Laboratorio di Informatica L-A
Prova d'Esame 6 – 19 Luglio 2007

Prima di cominciare: si scarichi il file **`StartKit6.zip`** contenente i file necessari.
Avvertenze per la consegna: nominare i file sorgenti come richiesto nel 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 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*.

Si realizzi un programma che permetta di gestire le informazioni relative ad un insieme di carte di credito ed ai pagamenti effettuati, individuando potenziali truffe. Le informazioni sono contenute in due differenti file, denominati rispettivamente **`clienti.txt`** e **`pagamenti.txt`**.
Il primo file memorizza le informazioni "statiche" relative ai clienti, ovvero **CODICE** della carta di credito (16 caratteri), **NOME** del proprietario (al più 63 caratteri), **DATA DI SCADENZA** della carta; i campi sono separati dal carattere '**@**'. Le date di scadenza sono memorizzate nel formato **MM/AA** (l'anno è da considerarsi riferito al 2000).
Il secondo file memorizza le informazioni "dinamiche" relative ai pagamenti effettuati con le carte di credito. Il file, per ogni riga, memorizza la **DATA DI PAGAMENTO**, il **CODICE** della carta utilizzata, l'**IMPORTO** (numerico), la **VALUTA** (3 caratteri) e la **DESCRIZIONE** (al più 63 caratteri); i cinque campi sono separati dal carattere spazio. Le date di pagamento sono memorizzate nella forma **GG/MM/AAAA** e la descrizione può a sua volta contenere degli spazi.

Esercizio 1 – Gestione delle date (data.h/data.c)

Il candidato realizzi un modulo per la gestione delle date. L'astrazione "data" serve per rappresentare giorno, mese e anno sia degli acquisti effettuati, sia delle date di scadenza delle carte di credito (si consideri come giorno di scadenza di una carta di credito il giorno 1 del mese in cui essa scade). Il modulo, oltre alla definizione di una struttura dati appropriata, deve contenere una funzione **`compare(…)`** **`(data.h/data.c)`** per determinare quale di due date passate come parametro preceda l'altra, e due funzioni per trasformare (a) una stringa in formato MM/AA nella corrispondente struttura di tipo data (**`shortStringToDate(…)`**, **`data.h/data.c`**); e (b) una stringa in formato GG/MM/AAAA nella corrispondente struttura di tipo data (**`longStringToDate(…)`**, **`data.h/data.c`**). Contestualmente, il candidato scriva nel main opportune istruzioni per invocare le funzioni definite, al fine di verificarne la corretta implementazione.

Esercizio 2 – Gestione dei clienti (clienti.h/clienti.c)

Il candidato realizzi un modulo che definisca strutture dati opportune per rappresentare un cliente, e definisca le funzioni per (a) leggere da file i clienti (**`readClienti(…)`**, **`client.h/client.c`**) e (b) visualizzare a video (**`printCliente(...)`**, **`clienti.h/clienti.c`**) le informazioni relative ad un cliente. La funzione di lettura deve ricevere come parametri di ingresso il nome di un file (in cui andare a leggere) e deve restituire un array di clienti allocato dinamicamente (si abbia cura di allocare solo ed unicamente lo spazio strettamente necessario); inoltre la dimensione del vettore deve essere restituita tramite un ulteriore apposito parametro in ingresso. Il candidato valuti l'eventuale utilizzo della funzione **`rewind(…)`**. Contestualmente, il candidato scriva nel main opportune istruzioni per invocare le funzioni definite al fine di verificarne la corretta implementazione: usi a tale scopo il file **`clienti.txt`** fornito nello starter kit.

Laboratorio di Informatica L-A
Prova d'Esame 6 – 19 Luglio 2007

Esercizio 3 – Gestione dei pagamenti (pagamenti.h/pagamenti.c)

Il candidato realizzi un modulo che definisca strutture dati opportune per rappresentare un pagamento, e definisca opportune funzioni per (a) leggere da file i pagamenti (**`readSpese(…)`**, **`pagamenti.h/pagamenti.c`**), (b) visualizzare le informazioni relative ad un pagamento (**`printSpesa(…)`**, **`pagamenti.h/pagamenti.c`**) e (c) visualizzare le informazioni relative ad una lista di pagamenti (**`printSpese(…)`**, **`pagamenti.h/pagamenti.c`**). La lettura, ricevuta in ingresso il nome del file, deve restituire la lista di tutti i pagamenti effettuati dai vari clienti (si definisca in maniera opportuna l'adt lista e le primitive necessarie). Le funzioni di stampa, invece, devono ricevere in ingresso rispettivamente una spesa e una lista di spese, e visualizzarle a video. Contestualmente, il candidato scriva nel main opportune istruzioni per invocare le funzioni definite, al fine di verificarne la corretta implementazione: usi a tale scopo il file **`pagamenti.txt`** fornito nello starter kit.

Il modulo deve inoltre fornire una funzione **`selectSpese(…)`** (**`pagamenti.h/pagamenti.c`**) per selezionare i pagamenti effettuati da un certo cliente. La funzione, ricevuto in ingresso un determinato cliente ed una lista di spese, deve restituire un vettore dei pagamenti da lui effettuati, ordinati per data (suggerimento: si sfrutti l'inserimento ordinato visto a lezione, modificandolo in maniera opportuna). Il vettore deve essere allocato dinamicamente; il candidato inoltre abbia cura di allocare solo ed unicamente lo spazio strettamente necessario: la dimensione del vettore dovrà essere restituita tramite un opportuno parametro. Contestualmente, il candidato scriva nel main opportune istruzioni per invocare le funzioni definite, al fine di verificarne la corretta implementazione: usi a tale scopo i files forniti nello starter kit.

Esercizio 4 – Gestione delle truffe (truffe.h/truffe.c)

Il modulo di gestione di eventuali truffe deve analizzare la lista dei pagamenti effettuati da un singolo cliente per individuare alcune situazioni anomale, mostrando, per ogni diversa situazione anomala, la lista dei clienti coinvolti.

In particolare, si considerino le seguenti anomalie:

- Presenza di pagamenti effettuati con una carta di credito scaduta: si realizzi una funzione **`findTruffaTipo1(…)`** (**`truffe.h/truffe.c`**) che, ricevuto in ingresso i dati di un cliente e un vettore con le spese da lui effettuate (si supponga tale vettore ordinato per data, ottenuto tramite le funzioni definite nell'esercizio precedente), restituisca un valore vero se almeno un pagamento è stato effettuato dopo la scadenza della carta di credito.
- Presenza di pagamenti effettuati lo stesso giorno con diverse valute: si realizzi una funzione **`findTruffaTipo2(…)`** (**`truffe.h/truffe.c`**) che, ricevuto in ingresso i dati di un cliente e un vettore con le spese da lui effettuate (si supponga tale vettore ordinato per data, ottenuto tramite le funzioni definite nell'esercizio precedente), restituisca un valore vero se nello stesso giorno sono stati effettuati almeno due pagamenti, con valuta diversa.

Contestualmente, il candidato scriva nel main opportune istruzioni per invocare le funzioni definite, al fine di verificarne la corretta implementazione.

Infine il candidato scriva nel **`main.c`** le istruzioni necessarie per stampare a video i nomi dei clienti che potrebbero essere stati coinvolti in un'eventuale truffa, ed abbia cura di deallocare (al termine del programma) tutte le strutture allocate dinamicamente.