Laboratorio di Informatica L-A
Prova d'Esame 1.3 – 18 Dicembre 2006

Prima di cominciare: si scarichi il file **`StartKit1.3.zip`** contenente i file sorgenti relativi all'ADT lista, il file di testo **`clienti.txt`** e il file binario **`accessi.dat`**
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**.

In una banca si è verificato un furto in una delle cassette di sicurezza. Il ladro, che ha sfruttato una falla nel sistema di protezione delle cassette, è probabilmente da ricercarsi fra i clienti, poiché l'ingresso alla sala non presenta nessun segno di manomissione.
La banca tiene l'elenco dei suoi clienti in un file di testo (**`clienti.txt`**).
Ogni riga del file memorizza un singolo **`cliente`**, col seguente formato:

- **`codice`** identificativo (intero);
- **`nome`** (stringa di al più 20 caratteri terminata da **`';'`**);
- **`cognome`** (stringa di al più 20 caratteri terminata da **`';'`**);
- **`classe`** del cliente che memorizza lo 'status' del cliente (carattere singolo) – **`'B'`** se base, **`'P'`** se premium.

In un secondo file binario (**`accessi.dat`**), il sistema di accesso alla sala memorizza gli accessi effettuati nella giornata precedente (un cliente, in un giorno, può accedere al massimo una volta alla sala).
Per ogni **`accesso`**, nel file vengono tracciati:

- **`codice`** del cliente;
- **`tempo`** di permanenza, in minuti, all'interno della sala (intero);
- sistema di **`autenticazione`** utilizzato (un carattere singolo) – **`'R'`** per la scansione della retina e **`'N'`** per la classica autenticazione manuale).

Il primo valore di tale file è un numero intero che rappresenta il **numero di accessi** registrato.

Esercizio 1
Si realizzi un modulo software (file **`clienti.h`** e **`clienti.c`**) contenente la definizione di **`cliente`** e le funzioni che seguono:

1. **`leggiClienti`**: legge da un file di testo contenente i clienti e memorizza in un vettore dalle opportune dimensioni l'elenco dei clienti; tale funzione deve prendere come parametri il nome di un file e restituire il vettore con i dati e la relativa dimensione.
2. **`stampaClienti`**: stampa l'elenco dei clienti che viene passato come primo parametro; come secondo parametro viene passato il numero di clienti da stampare.

Si verifichi il funzionamento del modulo scrivendo nel **`main`** le invocazioni di **`leggiClienti`** e **`stampaClienti`** ed utilizzando il file **`clienti.txt`** contenuto nello *Start Kit.*

Esercizio 2
Si realizzi un modulo software (file **`accessi.h`** e **`accessi.c`**) contenente la definizione di **`accesso`** ed una funzione di nome **`leggiAccessi`** che legga gli accessi da un file binario e memorizzi gli accessi in un vettore dalle opportune dimensioni; tale funzione deve prendere in ingresso il nome di un file e restituire il vettore con i dati e la dimensione del vettore restituito. Si verifichi il funzionamento del modulo scrivendo nel **`main`** l'invocazione di

leggiAccessi utilizzando il file **accessi.dat** (si lascino le istruzioni di verifica per l'esercizio 1).

Esercizio 3

Si realizzi un modulo software (file **sospetti.h** e **sospetti.c**) contenente la definizione di **sospetto** e le funzioni che seguono:

1. Una funzione di nome **trovaSospetti** che incroci i dati dei due vettori (clienti e accessi) per determinare una lista di possibili sospetti; tale funzione deve prendere come parametri un array di clienti ed un array di accessi (e le relative dimensioni logiche) e deve restituire una lista di sospetti (per lavorare con le liste utilizzare le primitive date). Per un **sospetto** si tiene traccia di:
   - **nome**;
   - **cognome**;
   - sistema di **autenticazione**;
   - numero di **minuti** di permanenza della sala.

   Devono essere considerati sospetti tutti i clienti base che hanno effettuato accesso alla sala e si sono trattenuti per un tempo compreso fra 20 e 30 minuti, tranne quelli che hanno effettuato accesso tramite scansione della retina.
2. Una funzione ricorsiva di nome **stampaSospetti** che, utilizzando le primitive sulle liste, stampi a video la lista di sospetti che viene passata come parametro.
3. Una funzione di nome **scriviSospetti** che, utilizzando la notazione a puntatori, scriva su un file di testo – il cui nome è passato come parametro alla funzione – i soli campi **nome**, **cognome** e **minuti** di permanenza dei sospetti contenuti nella lista passata come parametro.

