by inserting tab characters. I find editing much
easier if I indent the lines in my source files by using tabs.
No problems so far.

    The problem begins when one gets fussy about how far one
wants to indent lines. Over the years I have determined that for
me, the optimum indent size for C source files is 3 characters.
So I set up my text editor accordingly, and write my files. Now
comes the problem! When I either:
type the files to the screen,
print the files on particular printers
or even, try to debug the program under Lattice C CodeProbe
debugger,
the whole file format is a mess as the tab characters are
interpreted to represent 8 characters rather than 3 characters as
my editor did. Columns don't align, lines are far longer, words
are not where I expect them etc. Even worse, some devices I have
used do not like any tab characters.

My solution to this problem is this program. It will add/remove spaces/tabs so that the file is reformatted for an environment that uses a different size tab. Then you can print the file out in that environment and it will look exactly the same as it did in the other environment.

A quick definition of a tab character. This is 'the ascii character that is inserted in the file when the TAB key on the keyboard is pressed. This is almost always ascii 9'. Note: This does not include the case where the TAB character is expanded to the relevant number of space characters.

This program can operate on any text file. It can reformat it for new tab size, convert all tabs into spaces or it can convert spaces into tabs. It will follow a set of rules that ensure that the reformatting is done correctly. C programmers take note, an option exists that will ensure that any white space enclosed within double quotes will not be reformatted. Even C escaped double quotes (ie. \") and C comments are handled with this option. Another option will allow you to protect white space that is enclosed within any specified ascii character. Another option allows you to trim all spaces and tabs at the end of the lines. You can even use the Amiga shell environmental variables to specify various options.

Now down to the nitty gritty.

OPERATION INSTRUCTIONS

     The process Retab can only be used from the CLI or SHELL
environment. The following command line starts the process
retab.

          retab [opts] <source file> [opts] [<dest. file>] [opts]

     where:
          optsare sets of options that can be specified.
<source file>is the name of the source file. This file holds the
data that is to be reformatted.
<dest. file>is the name of the file to which the reformatted data
will be sent.
[]All parameters enclosed within square brackets are optional.

As you can see, the only parameter that has to be included is the
source file name. If the destination file name is not specified,
then the source file is overwritten with the reformatted data.

     The options let you control how the reformatting is to be
done. If an option is not specified on the command line, then a
default value will be used for that option. In some cases, the
default can be specified by Amiga environmental variables.   The
following four notes relate to how options are specified on the
command line.

NOTES:*The order of options on the command line is not important.
*The option letters are shown in lower case letters below. They
can be entered in either UPPER or LOWER case letters.
*Some options may be followed by some data relating to that
option (Eg. A Number). The option and the data may be separated
by an equal sign (=). The equal sign is shown below within
square brackets to show that it is optional. Do not type in the
square brackets in any case.
*Options can be strung together in one command line parameter.
For example:
ntspecifies both options n and t
ntalso specifies options n and t
nti=3specifies options n, t and i. The i option uses the data 3.
If an option is followed by data, that option must be the last
option specified in the command line parameter.

The next section describes all the options that can be specified
on the command line.

i[=]<source file tab size>

This option specifies what number of characters the tabs
represent in the source file. The number can be in the range of
0 to 32767. If i=0 is specified, then the following happens.
Retab will replace strings of white space with tab characters.
Any existing tab characters in the source file represent a size
as specified by the o option.
If i=0 and o=0, then no retabbing will occur.
If this option is not specified on the command line, then retab
will try to obtain this value by examining the number inside the
environmental variable 'ReTabInTab'. If this environmental
variable does not exist, or it does not contain a valid numerical
string, or the n option is specified, then the default value of 8
is used.

o[=]<destination file tab size>
This option specifies what number of characters the tabs will
represent in the destination file. The number can be in the
range of 0 to 32767. If o=0 is specified, the following happens.
Retab will expand all tabs in the source file into the relevant
number of spaces in the destination file.
If this option is not specified on the command line, then retab
will try to obtain this value by examining the number inside the
environmental variable 'ReTabOutTab'. If this environmental
variable does not exist, or if it does not contain a valid
numerical string, or the n option is specified, then the default
value of 0 is used.

t
This option will trim all lines of any trailing spaces and/or
tabs as it reformats it. That is, the line will end at the last
visible character. This option will also be active if the
environmental variable 'ReTabTrim' exists and the n option is not
specified. It does not matter what ReTabTrim contains.
p[[=]<protect character>]
In some cases you might want spaces and tabs that are enclosed
within a certain character, not to be reformatted. This will be
especially true of strings within language source files.
Example: (Say a variant of BASIC)
PHONE1$ = 'J. Smith   8765432'
PHONE2$ = 'S. Jackson   7654321'
In this example, you do not want to reformat any tabs or spaces
between the character ' hence, in the command line you would
specify <p='> (do not include the <> characters).
If there are an odd number of <protect characters> on the line,
then all of the tabs and spaces after the last <protect
character> to the end of the line are protected. Trimming though
will still occur if the t option is specified.

