```
#include <stdlib.h>
#include <string.h>
#include <ctype.h>

#define MAXPAROLA 30
#define MAXRIGA 80

int main(int argc, char *argv[])
{
  int freq[MAXPAROLA] ; /* vettore di contatori
  delle frequenze delle lunghezze delle parole */
  char riga[MAXRIGA] ;
  int i, inizio, lunghezza ;
  FILE * f ;

  for(i=0; i<MAXPAROLA; i++)
    freq[i]=0 ;

  if(argc != 2)
  {
    fprintf(stderr, "ERRORE, serve un parametro con il nome del file\n");
    exit(1);
  }
  f = fopen(argv[1], "rt") ;
  if(f==NULL)
  {
    fprintf(stderr, "ERRORE, impossibile aprire il file %s\n", argv[1]);
    exit(1);
  }

  while( fgets( riga, MAXRIGA, f ) != NULL )
```

UNIX/Linux environment

Filters

Stefano Quer, Pietro Laface, and Stefano Scanzio

Dipartimento di Automatica e Informatica

Politecnico di Torino

skenz.it/os stefano.scanzio@polito.it

Filters

- In UNIX/Linux a **filter** is a command that
  - Takes its input from standard input
  - Process (filters) it according to some parameters and options
  - Produces its output on standard output
- Commands that are useful tools for text file processing

Filters

- Most popular filters
  - awk, cat, cut, compress, grep, head, perl, sed, sort, tail, tr, uniq, wc
- Some of these commands
  - Are quite complex
    - grep, sort
  - Are scripting languages
    - sed, awk
  - Often
    - Use **regular expressions**
    - Are used in **pipe** (by means of the **|** operator) with other commands

cut

- Selects and outputs sections from each line of files
  - Format
    - cut [options] file
  - Main options
    - --characters=LIST, -c LIST
      - Select only the characters at the positions in LIST
    - --fields=LIST, -f LIST
      - Selects the (comma separated) list of fields
      - Format
        - n (=n), -n (≤n), n- (≥n), n1-n2 (≥n1 && ≤n2)
        - Examples: 3, -3, 3-, 3-5

Examples

- --delimiter=DELIM, -d DELIM
  - Uses DELIM to separate fields rather than the default **TAB** delimiter

```
cut -f 1,3 file.txt

cut -f 1-3,5-6 -d " " foo.txt
```

Selects fields 1 and 3 of all file lines

Selects fields 1 to 3 (1, 2 and 3) and 5 to 6 of file foo.txt fields are delimited by space rather than by TAB

Exercise

- Report the UNIX command that does what indicated, possibly using redirections and pipes
  - Print how many characters are present in all files with extension ".txt" (without displaying additional information)

Solution

- Report the UNIX command that does what indicated, possibly using redirections and pipes
  - Print how many characters are present in all files with extension ".txt" (without displaying additional information)

-exec: applies wc (word count) to the content of all the files (\{}) matched by find

```
find . -name "*.txt" -exec wc \{} \; | cut -f 3 -d " "
```

Selects the third field wc under the hypotesis (typically false) that fields are separated by a single spaces
(wc → number of rows, strings, characters and file name)

tr

- Translate, compress, and/or delete characters from standard input, writing to standard output.
- Must be used by redirecting the output of other commands to its input
  - Format
    - tr [options] set_1 [set_2]
  - Main options
    - --delete, -d
      - Delete the characters in set_1
    - --squeeze-repeats, -s
      - replace each input sequence of a repeated character that is listed in set_1 with a single occurrence of that character
    - --complement, -c, -C
      - Uses the complement of set_1

Examples

- Inside set1 and set2
  - \num = character with ASCII code num
  - \n = newline
  - \\ = backslash

```
tr -d abcd < file.txt

cat file.txt | tr ab BA

echo ciao | tr ia IA

echo "a b    c"  | tr -s ' '
```

Outputs file.txt eliminating characters a, b, c, d

Outputs the lines of file.txt, in which 'a' is substituted with 'B', and 'b' is substituted with 'a'

Outputs cIAo

squeezes (compresses) the sequence of spaces to a single space. Outputs a b c

uniq

- Report or omit repeated lines of the input file
  - Format
    - uniq [options] [inFile] [outFile]
  - The file must be sorted
  - Without options eliminates the repeated lines

uniq

- Main options
  - --count, -c
    - prefix lines by the number of occurrences
  - --repeated, -d
    - only print duplicate lines, one for each group
  - --skip-fields=N, -f N
    - avoid comparing the first N fields
  - --ignore-case, -I
    - Case insensitive