Si verifichi il funzionamento scrivendo nel **main** le invocazioni di **trovaSospetti**, **stampaSospetti** e **scriviSospetti** utilizzando l'output delle funzioni di lettura di cui agli esercizi precedenti e scrivendo la lista dei sospetti (**scriviSospetti**) sul file **sospetti.txt**.

Laboratorio di Informatica L-A
Prova d'Esame 1.3 – 18 Dicembre 2006

clienti.h

```
#include <stdio.h>
#include <stdlib.h>

#ifndef CLIENTI
#define CLIENTI

#define DIM_NOME 21
#define DIM_COGNOME 21
typedef struct
{
	int codice;
	char nome[DIM_NOME];
	char cognome[DIM_COGNOME];
	char classe;
}
cliente;
#endif

cliente * leggiClienti(char * fileName, int * dim);
void stampaClienti(cliente v[], int dim);
```

clienti.c

```
#include "clienti.h"

/* PRESA DALLE SLIDES DEL CORSO... */
int readField(char buffer[], char sep, FILE *f)
{
	int i = 0;
	char ch = fgetc(f);
	while (ch != sep && ch != 10 && ch != EOF)
	{
		buffer[i++] = ch;
		ch = fgetc(f);
	}
	buffer[i] = '\0';
	return i;
}

cliente * leggiClienti(char * fileName, int * dim)
{
	FILE * fp;
	cliente temp;
	cliente * result;
	int i, count, ok;

	if ((fp=fopen(fileName, "r")) == NULL)
	{
		printf("Error opening the file %s\n", fileName);
		exit(-1);
	}
	count = 0;
	ok=1;
	while (ok)
	{
		if (fscanf(fp, "%d", &temp.codice)!= 1)
```

```
                ok=0;
        if (ok)
                ok = readField(temp.nome, ';', fp);
        if (ok)
                ok = readField(temp.cognome, ';', fp);
        if (ok)
                ok = (temp.classe = fgetc(fp));
        if (ok)
                count++;
    }

    result = (cliente *) malloc(count * sizeof(cliente));
    rewind(fp);
    i=0;
    ok = 1;
    while (ok && i<count)
    {
        if (fscanf(fp, "%d", &result[i].codice)!= 1)
                ok=0;
        fgetc(fp);
        if (ok)
                ok = readField(result[i].nome, ';', fp);
        if (ok)
                ok = readField(result[i].cognome, ';', fp);
        if (ok)
                ok = (result[i].classe = fgetc(fp));
        if (ok)
                i++;
    }
    fclose(fp);
    *dim = i;
    return result;
}

void stampaCliente(cliente c)
{
    printf("Codice Cliente: %d\n", c.codice);
    printf("Cognome:        %s\n", c.cognome);
    printf("Nome:           %s\n", c.nome);
    printf("Classe:         %c\n", c.classe);
    printf("\n");
}

void stampaClienti(cliente v[], int dim)
{
    int i;
    for (i=0; i<dim; i++)
        stampaCliente(v[i]);
}
```

Laboratorio di Informatica L-A
Prova d'Esame 1.3 – 18 Dicembre 2006

accessi.h

```
#include <stdio.h>

#ifndef ACCESSO
#define ACCESSO
typedef struct
{
	int codice;
	int tempo;
	char authMode;
}
accesso;
#endif

accesso * leggiAccessi(char * fileName, int * dim);
void creaBinario(char * fileName);
```

accessi.c

```
#include <stdlib.h>
#include "accessi.h"

accesso * leggiAccessi(char * fileName, int * dim)
{
	FILE * fp;
	accesso * result;
	int i;

	if ((fp=fopen(fileName, "rb")) == NULL)
	{
		printf("Error opening the file %s\n", fileName);
		exit(-2);
	}

	fread(dim, sizeof(int), 1, fp);
	result = (accesso *) malloc(*dim * sizeof(accesso));
	i=0;
	while (fread(&result[i], sizeof(accesso), 1, fp)==1 && i<*dim)
		i++;

	fclose(fp);
	return result;
}

void creaBinario(char * fileName)
{
	FILE * fp;
	int i = 4;
	accesso temp;

	fp = fopen(fileName, "wb");

	fwrite(&i, sizeof(int), 1, fp);

	temp.codice = 12;
	temp.authMode = 'N';
	temp.tempo = 15;
```

```
        fwrite(&temp, sizeof(accesso), 1, fp);

        temp.codice = 45;
        temp.authMode = 'N';
        temp.tempo = 16;
        fwrite(&temp, sizeof(accesso), 1, fp);

        temp.codice = 37;
        temp.authMode = 'R';
        temp.tempo = 22;
        fwrite(&temp, sizeof(accesso), 1, fp);

        temp.codice = 42;
        temp.authMode = 'R';
        temp.tempo = 22;
        fwrite(&temp, sizeof(accesso), 1, fp);

        fclose(fp);
}
```

