

Relazione Progetto S.O.

Applicazione Find

Descrizione

L'obiettivo del progetto è quello di sviluppare l'applicazione `find` che consente di individuare il numero di occorrenze di un insieme di stringhe all'interno di un gruppo di file. Per determinare il numero di occorrenze (e la loro posizione) si è utilizzato l'Algoritmo di [Knuth-Morris e Pratt](#). Ad ogni esecuzione l'applicazione può produrre in output la lista dei file analizzati con le occorrenze della stringa nel testo insieme alle informazioni riguardanti la posizioni della stesse. Le stesse informazioni prodotte in output potranno essere salvate su di un file esterno.

Funzionalità e Uso

L'applicazione `find` può essere usata sia per generare dei report delle analisi che per estrarre informazioni da report generati in esecuzioni precedenti.

Generazione di report :

Per generare un nuovo report occorre indicare le parole da ricercare e le directory o file dove eseguire la ricerca. I file e le directory da analizzare verranno elencate in un file, da passare come input al programma, con la seguente struttura:

```
<path1> [r]\r\n
<path2> [r]\r\n
...
<pathk> [r]\r\n
\r\n
```

Ogni `<pathi>` potrà essere un path assoluto o relativo ad un file o directory. Nel secondo caso, il parametro `r`, opzionale, indicherà se occorra analizzare ricorsivamente le sottodirectory. Se non presente solo i file regolari nella directory verranno analizzati.

L'elenco delle parole da ricercare, invece, verrà passato al programma attraverso un file della forma:

```
word1\r\n
word2\r\n
...
```

L'esecuzione del programma avverrà quindi con i seguenti parametri:

```
find (--words|-w) <wordfile> (--input|-i) <inputfile>
```

In questo caso il report verrà stampato a termine dell'esecuzione.

Per salvare il report su un file particolare occorrerà aggiungere il parametro

```
--output|-o <outputfile>
```

Durante la fase di analisi sarà possibile ignorare i file con specifiche estensioni, aggiungendo il parametro:

```
--exclude|-e <ext>
```

Infine, per visionare il processo di analisi si potrà aggiungere il parametro

```
--verbose|-v
```

In questo caso per ogni file (o directory) analizzata occorrerà stampare messaggi che indicano l'inizio dell'elaborazione, il termine ed il tempo necessario all'analisi :

```
Inizio elaborazione parola: aria
```

```
Inizio elaborazione directory: /home/loreti/Documents
```

```
Inizio elaborazione file: /home/loreti/Documents/marzo1821.txt
```

```
Fine elaborazione file: /home/loreti/Documents/marzo1821.txt (0.2 sec)
```

```
Inizio elaborazione file: /home/loreti/Documents/5maggio.txt
```

```
Fine elaborazione file: /home/loreti/Documents/5maggio.txt (0.1 sec)
```

```
Fine elaborazione parola: aria
```

Analisi dei report

Una volta generato il file di report, il programma `find` può essere usato per recuperare le informazioni salvate. Potremmo:

* Stampare la lista dei file dove occorre almeno `<n>` volte la parola `<word>`:

```
find --report|-r <reportfile> --show <word> <n>
```

Se `<n>` viene omissso, si utilizza il valore `1`.

* Stampare tutte le posizioni dove la parola `<word>` occorre nel file `<file>`:

```
find --report|-r <reportfile> --show <word> --file <file>
```

Se `<word>` non occorre in `<file>`, viene stampato a video un messaggio opportuno.

Librerie utilizzate

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

Le funzioni dichiarate in `stdio.h` possono generalmente essere divise in due categorie: le funzioni per la manipolazione di file e quelle per la manipolazione dell'input/output.

`#include <stdlib.h>`

dichiara funzioni e costanti di utilità generale: allocazione della memoria, controllo dei processi, e altre funzioni generali comprendenti anche i tipi di dato

`#include <string.h>`

contiene definizioni di macro, costanti e dichiarazioni di funzioni e tipi usati non solo nella manipolazione delle stringhe ma anche nella manipolazione della memoria

`#include <getopt.h>`

getopt è una funzione di libreria C utilizzata per analizzare le opzioni della riga di comando dello stile Unix / POSIX. Fa parte delle specifiche POSIX ed è universale per i sistemi simili a Unix
E' anche il nome di un programma UNIX per l'analisi degli argomenti della riga di comando negli script di shell.

`#include <time.h>`

time.h è l'header file della libreria standard del C che fornisce un accesso standardizzato alle funzioni di acquisizione e manipolazione del tempo.

