RELAZIONE PROGETTO “GESTIONE DATABASE ANAGRAFICO” IN C.

Francesco Avantaggiato.

Il progetto è stato realizzato e testato con l’utilizzo dell’IDE Dev-C++ su computer Windows.

- **Funzionamento generale:**

Il programma gestisce gli accessi ad un database (in formato file txt) di dati anagrafici (nome, cognome, data di nascita, comune di residenza, numero di telefono) di persone.
L’accesso al database avviene tramite username e password (è presente file contenente username e password per ogni utente, da utilizzare come riscontro per consentire o negare il login).
Nel caso del login da parte dell’administrator, lo stesso potrà inserire nuovi dati anagrafici, modificare/eliminare i dati presenti, aggiungere nuovi utenti(nuove credenziali), assegnare password attraverso un generatore di password casuali o manualmente, visualizzare il database, visualizzare un riepilogo delle credenziali (username e password per utenti ed admin), modificare/eliminare credenziali.
Nel caso l’accesso sia da parte di un utente, esso potrà solo visionare il database ed utilizzare una funzione di ricerca tramite keyword.

Per il corretto funzionamento del programma, lo stesso deve essere salvato nella stessa cartella dei file “database.txt” e “password.txt”, entrambi allegati alla e-mail.

- **Il main()**

Il main contiene la prima richiesta in output per l’utente, nel particolare per effettuare o meno il login al sistema. In caso di accesso, viene richiamata la funzione login() descritta in seguito. Dopo l’accesso si procede all’elencazione delle funzioni disponibile per quel profilo (admin o utente). Una volta che in input si ha la scelta, si effettua una selezione per chiamare le funzioni corrispondenti. Alla fine dell’utilizzo di una singola funzione, si chiede la volontà o meno di effettuare altre operazioni, concludendo con la chiusura del programma.

- **La funzione login()**

La funzione login necessita del file di riscontro “password.txt” per verificare la correttezza dell’username e della password inseriti in input.
Tramite la funzione fgets() viene letta una riga per volta fino al carattere “\n” (a capo) o fino al carattere terminatore del file (EOF).
Per effettuare il confronto dei dati, viene utilizzata prima la funzione strtok(), contenuta nell’header file “string.h”, la quale consente di suddividere la stringa in una serie di token, con l’utilizzo del carattere delimitatore (delim, dichiarato nei #define all’inizio del codice). In questo caso il delimitatore è il tab “ “.
In seguito viene confrontata con strcmp() l’eventuale uguaglianza o meno dei dati, restituendo un valore pari a 0 (zero) nel caso in cui i due dati siano uguali.
L’utilizzo nel codice di login() delle funzioni strtok e strcmp avviene due volte, rispettivamente per dividere e confrontare l’username, la password; successivamente tramite un controllo della presenza o meno del carattere “a” si identifica se l’accesso viene effettuato da utente o da admin (“u” per utente, “a” per admin).

La funzione login() ritornerà al main:

- il valore 0 (zero) se l'accesso è effettuato come admin, mostrando in seguito le relative possibilità di scelta dell'amministratore;
- il valore 1 se l'accesso è effettuato come utente, mostrando le possibilità di scelta dell'utente;

In caso di credenziali errate, verrà mostrato in output il relativo avviso.

- **La funzione pwdgenerator(), generatore di password casuali**

Il generatore di password casuali si avvale dell'utilizzo fondamentale di srand().
La stessa inizializza la funzione per la generazione dei numeri casuali insieme alla funzione time() (in header file "time.h"), che restituisce il tempo attuale in secondi dall'epoca (01/01/1970).
Per una password alfanumerica con caratteri speciali prendiamo in considerazione i caratteri ASCII tra il 33 ed il 126.
Per restringere questo campo numerico si effettua l'operazione con il modulo: "33+rand() % 93".
Tramite questo codice, vengono generate delle lettere singole per volta, le quali, unite, vengono salvate all'interno di una stringa con l'aggiunta del carattere terminatore "\0".
In conclusione della funzione pwdgenerator, è presente getchar(): ciò consente di acquisire un carattere in input, in questo caso un "enter" per andare avanti nel programma.

- **La funzione nuovecredenziali()**

La funzione lavora sul file "password.txt" ed è un'esclusiva dell'amministratore.
Viene richiesto l'inserimento dei nuovi dati per l'accesso e contestualmente, per quanto riguarda la password, si richiede se la stessa deve essere generata casualmente da pwdgenerator() oppure inserita manualmente. Inoltre si richiede se il nuovo utente è admin o utente ("a" o "u").
In conclusione vengono scritti su file i dati inseriti attraverso fprintf() ed sprintf().

- **La funzione crea()**

"crea()" consente di inserire nuovi dati anagrafici sul file "database.txt". Anche questa è utilizzata solo dall'admin.
Vengono richiesti in input tutti i nuovi dati (nome, cognome, data di nascita, comune di residenza, numero di telefono) e successivamente scritti su file con la funzione fprintf().
Nel particolare, i dati vengono scritti su file aperto in modalità "a+" (append) ovvero la scrittura dalla fine del file in poi, senza alcuna anomalia sull'eventuale possibilità di sovrascrivere i dati.