Laboratorio di Informatica L-A
Prova d'Esame 1.3 – 18 Dicembre 2006

sospetti.h

```
#include <stdio.h>
#include "clienti.h"
#include "accessi.h"
#include "element.h"
#include "list.h"

list trovaSospetti(cliente v[], int dimCliente, accesso a[], int dimAccesso);
void stampaSospetti(list temp);
void scriviSospetti(char * fileName, list listaSospetti);
```

sospetti.c

```
#include <string.h>
#include "sospetti.h"

list trovaSospetti(cliente v[], int dimClienti, accesso a[], int dimAccesso)
{
	int i, j;
	int trovato = 0;
	sospetto temp;
	list result;

	result = emptylist();
	for (i=0; i<dimAccesso; i++) {
		if (a[i].tempo>=20 && a[i].tempo<=30 && !(a[i].authMode == 'R'))
		{
			trovato = 0;
			for (j=0; j<dimClienti && !trovato;)
			{
				if (a[i].codice == v[j].codice)
					trovato = 1;
				else
					j++;
			}
			if (v[j].classe == 'B')
			{
				strcpy(temp.cognome, v[j].cognome);
				strcpy(temp.nome, v[j].nome);
				temp.authMode = a[i].authMode;
				temp.tempo = a[i].tempo;
				result = cons(temp, result);
			}
		}
	}
	return result;
}

void stampaSospetto(sospetto s)
{
	printf("Cognome:	%s\n", s.cognome);
	printf("Nome:		%s\n", s.nome);
	printf("AuthMode:	%c\n", s.authMode);
	printf("Tempo:		%d\n", s.tempo);
	printf("\n");
}
```

Laboratorio di Informatica L-A
Prova d'Esame 1.3 – 18 Dicembre 2006

```
void stampaSospetti(list temp)
{
	if (empty(temp))
		return;
	else
	{
		stampaSospetto(head(temp));
		stampaSospetti(tail(temp));
		return;
	}
}

void scriviSospetti(char * fileName, list listaSospetti)
{
	FILE * fp;

	if ((fp=fopen(fileName, "w")) == NULL)
	{
		printf("Error opening file %s\n", fileName);
		exit(-3);
	}
	while (listaSospetti != NULL)
	{
		fprintf(fp, "%s %s %d\n",
			listaSospetti->value.nome,
			listaSospetti->value.cognome,
			listaSospetti->value.tempo);
		listaSospetti = listaSospetti->next;
	}
	fclose(fp);
}
```

Laboratorio di Informatica L-A
Prova d'Esame 1.3 – 18 Dicembre 2006

element.h

```
#include "clienti.h"
#include "accessi.h"

#ifndef ELEMENT
#define ELEMENT

typedef struct
{
	char nome[DIM_NOME];
	char cognome[DIM_COGNOME];
	char authMode;
	int tempo;
}
sospetto;

typedef sospetto element;
#endif
```

main.c

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>

#include "clienti.h"
#include "accessi.h"
#include "sospetti.h"

int main(void)
{
	int i=0;
	cliente * vClienti;
	int dimClienti;
	accesso * vAccessi;
	int dimAccessi;
	list listaSospetti;

	vClienti = leggiClienti("clienti.txt", &dimClienti);
	stampaClienti(vClienti, dimClienti);

	/*	creaBinario("accessi.dat"); */
	vAccessi = leggiAccessi("accessi.dat", &dimAccessi);
	listaSospetti = trovaSospetti(vClienti, dimClienti, vAccessi, dimAccessi);
	stampaSospetti(listaSospetti);
	scriviSospetti("sospetti.txt", listaSospetti);

	system("PAUSE");
	return (0);
}
```