If p is placed on the command line without a <protect character>
then the default character " (double quote) is used.
If the p option is not specified on the command line, then (if
the n option is not specified) retab will see if the
environmental variable 'ReTabProtect' exists. If so, the protect
feature will be active and the protect character will be the
first character in the string contained in the environmental
variable. A double quote character will be used if it holds a
Null string.

c
This option is very similar to the p=" option but it is used for
reformatting C source files. Like p=", it will protect all
spaces and tabs between double quote characters, but with the
following two differences.
It will detect C escaped double quotes, (that is, \") and it will
not consider these as protect characters.
It will not consider any double quote characters that are within
C comment fields to be protect characters.
Always use this option if you are reformatting C source files.
This option will also be active if the environmental variable
'ReTabCQuote' exists and the n option is not specified. It does
not matter what ReTabCQuote contains.
n
If this option is specified, it will disable retab from checking
for environmental variables if an option is not specified on the
command line. The inbuilt default will be used for all options
not specified.

l[=]<Maximum Source file line length>
This specifies the maximum length line the source file can have
within it. If a line in the source file is longer than this, the
task will abort the retab operation and print a message showing
which line number was too long. This feature will probably help
you if you inadvertently retab a binary file. It also is used
internally in the program to determine how much memory is needed.
The value specified must be in the range 0 to 32767. If this
option is not specified on the command line, a default value of
256 is used.

      ?
This option prints a help message to the standard output file
(normally the screen). This message summarises what retab does
and how it is used. It briefly describes all the options. The
message will also display the programs release number.

THE ENVIRONMENTAL VARIABLES

There are 5 environmental variables which can be set up to specify what actions the process retab should take if the relevant options are not on the command line. In effect, these environmental variables specify new defaults. The n option forces the program to ignore all environmental variables and use the inbuilt defaults. The use of the environmental variables is covered in the previous section, but they will be summarised below.

ReTabInTab   (i option)
This environmental variable defines the default size of tabs in the source file. It must contain a numerical ascii string in the range 0 to 32767.
ReTabOutTab   (o option)
This environmental variable defines the default size of tabs in the destination file. It must contain a numerical ascii string in the range 0 to 32767.
ReTabTrim   (t option)
If this environmental variable exists, then all lines will be trimmed of trailing spaces and tabs. The contents of ReTabTrim are ignored.ReTabProtect   (p option)
The first character within this environmental variable is the protect character to be used. If it does not contain any characters, a double quote character (") is used. All spaces and tabs enclosed within the protect character are not reformatted.
ReTabCQuote   (c option)
If this environmental variable exists, all spaces and tabs enclosed within double quote characters (") will not be reformatted. Double quote characters are not considered as protect characters in the following two instances. A C escaped double quote character (\") is not considered one of the protect characters. Double quotes within C comment fields are not considered as protect characters.

USAGE EXAMPLES

     The following examples demonstrate the usage of the task
retab.

a)If you have a file called 'mysrc' that in which tabs represent
3 spaces and you want them to represent 8 spaces:

          retab mysrc i=3 o=8

b)If you want to also trim all lines of trailing spaces and tabs:

          retab i=3 o=8 mysrc t

Note that the position of options in the command line do not
matter

c)You may always want to trim all lines. To save you from always
having to type in the option t, you may specify it with its
environmental variable.

          setenv ReTabTrim 1

          retab i=3 mysrc o=8
          retab i=3 yoursrc o=8

It does not matter what data the environmental variable ReTabTrim
contains.

d)You may want to leave the file 'mysrc' unchanged but create a
new file, 'newdest', which has the new format.

retab mysrc newdest i=3 o=8

e)If the new file is to have all the tab characters replaced by
spaces:

          retab i=3 mysrc o=0 spacdest

Once again, note that the position of the options in the command
line do not matter.

f)If you want to replace spaces in a file with tabs (you may find
it easier to edit the file this way):

          retab i=0 spacesrc o=4 tabdest

The file 'tabdest' will have as many spaces as allowable replaced
with tabs that represent a size of four characters.
g)You may need to do this for a lot of files. In this case it
would be easier to set up some environmental variables. You may
also want to trim the lines of some of these files.

```
        setenv ReTabInTab 0
        setenv ReTabOutTab 4
        setenv ReTabTrim

        retab spac1src tab1dest
        retab spac2src tab2dest
        retab spac3src tab3dest
        retab i=3 tab1src o=0 spac1dest
retab i=3 tab2src spac2dest n
```