Struttura del programma

main.c

Classe principale dove grazie alla libreria getopt riusciamo a creare un set di opzioni per l'input così da garantire le varie funzionalità.

kmp.c

Algoritmo che effettua la ricerca di una parola in una riga di un file e se la trova, aggiunge una occorrenza ad una lista

find.c

esegue la ricerca di una parola all'interno di tutti i file ottenuti dalla lettura del file contenente i path

exec.c

Si occupa di tutte le funzionalità che richiedono la ricerca delle parole e l'analisi del report

report.c

Si occupa di scrivere un file di report o di stampare il report su console

record.c

Si occupa delle strutture dati utilizzate per il report

list.c

Si occupa della definizione di strutture dati lista per memorizzare elementi al suo interno

verbose.c

Si occupa di stampare le informazioni sul processo di analisi con le relative tempistiche

Strutture dati

struttura per memorizzare una singola coppia linea-carattere

```
typedef struct recordDataLine {  
    int line;  
    int lineChar;  
} recordDataLine;
```

struttura per memorizzare un singolo file path, il numero di occorrenze della parola in quel file, e una lista di coppie linea-carattere del testo in cui è possibile trovare la parola

```
typedef struct recordData {  
    char *file;  
    int occurrences;  
    list *lines;  
} recordData;
```

un record contiene una parola, il numero di occorrenze totali e una lista di file in cui è possibile trovare la parola

```
typedef struct record {  
    char *word;  
    int total;  
    list *recordData;  
} record;
```

un singolo elemento della lista

```
typedef struct listNode {
```

 usato puntatore a void per memorizzare dati di qualsiasi tipo

```
    void *data;
```

```
    struct listNode *next;
```

```
} listNode;
```

lista concatenata con guardie (ha cioè riferimento a primo e ultimo elemento)

```
typedef struct list {
```

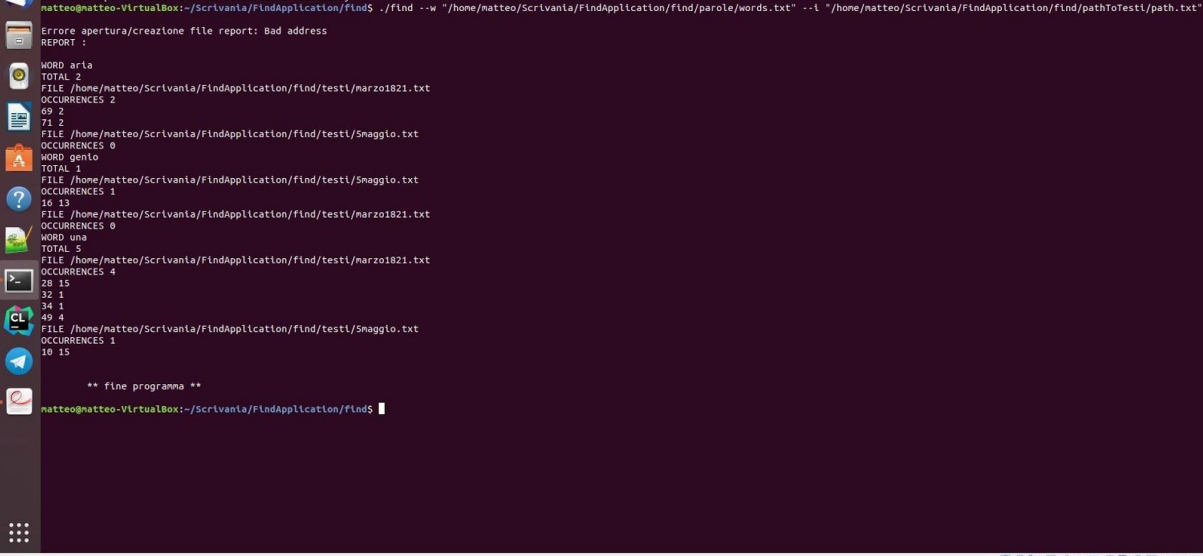
```
    listNode *head;
```

```
    listNode *tail;
```

```
} list;
```

