

Corso di Sistemi Operativi e Reti, corso di Sistemi Operativi - 23 Luglio 2013 -

1. **PER GLI STUDENTI DI SISTEMI OPERATIVI E RETI:** è necessario sostenere e consegnare **entrambi** gli esercizi. Sarà attribuito un **unico** voto su tutta la prova.
2. **PER GLI STUDENTI DI SISTEMI OPERATIVI:** si può sostenere solo uno dei due esercizi se si è già superata in un appello precedente la corrispondente prova. Il tempo a disposizione in questo caso è di **2 ORE**.

Troverete sul vostro Desktop una cartella chiamata "CognomeNomeMatricola" che contiene la traccia dell'elaborato ed eventuali altri file utili per lo svolgimento della prova. Ai fini del superamento della prova è indispensabile rinominare tale cartella sostituendo "Cognome" "Nome" e "Matricola" con i vostri dati personali. Ad esempio, uno studente che si chiama Alex Britti ed ha matricola 66052 dovrà rinominare la cartella "CognomeNomeMatricola" in "BrittiAlex66052".

Non saranno presi in considerazione file non chiaramente riconducibili al proprio autore. E' possibile caricare qualsiasi tipo di materiale didattico sul desktop nei primi 5 minuti della prova.

Si consiglia di salvare SPESSO il proprio lavoro.

ESERCIZIO 1 (Linguaggi di scripting)

Si vuole realizzare uno script perl **antispammer.pl** in grado di filtrare i messaggi indesiderati in una *mailbox*. Nello specifico, si supponga data la struttura di file/directory riportata nella Fig. 1, dove:

- la cartella *mailbox* contiene un insieme di messaggi email (il cui formato è descritto in Fig. 2)
- le cartelle *SPAM* e *FORSESPAM* sono originariamente vuote
- un file *regole.txt* (il cui formato è descritto in Fig3)

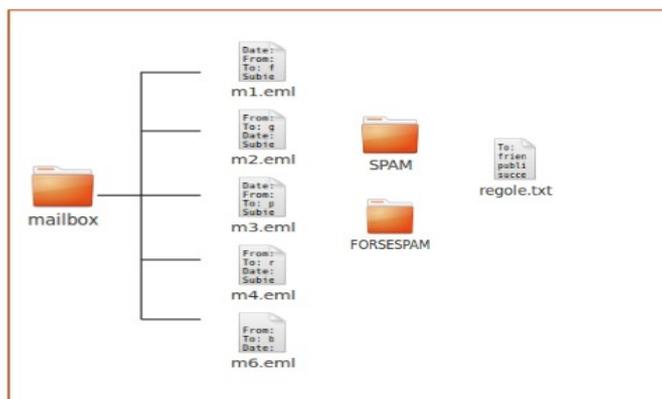


Fig1: Il contenuto della cartella dell'esame.

Un file di testo *.eml* contenuto nella directory *mailbox* rappresenta il contenuto di una email nel formato riportato di seguito in Fig. 2, dove sono evidenziate in rosso le parti rilevanti per l'antispammer (il valore del campo *From*, *Subject* ed il *contenuto* della mail).

Date: una data
From: indirizzo email mittente
To: indirizzo email destinatario
Subject: oggetto della mail

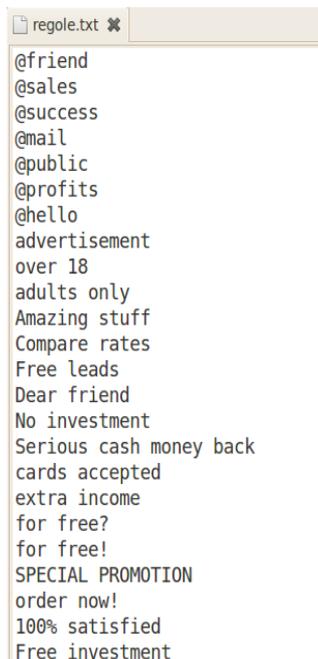
Contenuto della mail.

```
Date: Tue, 9 Jul 2013 10:03:30 +0100
From: Ernest@yahoo-lavoro.com
To: friend-list@systems.it
Subject: Aumentate il vostro guadagno per 2100 euro al mese
```

```
Io sono HR manager di una grande compagnia internazionale.
La nostra azienda È alla ricerca di collaboratori ....
```

Fig2: Il formato di una email e un esempio.