The second last line in the above examples, show how the command
line options will overrule environmental variables. File
spac1dest will have no tabs within it. The last line shows the
action of the n option. File 'spac2dest' will not have any of
its lines trimmed. It will also have all its tabs replaced by
spaces as that is the inbuilt default.

h)C programmers should always reformat their C source files with
the c option.

```
        retab i=3 csrc.c co=8 cdest.c
            or

        setenv ReTabCQuote 1

        retab i=3 csrc.c o=8 cdest.c
```

i)Some other files may use other characters to mark strings or
other data. An example are the Lattice LMK files (makefiles).
They use the character < to enclose data.

```
        retab i=0 makesrc o=8 makedest p=<
```

j)If you have a file that has very long lines in it, say up to
10000 characters, you will have to use the l option.

```
        retab i=3 longsrc o=8 longdest l=10001
```

Do not make the l value too large or retab may not be able to run
as it may not be able to get enough memory for its internal work

areas. In this case you will get the message,
                  'Could not obtain the required memory!'
and it will not perform the retabbing operation.
k)If you have files with tab characters representing a size of 3
characters, you could develop a special 'type' command for these
files.

          alias mytype retab i=3 o=8 [] *

          mytype <source file>

l)Retab can be used to trim all lines of trailing spaces and tabs
without any retabbing of tabs within a file.

retab i0 o0 t srcfile

TECHNICAL INFORMATION

     This section presents some technical information about how
the task works.

The Retabbing
Retab identifies space/tab groups. Space/tab groups are sets of
consecutive characters that are either space or tab characters.
Space/tab groups can not cross line boundaries.
If the i option <> 0
The spaces and tabs within a space/tab group will be reformatted
if that space/tab group contains at least one tab character.
Otherwise the spaces in the space/tab group will not be retabbed.
If the i option = 0
As in the previous case, spaces and tabs within a space/tab group
will be reformatted if that space/tab group contains at least one
tab character. All tab characters represent a tab size as
specified by the o option.
If there are only spaces in the space/tab group, it will still be
reformatted if the group covers at least one tab division on the
line. A line is divided into tab divisions. The o option
specifies the size of these divisions. For example, on a line:
If o=3,
characters 1 to 3   1st tab division
characters 4 to 6   2nd tab division
characters 7 to 9   3rd tab division
                           etc.
If o=8,
characters 1 to 8   1st tab division
characters 9 to 16   2nd tab division
characters 17 to 24   3rd tab division
                           etc.
If i=0 and o=0 then tab divisions are undefined as no retabbing
takes place.
Note:If the i option does not equal 0, then the i option
specifies the tab division size in the source file and the o
option specifies the tab division size in the destination file.

Temporary files and multitasking.
If the retab is started with no destination file in the command
line (ie, source file will be overwritten), retab will create a
temporary file in which it places the reformatted data. In this
case, if the retabbing wassuccessful, then the source file is
deleted and the temporary file is renamed to the name of the
source file. This temporary file will always be created in the

same directory as the source file. It will have the name:

retab_xx.tmp

xx is a number between 00 and 99

Retab will always ensure that it does not overwrite an existing
temporary file. This allows retab to be made resident and more
that one copy of it can run simultaneously. Retab is a pure
process!
If for any reason retab is aborted, then you may have to purge
the directory of any retab temporary files.

Command line parameters
Some notes about command line parameters.
*For the i, o, and l options, white space may be placed between
the option and the option data. If the equal sign is used (=),
then it must immediately follow the option letter.
*For the p option, no white space can exist between the option
letter and the option data.
*If a file name contains white space, enclose it within double
quotes.
*If a file name begins with a dash (), then the file will have to
be renamed as retab cannot accept a file name beginning with a
dash.

SHAREWARE AND OTHER STUFF

     Three files should be included within this package:

retabThe executable process
retab.docThis documentation in ascii format
retab.wpfThis documentation in WordPerfect format. (Amiga
version 4.1)

No ancillary files are required when running retab.

     This is shareware and the normal shareware rules apply. If
you find this program useful you owe me. Say $20 (AUS). Send
cheques payable to an Australian Bank to my address:

Paul Klink
PO. Box 169
WooriYallock 3139
Australia

I'm interested in any feedback. Especially of bugs. If you find
any bugs, and want an immediate update, I'll try my best to fix
them as soon as possible and send you out an update BUT you will
have to have registered plus send $7 for disk and postage.

     If you are interested in the source files, write to me and I
will think about it.

          HAVE FUN