Esempi delle funzionalità

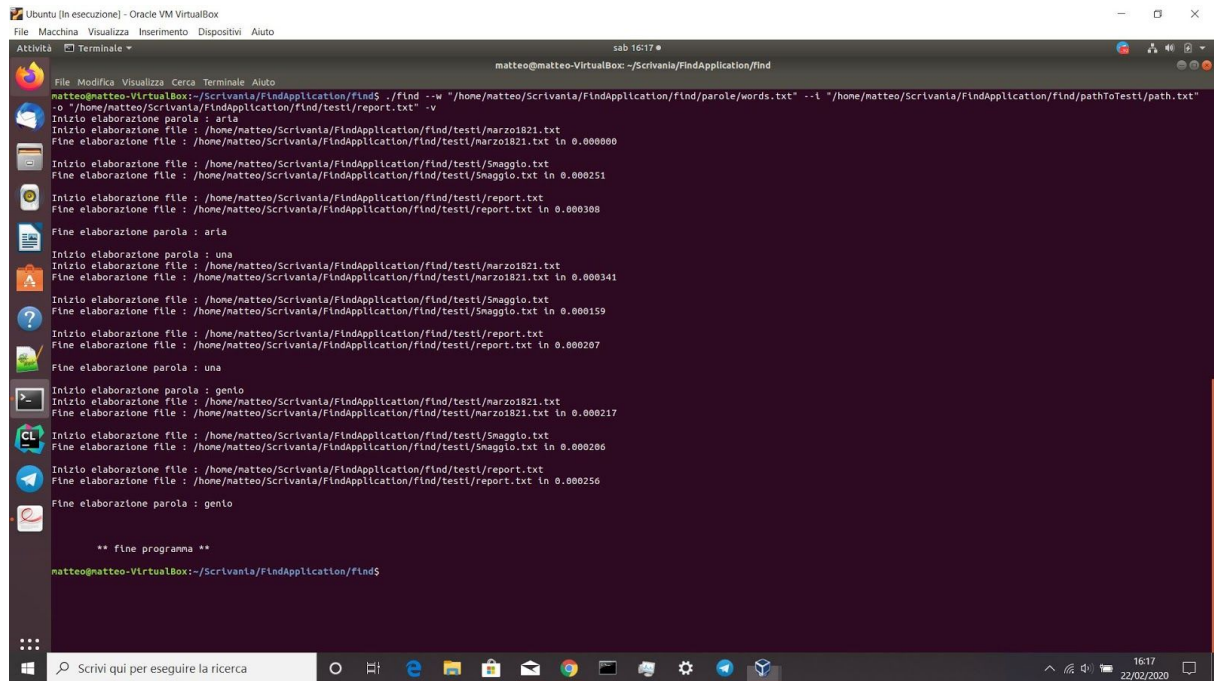
Ricerca parole + report su console



```
matteo@matteo-VirtualBox:~/Scrivania/FindApplication/Find$ ./find -w "/home/matteo/Scrivania/FindApplication/Find/parole/words.txt" -l "/home/matteo/Scrivania/FindApplication/Find/pathToTestI/path.txt"
Errore apertura/creazione file report: Bad address
REPORT :
WORD aria
TOTAL 2
FILE /home/matteo/Scrivania/FindApplication/Find/testI/narzo1821.txt
OCCURRENCES 2
69 2
71 2
FILE /home/matteo/Scrivania/FindApplication/Find/testI/Smaggio.txt
OCCURRENCES 0
WORD genio
TOTAL 1
FILE /home/matteo/Scrivania/FindApplication/Find/testI/Smaggio.txt
OCCURRENCES 1
10 13
FILE /home/matteo/Scrivania/FindApplication/Find/testI/narzo1821.txt
OCCURRENCES 0
WORD una
TOTAL 5
FILE /home/matteo/Scrivania/FindApplication/Find/testI/narzo1821.txt
OCCURRENCES 4
28 15
32 1
34 1
49 4
FILE /home/matteo/Scrivania/FindApplication/Find/testI/Smaggio.txt
OCCURRENCES 1
10 15

** fine programma **
matteo@matteo-VirtualBox:~/Scrivania/FindApplication/Find$
```

