

Laboratorio 5 — Generazione di un Insieme di Password

Assignment — password_set.c

Università di Catania — Dipartimento di Matematica e Informatica

Corso di Laurea in Informatica

Programmazione 1 e Laboratorio (Canale O-Z)

Descrizione del Programma

Si scriva un programma in linguaggio C, chiamato `password_set.c`, che generi un insieme di password casuali basato su vincoli di robustezza locale (per singola password) e coerenza globale (per l'intero insieme), memorizzando e gestendo i dati all'interno di una matrice bidimensionale di caratteri.

Il programma legge da standard input la lunghezza delle password L e il numero di password N da generare. Inizializza il generatore di numeri casuali utilizzando l'ora di sistema corrente tramite la funzione `time(NULL)` (dalla libreria `<time.h>`) per garantire che ad ogni esecuzione vengano prodotte password differenti.

Il programma lavora su un array bidimensionale:

```
char passwords[MAX_N][MAX_L+1]
```

dove ogni password generata deve rispettare determinati criteri per essere considerata valida.

Requisiti di Implementazione

Il programma deve essere strutturato nei seguenti punti:

PUNTO A: Acquisizione Parametri: Leggere da standard input la lunghezza L e il numero N di password da generare.

- Si stampi il prompt: `Inserisci la lunghezza L e il numero N delle password da generare: .`
- Validare che $6 \leq L \leq 50$ e $1 \leq N \leq 100$. In caso di valori fuori range, stampare il messaggio di errore `Errore: parametri fuori range.` e uscire con exit status 1.

PUNTO B: Vincoli Locali (Funzione): Definire la funzione `int check_local_constraints(char *pw)`. Ciascuna password generata deve:

- Avere lunghezza esattamente L .
- Contenere almeno una lettera minuscola (a-z), una maiuscola (A-Z), una cifra (0-9) e un carattere speciale (tra i simboli !@#\$%&*?).
- Non contenere tre caratteri identici consecutivi (ad esempio aaa o 999).
- Non contenere sequenze consecutive incrementali di 3 caratteri in ordine ASCII crescente (ad esempio abc o 123, ovvero caratteri tali che $pw[i] == pw[i - 1] + 1$ e $pw[i - 1] == pw[i - 2] + 1$).

PUNTO C: Vincoli Globali e Somiglianza (Funzioni):

- Definire la funzione `int is_duplicate(const char pw[] [MAX_L+1], int N, int idx)` per verificare se la password all'indice `idx` compare nelle righe precedenti della matrice.
- Definire la funzione `int hamming_distance(const char *a, const char *b)` per calcolare la distanza di Hamming (numero di posizioni in cui i caratteri differiscono). Due password sono considerate "troppo simili" se hanno una distanza di Hamming strettamente inferiore a $L/2$.

PUNTO D: Generazione e Report:

- Inizializzare la sorgente casuale tramite `srand((unsigned int)time(NULL))`.
- Generare le N password inserendole nella matrice, assicurando che ciascuna soddisfi i vincoli locali e che non vi siano duplicati globali (in caso contrario, rigenerare la password).
- Stampare l'elenco finale delle password generate una per riga, preceduto dall'etichetta **Password generate:**.
- Calcolare la distanza di Hamming per ogni coppia di password distinte e contare quante coppie risultano "troppo simili".
- Stampare il riepilogo finale riportando il numero totale di password generate e il numero di coppie troppo simili.
- Terminare con `exit status 0`.

Output di Controllo (Esempi di Esecuzione)

Caso 1 — Generazione valida di un insieme di password

Inserendo i dati:

10 5

Il programma deve stampare a schermo un output di questo tipo (le password reali cambieranno ad ogni esecuzione):

Inserisci la lunghezza L e il numero N delle password da generare:

Password generate:

C&hw0oUuR#

%9cu&!3TE?

A8P19F\$W2i

KbcB8?Re@9

E@H4NgKm8I

Password generate: 5

Coppie troppo simili: 0

(Exit code atteso: 0)

Caso 2 — Inserimento di parametri fuori range

Inserendo i dati:

4 5

Il programma deve stampare a schermo:

Inserisci la lunghezza L e il numero N delle password da generare: Errore:

↪ parametri fuori range.

(Exit code atteso: 1)