Il file delle *regole.txt* contiene un elenco (separato su righe) di parole o insieme di parole vietate eventualmente rintracciabili nei campi *From*, *Subject* oppure nel *contenuto* di una email e che, se presenti nella mail, consentono di scartarle.



```
regole.txt *
@friend
@sales
@success
@mail
@public
@profits
@hello
advertisement
over 18
adults only
Amazing stuff
Compare rates
Free leads
Dear friend
No investment
Serious cash money back
cards accepted
extra income
for free?
for free!
SPECIAL PROMOTION
order now!
100% satisfied
Free investment
```

Fig3: Il file delle regole.

In particolare:

1. una mail (file *.eml* della *mailbox*) che presenta nel campo *From* oppure nel campo *Subject* delle parole vietate è da considerare come *possibile spam*
2. una mail (file *.eml* della *mailbox*) che presenta parole vietate nel campo *From* oppure *Subject* ed anche nel *contenuto* è da considerare come *spam*.

Lo script deve:

1. accedere ai file delle email presenti nella directory *mailbox* (file *.eml*)
2. verificare per ciascun file *.eml* se contiene delle parole vietate nei relativi campi rilevanti e:
 1. spostare il file in esame nella directory *FORSESPAM* se contiene parole vietate nel campo *From* oppure *Subject* (e non nel *contenuto*)
 2. *spostare il file in esame nella directory SPAM se contiene parole vietate nel campo From oppure Subject e anche nel contenuto*

ESERCIZIO 2 (Programmazione multithread)

Sia dato un tipo di classe che modella una matrice quadrata di interi di dimensione $(N^2) \times (N^2)$. Gli interi ammissibili come valori all'interno della matrice sono quelli che vanno da **1** a **N**.

Si scriva un metodo di questa classe, facente uso di parallelismo per migliorare le prestazioni, che, avendo a disposizione un processore con **N** core, ritorni *true* se i valori contenuti nella matrice rispettano le classiche regole del gioco del Sudoku per una tavola completamente risolta, e cioè:

1. Nella matrice non può comparire alcun valore fuori dall'intervallo $1..N$;
2. Nessun numero può comparire più di una volta sulla stessa riga;
3. Nessun numero può comparire più di una volta sulla stessa colonna;
4. (Bonus) Supponendo di dividere la matrice in oggetto in sottoblocchi quadrati di dimensione $N \times N$, nessun numero può comparire più di una volta nello stesso sottoblocco (si veda figura di esempio).

Il metodo ritorna *false* altrimenti.

5. (Bonus) Si sviluppi il metodo in maniera tale che sia possibile l'*early exit*: il metodo deve cioè terminare immediatamente ritornando *false* se viene trovato che uno tra i punti da 1 a 4 viene violato.

E' consentito (ma non obbligatorio), fare uso delle funzioni disponibili nel JDK di Java 6, e di qualsiasi altra funzione predefinita a disposizione. Non saranno valutate versioni sequenziali del programma. Il lavoro può essere sviluppato o in Java o in C++

SI NOTI CHE, PENA L'ESCLUSIONE DALLA PROVA, IL METODO DEVE POTER ELABORARE TAVOLE SUDOKU DI DIMENSIONE GENERALIZZATA, NON SOLO LE CLASSICHE TAVOLE 9X9.

Figura di esempio:

7	15	14	10	3	16	4	12	9	13	5	2	11	8	6	1
6	11	13	1	5	14	9	2	16	4	15	8	10	7	3	12
16	12	5	8	11	13	6	15	3	1	10	7	9	4	2	14
3	4	2	9	8	7	10	1	12	6	11	14	13	5	15	16
2	7	10	15	12	3	5	6	4	11	8	16	14	1	13	9
9	1	12	4	2	11	14	16	7	5	6	13	8	3	10	15
5	14	3	6	9	1	13	8	10	15	2	12	16	11	4	7
13	16	8	11	10	4	15	7	1	14	9	3	6	12	5	2
15	13	11	12	16	6	7	4	14	9	3	5	2	10	1	8
1	10	6	7	14	5	2	13	11	8	16	15	3	9	12	4
14	5	9	3	15	8	12	11	2	10	1	4	7	6	16	13
8	2	4	16	1	10	3	9	13	7	12	6	5	15	14	11
10	9	7	5	6	12	1	14	15	2	13	11	4	16	8	3
11	3	16	14	13	9	8	10	6	12	4	1	15	2	7	5
12	6	15	13	4	2	16	5	8	3	7	9	1	14	11	10
4	8	1	2	7	15	11	3	5	16	14	10	12	13	9	6