Examples

```
uniq --count file.txt
uniq -c file.txt

uniq -d a.x
```

Eliminates duplicate lines of a sorted text file outputs the others and inserts the number of occurrences of the duplicated lines

Outputs the duplicated lines only

basename

- Eliminates the directories (path) from a pathname, and possibly its extension
  - Format
    - `basename pathname [extension]`

Examples

```
> basename /home/user1/current/file.txt
file.txt
> basename /home/user1/current/file.txt ".txt"
file
> basename /home/user1/current/file.txt .txt
file
> basename /home/user1/current/file.txt txt
file.
```

sort

- Sort the input file in alphabetic order
  - Format
    - `sort [options] [file]`
  - Main options
    - `--ignore-leading-blanks, -b`
      - Ignore the initial spaces
    - `--dictionary-order, -d`
      - Considers spaces and alphabetic characters only
    - `--ignore-case, -f`
      - Transforms lowercase characters in uppercase characters (Case insensitive)

sort

- **`--numeric-sort, -n`**
  - Sort in numeric order
- **`--reverse, -r`**
  - Sort in reverse order
- **`--key=c1,[,c2], -k c1[,c2]`**
  - Sort on the basis of the selected fields
- **`--merge, -m`**
  - Merges sorted files, no sort is performed without other options
- **`--output=f, -o=f`**
  - Writes its output on file f rather than on standard output

Examples

```
sort file.txt

cat file1.txt file2.txt | \
sort -r -k 1,3 -f
```

Sorts the lines of the file file.txt interpreting them as a sequence of ASCII characters

Concatenates files file1.txt and file2.txt, and sorts the rows of the two files in descending order using the fields 1, 2 and 3 and ignoring the difference between uppercase and lowercase letters

Exercise

- Report the UNIX command that does what indicated, possibly using redirections and pipes
  - Print all the files in the current directory by sorting the rows by increasing file size

Solution

- Report the UNIX command that does what indicated, possibly using redirections and pipes
  - Print all the files in the current directory by sorting the rows by increasing file size

Output of ls -al

```
total 28
drwxr-xr-x 1 scan scan 512 Nov 12 10:17 .
drwxr-xr-x 1 root root 512 Sep 26 16:08 ..
-rw------- 1 scan scan 1669 Oct 8 22:23 .bash_history
-rw-r--r-- 1 scan scan 220 Sep 26 16:08 .bash_logout
...
```

```
ls -la | sort -n -k 5
```

The lines referred to the directories ".", ".." and the word "total 20" remain

grep

- Global Regular Expression Print
  - Searches the input files for lines containing a match to the given pattern. If no files are specified, or if the file "-" is given, grep searches standard input. By default, grep prints the matching lines.
- Versions
  - grep
    - Standard version
  - egrep, fgrep, rgrep
    - egrep equivalent to ”grep –E”
    - Uses Extended RE for matching the pattern

grep

- Format
  - **`grep [options] pattern [file]`**
- Main options
  - **`--regexp=PATTERN, -e PATTERN`**
    - Specifies the search patterns
    - Allows you to specify multiple patterns
  - **`--line-number, -n`**
    - Outputs the matching line number
  - **`--recursive, -r, -R`**
    - Search recursively the sub-trees
  - **`--inverse-match, -v`**
    - Outputs only the lines that do not match

grep

- **`--ignore-case, -I`**
  - Case insensitive
- **`--after-context=N, -A N`**
  - Outputs N lines after each match line (in addition to the line in which the match was found)
- **`--before-context=N, -B N`**
  - Outputs N lines before each match line (in addition to the line in which the match was found)
- **`--with-filename, -H`**
  - Outputs the filename for each matching line

grep

Outputs the file lines that include the string "abc"

```
grep abc file.txt

grep -e "l." -e a file.txt

grep -H -A 4 abc file.txt
```

Outputs the file lines that include character 'l' followed by any other character, or include character 'a'

Outputs the file lines that include string "abc", and the next 4 lines, preceded by the filename

