Laboratorio di Informatica L-A
Prova d'Esame 4.1 – 19 Aprile 2007

Prima di cominciare: si scarichi il file **`StartKit4.1.zip`** contenente i file di testo necessari.
Avvertenze per la consegna: nominare i file sorgenti come richiesto durante il testo del compito, apporre all'inizio di ogni file sorgente un commento contenente i propri dati (**cognome, nome, numero di matricola**) e il **numero** della prova d'esame. Al termine, **consegnare tutti i file sorgenti** ed i file contenuti nello StartKit.
Rispettare **ALLA LETTERA** le specifiche, in particolare inserire le funzioni nei file specificati fra parentesi dopo il nome della funzione. Chi non rispetta le specifiche sarà opportunamente penalizzato. **NON SARANNO CORRETTI** gli elaborati che presenteranno un numero "non affrontabile" di errori di compilazione.
Consiglio: per verificare l'assenza di *warnings*, effettuare di tanto in tanto un *Rebuild All*.

Un dentista tiene traccia dei propri pazienti tramite un insieme di files. In particolare, in un file di testo denominato "pazienti.txt" tiene traccia, in ogni riga del file, dei seguenti dati:

- Codice fiscale, 16 caratteri terminati da un spazio
- Cognome del paziente, al massimo 64 caratteri, terminati dal carattere '@'
- Nome del paziente, al massimo 64 caratteri, terminati dal carattere '@'
- Nome del file contenente le fatture relative a quel cliente, al massimo 64 caratteri

Poi per ogni paziente, registra in un file di testo (il cui nome è salvato in pazienti.txt) i dati relativi alle fatture, nel seguente modo:

- Codice fiscale, ancora 16 caratteri terminati da spazio
- Costo dell'intervento, un numero (con virgola) rappresentante il costo in euro, terminato da uno spazio
- Data dell'intervento odontoiatrico, nella forma di aaaammgg

I problemi sorgono dal fatto che la segretaria del dentista, un po' sbadata, ha registrato i dati in maniera poco ortodossa. Il dentista infatti si è accorto che ogni tanto il codice fiscale non è completo (ad esempio, mancano le ultime n lettere), e che a volte, nel file relativo ad un cliente sono registrate le fatture relative ad un altro cliente.
Il dentista ha rinunciato a correggere il problema del codice fiscale parzialmente specificato, e si aspetta che il computer sia in grado di capire che "CHSFRC75H06A445X" è da considerarsi equivalente a "CHSFRC75" (eventuali problemi di omonimia non gli interessano, i suoi pazienti non sono poi così tanti).
Invece il dentista vuole a tutti i costi correggere l'anomalia delle fatture relative ad un cliente ma registrate nel file relativo ad un altro cliente (altrimenti non sa come fare a calcolare gli importi totali e ad incassare i soldi…).
Si deve realizzare un programma che stampi a video, in ordine per data ed, in caso di fatture con stessa data, anche in ordine crescente per l'importo, tutte le fatture registrate nei file sbagliati, con a fianco di ognuna il nome del file sbagliato. Nello **StartKit** si trovano alcuni file di testo di esempio: usare questi al fine di testare opportunamente la funzione.

Esercizio 1 – Lettura del codice fiscale (element.h/patient.h/patient.c)
Al fine di rappresentare correttamente i dati di un paziente, si dichiari in element.h una struttura patient con i campi opportunamente dimensionati. Si noti che il codice fiscale non è una stringa, bensì un insieme di 16 caratteri: per rappresentarlo si usi quindi un array di soli 16 caratteri. Qualora il c.f. memorizzato dovesse risultare di dimensione inferiore a 16, si termini tale array con un terminatore di stringa '\0' per segnalare l'anomalia.
Si realizzi la funzione readCF (patient.h/patient.c) che, presi in ingresso un buffer di caratteri, un carattere di separatore ed un puntatore a file, legga da tale file un codice fiscale e lo memorizzi nel buffer passato come parametro.