- **La funzione modifica()**

La funzione consente al solo amministratore di modificare o eliminare i dati anagrafici presenti sul file "database.txt". Viene aperto il file "database.txt" e creato ed aperto un file "temporaneo.txt".
Viene richiesto in input se modificare o eliminare i dati.
Nel caso in cui l'amministratore scelga di modificare i dati, viene chiesto quale dato anagrafico deve essere modificato (tramite scelta con valore intero, variabile "r") e viene chiamata la funzione "cercamodifica()".
Successivamente si richiede il nuovo dato e si ricopiano tutti i dati sul file temporaneo tranne la riga corrispondente al dato immesso.
Attraverso l'utilizzo di un vettore di puntatori (stringhe) "*m[5]" (5 campi anagrafici) e con strtok() viene suddivisa la stringa in token assegnando gli stessi al vettore "*m[5]".
Infine nella posizione "r-1" viene copiato il nuovo dato ed il tutto scritto sul file "temporaneo.txt".
Una volta chiusi i file con la funzione fclose(), si procede alla rimozione del vecchio file "database.txt" con la funzione remove() ed alla rinomina del file "temporaneo.txt" in "database.txt" con la funzione rename(). Attraverso due condizioni if, si controlla l'effettivo

funzionamento di remove() e rename(), con relativo messaggio di errore in caso di operazioni fallite, presumibilmente a causa di uno dei due file rimasti aperti precedentemente.

- **La funzione visualizzadb()**

La funzione può essere utilizzata sia dall'amministratore che dall'utente. Viene richiesta in output la volontà o meno di visionare i dati del database. In caso positivo, vengono mostrati gli stessi in output tramite un ciclo while con fgets(), stampando le singole righe, una per volta.
In conclusione viene chiuso il file "database.txt".

- **La funzione cerca()**

La funzione cerca() è in utilizzo all'utente e all'amministratore, essa consente di ricercare con keyword all'interno del database, mostrando in output la stringa corrispondente.
Il suo funzionamento di basa sulla richiesta in input della keyword, successivamente attraverso un ciclo con la funzione fgets(), vengono fatti scorrere linea per linea i dati presenti nel file. Con l'utilizzo della funzione strstr() (appartenente anch'essa all'header file string.h), si efffettua la ricerca della stringa data in input (la keyword) all'interno della riga fornita dal ciclo; nel caso in cui la ricerca dia esito positivo, viene stampato in output tramite printf la stringa corrispondente.

- **La funzione cercamodifica()**

Questa funzione risulta essere identica nel funzionamento alla funzione cerca(). In questo caso però, cercamodifica() viene chiamata internamente alla funzione modifica(), dunque viene utilizzata per la ricerca del dato da modificare in "database.txt".
Inoltre essa ritorna una sola stringa, ovvero la prima occorrenza che coincide con la chiave di ricerca, mentre cerca() le stampa tutte e non effettua nessun ritorno.

- **La funzione visualizzapwd()**

Questa funzione è un'esclusiva dell'admin e consente allo stesso di visionare un riepilogo delle credenziali per ogni utente. È una semplice stampa in output di stringhe di un file ed avviene con l'utilizzo di un ciclo while con la funziona fgets(), la quale consente di scansionare linea per linea il file e stamparlo a video.

- **La funzione elimina()**

Questa funzione viene richiamata internamente alla funzione modifica(). Consente di eliminare dei dati anagrafici da "database.txt". Il funzionamento si basa sull'utilizzo di cercamodifica() per l'individuazione della stringa da eliminare, chiedendo un'ulteriore conferma all'utente sulla volontà di cancellare i dati presenti, e sulla presenza di un file txt di appoggio ("temporaneo.txt"). Una volta confermata la cancellazione, la funzione scrive su file, tramite fprint(), tutte le linee (scansionate da un while con fgets()), tranne la stringa da eliminare.
In seguito viene eliminato con la funzione delete() il vecchio file "database.txt" e rinominato il file "temporaneo.txt" in "database.txt" con la funzione rename().
Entrambe queste funzioni vengono controllate tramite if(*funzione)*, restituendo un messaggio di errore in output nel caso in cui non fosse possibile eliminare e/o rinominare il file. Un caso di errore di questo genere, può presentarsi se il singolo file o entrambi risultano essere ancora aperti durante l'esecuzione.

- **Le funzioni modificautente(), cercamodifcacredenziali() ed eliminautente()**

Queste funzioni riprendono il funzionamento delle funzioni modifica(), cercamodifica() ed elimina(), però, si dedicano alla modifica, alla ricerca ed alla eliminazione di credenziali, dunque operano sul file "password.txt". Vengono utilizzate le funzioni già citate per le operazioni sui file, sulle stringhe e sulle variabili.

- **Considerazioni di carattere generale**

Tutti i nomi dei file utilizzati all'interno del programma sono dichiarati all'inizio attraverso la direttiva "#define".
In ogni operazione di apertura a file viene restituito un puntatore a una variabile di tipo FILE. Per poter leggere e scrivere i file è necessario l'uso di variabili di tipo puntatore a file (FILE *fp).
L'apertura di tutti i file avviene tramite la funzione fopen(nomefile, modalità).
Nell'intero programma è presente la gestione dell'apertura e della chiusura del file nelle varie modalità quali read, write ed append.

All'interno della funzione modifica() viene utilizzato un array di puntatori. La scelta di implementare questa soluzione sta nella migliore gestione della suddivisione della stringa attraverso strtok(). Salvando i vari token nell'array, la modifica del singolo dato avviene in maniera precisa ed immediata.

In tutto il programma sono utilizzate le funzioni dell'header file string.h, esse consentono l'utilizzo delle stringhe e la loro manipolazione (strtok(), strstr(), strcmp()).

Francesco Avantaggiato