Ricerca parole report su file specifico + verbose

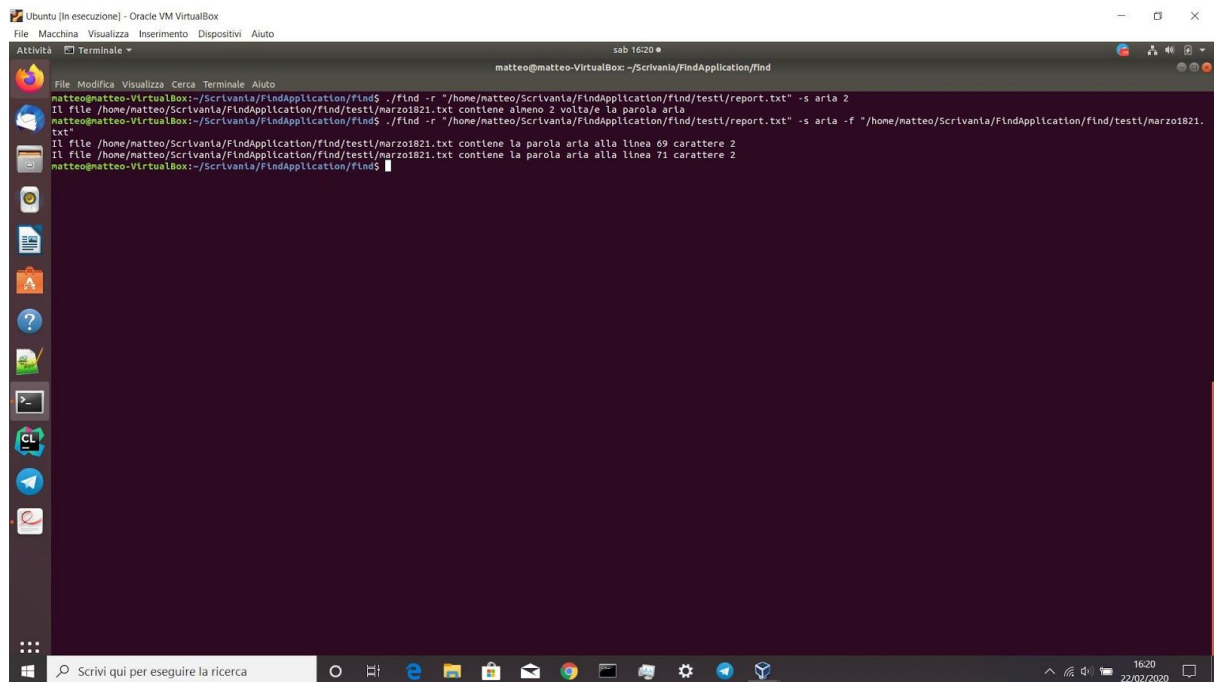


```
matteo@matteo-VirtualBox: ~/Scrivania/FindApplication/Find$ ./find --w "/home/matteo/Scrivania/FindApplication/Find/parole/words.txt" --l "/home/matteo/Scrivania/FindApplication/Find/pathToTest/path.txt" -o "/home/matteo/Scrivania/FindApplication/Find/test/report.txt" -v
Inizio elaborazione parola : aria
Fine elaborazione file : /home/matteo/Scrivania/FindApplication/Find/test/narzo1821.txt in 0.000000
Inizio elaborazione file : /home/matteo/Scrivania/FindApplication/Find/test/Snaggio.txt
Fine elaborazione file : /home/matteo/Scrivania/FindApplication/Find/test/Snaggio.txt in 0.000251
Inizio elaborazione file : /home/matteo/Scrivania/FindApplication/Find/test/report.txt
Fine elaborazione file : /home/matteo/Scrivania/FindApplication/Find/test/report.txt in 0.000308
Fine elaborazione parola : aria
Inizio elaborazione parola : una
Inizio elaborazione file : /home/matteo/Scrivania/FindApplication/Find/test/narzo1821.txt
Fine elaborazione file : /home/matteo/Scrivania/FindApplication/Find/test/narzo1821.txt in 0.000341
Inizio elaborazione file : /home/matteo/Scrivania/FindApplication/Find/test/Snaggio.txt
Fine elaborazione file : /home/matteo/Scrivania/FindApplication/Find/test/Snaggio.txt in 0.000159
Inizio elaborazione file : /home/matteo/Scrivania/FindApplication/Find/test/report.txt
Fine elaborazione file : /home/matteo/Scrivania/FindApplication/Find/test/report.txt in 0.000207
Fine elaborazione parola : una
Inizio elaborazione parola : genio
Inizio elaborazione file : /home/matteo/Scrivania/FindApplication/Find/test/narzo1821.txt
Fine elaborazione file : /home/matteo/Scrivania/FindApplication/Find/test/narzo1821.txt in 0.000217
Inizio elaborazione file : /home/matteo/Scrivania/FindApplication/Find/test/Snaggio.txt
Fine elaborazione file : /home/matteo/Scrivania/FindApplication/Find/test/Snaggio.txt in 0.000206
Inizio elaborazione file : /home/matteo/Scrivania/FindApplication/Find/test/report.txt
Fine elaborazione file : /home/matteo/Scrivania/FindApplication/Find/test/report.txt in 0.000256
Fine elaborazione parola : genio

** fine programma **

matteo@matteo-VirtualBox:~/Scrivania/FindApplication/Find$
```

Analisi report



```
matteo@matteo-VirtualBox:~/Scrivania/FindApplication/Find$ ./find -r "/home/matteo/Scrivania/FindApplication/Find/test/report.txt" -s aria 2
Il file /home/matteo/Scrivania/FindApplication/Find/test/narzo1821.txt contiene almeno 2 volta/e la parola aria
matteo@matteo-VirtualBox:~/Scrivania/FindApplication/Find$ ./find -r "/home/matteo/Scrivania/FindApplication/Find/test/report.txt" -s aria -f "/home/matteo/Scrivania/FindApplication/Find/test/narzo1821.txt"
Il file /home/matteo/Scrivania/FindApplication/Find/test/narzo1821.txt contiene la parola aria alla linea 69 carattere 2
matteo@matteo-VirtualBox:~/Scrivania/FindApplication/Find$
```