Laboratorio di Informatica L-A
Prova d'Esame 4.1 – 19 Aprile 2007

Si realizzi la funzione printCF (patient.h/patient.c) che, ricevuto un array di 16 caratteri, provveda a stampare il codice a video: si consideri attentamente che, nel caso il codice sia corretto, bisogna stampare esattamente 16 caratteri; nel caso invece in cui il codice non sia corretto, bisogna stampare tutti i caratteri fino a che si incontra il terminatore.
Suggerimenti: Per realizzare la funzione readCF, si consiglia di copiare e modificare in maniera opportuna la funzione readField in dotazione nel bundle.

Esercizio 2 – Lettura dei pazienti (patient.h/patient.c)
Si realizzi la funzione readPatient che, ricevuti in ingresso un puntatore ad un area di memoria sufficientemente grande (puntatore a patient), ed un puntatore a file, legga da tale file una struttura dati di tipo patient e la memorizzi nell'area di memoria passata come parametro. La funzione restituisca poi l'ultimo carattere letto (separatore) dal file (un newline o un EOF).
Si realizzi poi la funzione readPatientList che, ricevuto in ingresso il nome di un file, legga e restituisca la lista dei pazienti ivi contenuti. A tal scopo, si ridefinisca opportunamente il file element.h visto a lezione.
Suggerimenti: Per la funzione readPatient, si suggerisce di usare le funzioni readCF e la funzione readField. Per la readPatientList invece, si consiglia di utilizzare le sole primitive sulle liste viste a lezione.

Esercizio 3 – Lettura di fatture (fattura.h/fattura.c)
Dopo aver creato in fattura.h una struttura dati di nome fattura, opportunamente definita per rappresentare i dati relativi ad una fattura, si realizzi la funzione readFattura che, ricevuto in ingresso un'area di memoria (puntatore a fattura) allocata opportunamente ed un puntatore a file, legga da tale file una fattura e la memorizzi nell'area di memoria; la funzione restituisca poi l'ultimo carattere separatore letto.
Suggerimenti: Si consiglia di prevedere nella struttura dati fattura un campo nomeFile dove andare a scrivere il nome del file da cui tale fattura è stata letta: questo faciliterà di molto la soluzione dei punti successivi.

Esercizio 4 – Confronto di codici fiscali (patient.h/patient.c)
Si realizzi una funzione compareCF che, dati due codici fiscali, restituisca un intero di valore diverso da 0 se questi sono identici secondo la notazione stravagante usata dalla segretaria, o 0 se sono diversi. Si ipotizzi che la segretaria abbia sempre inserito almeno un carattere del c.f..

Esercizio 5 – Funzione di filtro e confronto di fatture (fattura.h/fattura.c)
Si realizzi una funzione compareFatture che, ricevuti in ingresso due fatture, restituisca un valore negativo, pari a 0 o positivo a seconda che la prima fattura preceda (per data) o segua la seconda. In caso di fatture con stessa data, si consideri allora anche l'importo (la fattura con importo minore precede la fattura con importo maggiore).
Si realizzi poi una funzione filter che, ricevuta in ingresso una lista di pazienti, per ognuno di questi apra il file relativo, legga tutte le fatture ivi registrate e le memorizzi in un opportuno vettore allocato dinamicamente (di dimensione massima 1000). L'inserimento delle fatture deve essere effettuato tramte la funzione insert vista a lezione e modificata opportunamente (si utilizzi a tal scopo la funzione compareFatture). Si abbia cura di chiudere i files non più in utilizzo. La funzione restituisca un puntatore all'area di memoria allocata, ed un intero (passato se necessario per riferimento) rappresentante la dimensione logica del vettore.

Esercizio 6 – Main (main.c)
Si scriva un programma main che, tramite la funzione readPatientList legga la lista dei pazienti, e tramite la funzione filter legga tutte le fatture sbagliate. Si abbia cura di stampare a video le fatture, indicando per ogni fattura il file dove essa è contenuta…