Exercise

- Report the UNIX command that does what indicated, possibly using redirections and pipes
  - Print all the files of the current directory ordering the lines by decreasing creation time

Solution

- Report the UNIX command that does what indicated, possibly using redirections and pipes
  - Print all the files of the current directory ordering the lines by decreasing creation time

Output of ls -al

```
total 28
drwxr-xr-x 1 scan scan 512 Nov 12 10:17 .
drwxr-xr-x 1 root root 512 Sep 26 16:08 ..
-rw------- 1 scan scan 1669 Oct 8 22:23 .bash_history
-rw-r--r-- 1 scan scan 220 Sep 26 16:08 .bash_logout
...
```

```
ls -la | \
  grep -v -e "total" -e "\.$" -e "\.\.$" | \
  sort -n -r -k 8
```

Deletes the directories ".", ".." and the "total" line

Exercise

- Report the UNIX command that does what indicated, possibly using redirections and pipes
  - Print all the rows of the files with extension ".txt" that contain a palindrome string
    - Of 3 characters (e.g., "aba")
    - Of 5 characters (e.g., "abcba")

Solution

- Report the UNIX command that does what indicated, possibly using redirections and pipes
  - Print all the rows of the files with extension ".txt" that contain a palindrome string
    - Of 3 characters (e.g., "aba")
    - Of 5 characters (e.g., "abcba")

Basic Reg Exp

```
grep -e "\(.\).\1" *.txt
grep -e "\(.\)\(.\).\2\1" *.txt
```

```
grep -extended-regexp -e "(.).\1" *.txt
grep -E -e "(.)(.).\2\1" *.txt
```

Extended Reg Exp

Exercise

- Report the UNIX command that does what indicated, possibly using redirections and pipes
  - In the directory "/home/foo" search the files with the name starting with the character "L" and extension "txt". In these files search the presence of the string "laib". Print the file name and the entire line where this string is matched.

Solution

- Report the UNIX command that does what indicated, possibly using redirections and pipes
  - In the directory "/home/foo" search the files with the name starting with the character "L" and extension "txt". In these files search the presence of the string "laib". Print the file name and the entire line where this string is matched.

```
find /home/foo -name "L*.txt" -exec \
  grep -H "laib" '{}' \;
```

Exercise

- Report the UNIX command that does what indicated, possibly using redirections and pipes
  - Find all the files with extension "txt" in the "/home" directory stored between the depth level 3 (included) and the depth level 5 (included) of the directory tree, and that are readable. For the selected files change the owner to "ugo".

Solution

- Report the UNIX command that does what indicated, possibly using redirections and pipes
  - Find all the files with extension "txt" in the "/home" directory stored between the depth level 3 (included) and the depth level 5 (included) of the directory tree, and that are readable. For the selected files change the owner to "ugo".

```
find /home -mindepth 3 -maxdepth 5 \
-name "*.txt" -readable \
-exec chown "ugo" '{}' \;
```

Exercise

- Report the UNIX command that does what indicated, possibly using redirections and pipes
  - For each file with extension "txt" located in the current directory, obtain the name and number of characters present in the file. The list must be sorted in inverse numerical order using the number of characters, and stored in a file named "stat.txt".

Solution

- Report the UNIX command that does what indicated, possibly using redirections and pipes
  - For each file with extension "txt" located in the current directory, obtain the name and number of characters present in the file. The list must be sorted in inverse numerical order using the number of characters, and stored in a file named "stat.txt".

or -k 1

```
find . -name "*.txt" \
-exec wc -c '{}' \; | sort -rn -k 1,1 > stat.txt
```

Exercise

- Report the UNIX command that does what indicated, possibly using redirections and pipes
  - A C application consists of main.c, f1.c, f2.c and main.h. Write a Makefile with two targets
    - The first is able to compile the application naming the executable file "myapp"
    - The second removes any temporary files and moves the executable to the "/use/bin" directory

Solution

- Report the UNIX command that does what indicated, possibly using redirections and pipes
  - A C application consists of main.c, f1.c, f2.c and main.h. Write a Makefile with two targets
    - The first is able to compile the application naming the executable file "myapp"
    - The second removes any temporary files and moves the executable to the "/use/bin" directory

<tab>

```
compile: main.c f1.c f2.c
    gcc -o myapp main.c f1.c f2.c

install:
    rm *.tmp
    cp myapp /user/bin
```