Algorithms and Programming

Laboratory number 06

--------------------

Exercise 01 (A, B, C)

Extension of Exercise 01, Laboratory 03

---------------------------------------

A text of unknown length, but whose rows are limited to 1000

characters, is stored in a file.

Write a C application able to evaluate the absolute frequency of all

strings appearing in the file.

More specifically the program has to:

\* Receive on the command line two file names (input and output

files).

\* Read the input file and store all words in a list.

Each node of the list must contain the string and its absolute

frequency within the input file.

Capital and small letters has to be considered as equivalent

(i.e., "WORD", "Word", and "word" are the same string).

\* Store the list into the output file.

The list may be manipulated using:

1. Head insertions.

In this way the record order within the file will be inverted into

the list.

2. Tail insertions (using a list with a tail pointer with or without a

circular implementation).

In this way the record order within the file will be maintained into

the list.

3. Ordered insertion (into an ordered list).

In this way the record order within the file will be reorder into

the list.

At choice implement version A, B, or C or all of them.

Example

-------

This is a correct input file

Mi illumino di immenso

Illumino di immenso

Di immenso

IMMENSO

The output file has to be the following (order depends on the type of

list manipulated):

immenso 4

di 3

illumino 2

Mi 1

Exercise 02

-----------

A file contains data on a set of employees.

For each employee there is a row of the file, including:

\* Last and first name (a single C string, maximum 50 characters,

e.g., Smith\_John).

\* Personal identification (exactly 16 characters).

\* Data of hiring (format dd.mm.yyy, e.g, 30.05.1995).

\* Salary (integer value, in euro).

Fields are space-separated.

Employees do not appear in any specific order.

A C program receives 3 parameters on the command line:

\* Input file name (the format is the previously defined one).

\* Last and first name (single string, e.g., Clinton\_Bill).

\* A string made of only + and - characters (e.g., +++---+-+).

The program has to:

\* Read the file.

\* Store its content in a LIFO-logic list, but with two pointers for

each element one pointing ahead and one pointing behind the element

itself as:

pHead --> ----- --> ----- --> ----- --> ----- -X

|###| |###| |###| |###|

X- ----- <-- ----- <-- ----- <-- -----

\* Find in the list the employee whose name is passed on the command

line as a second parameter.

\* Move along the list in the

- right direction for each '+' character

- left direction for each '-' character

in the third string parameter.

For each visited node of the list (included the first one) the

program has to print out (on standard output) all data of the

employee (with the same format those data appear in the original

input file).

If the end of the list is reached (either side) the program has to

print-out the same element data repeatedly.

Example

-------

Let the command line parameters be the following

file.txt Giallo\_Antonio ---+

and the file be the following

Rossi\_Alberto AAABBBCCDEEFGGGH 03.12.1998 1845

Giallo\_Antonio AAABBBCCDEEFGGGH 13.11.2007 1140

Verdi\_Federica AAABBBCCDEEFGGGH 25.09.1989 2157

Bianchi\_Elena AAABBBCCDEEFGGGH 15.02.2004 1345

The file has to be stored in the LIFO structure as:

Bianchi -> Verdi -> Giallo -> Rossi

Then, the program has to:

- find Giallo\_Antonio in the list and print its data

- move let on Verdi and print its data (first -)

- move left again and print Bianchi (second -)

- do not move and print Bianchi again (third -)

- move right and print Verdi (first +):

Giallo\_Antonio AAABBBCCDEEFGGGH 13.11.2007 1140

Verdi\_Federica AAABBBCCDEEFGGGH 25.09.1989 2157

Bianchi\_Elena AAABBBCCDEEFGGGH 15.02.2004 1345

Bianchi\_Elena AAABBBCCDEEFGGGH 15.02.2004 1345

Verdi\_Federica AAABBBCCDEEFGGGH 25.09.1989 2157