ELEMENTI DI INFORMATICA LB - PROVA PRATICA DEL 27/03/2007

1. LINGUAGGIO C (15 punti)

È dato un file di testo denominato BIOLOGICO.TXT che contiene un elenco di informazioni sui prodotti di una grande azienda italiana specializzata in prodotti biologici.
Il file contiene al massimo 20 righe e ogni riga contiene:

- un insieme di caratteri e cifre in cui sono specificati il codice identificativo del prodotto (8 caratteri e/o cifre)
- la data di scadenza, nel formato aaaa/mm/gg
- un insieme di 10 caratteri e/o cifre che rappresentano un codice per l'identificazione dei principali ingredienti contenuti nel prodotto (se composito).
- un booleano che determina se il prodotto è di origine naturale non composito o composto da sostanze di origine naturale (ad esempio confezione di susine: prodotto di origine naturale non composito; confezione di insalata: prodotto di origine naturale composito (ingredienti: l'insieme delle verdure utilizzate).

Si chiede di scrivere un programma C, biologico, con le seguenti caratteristiche:

Il programma dovrà essere suddiviso in 3 file.

1. **funzioni.h:** dovrà contenere le definizioni dei tipi e le dichiarazioni delle funzioni sotto descritte
2. **funzioni.c:** dovrà contenere le definizioni delle funzioni dichiarate in funzioni.h
3. **main.c:** dovrà contenere solo la funzione main()

Il programma dovrà comprendere:

Tipi di strutture

1. **prodotto** destinato a contenere i dati di una riga del file BIOLOGICO.TXT
2. **prodotti** che contiene un array di strutture di tipo **prodotto** e un intero che rappresenta il numero di elementi presenti nell'array.

Funzioni e procedure

1. **funzione** denominata **leggi():** dato il **nome di un file** riempie una lista di strutture di tipo **prodotto** contenente le righe lette dal file BIOLOGICO.TXT. La funzione restituisce 1 se l'operazione è andata a buon fine, altrimenti restituisce 0.

2. **funzione** denominata **conta():** data una lista di strutture di tipo **prodotto** restituisce il **numero di prodotti in scadenza in data odierna (27/03/2007)**

Il programma, utilizzando le suddette funzioni, dovrà:

1. leggere il file **BIOLOGICO.TXT** usando la funzione **leggi()** segnalando a video eventuali problemi; **(indicativamente fino a qui 10 punti)**
2. usare la funzione **conta()** per calcolare il numero dei prodotti in scadenza nella data odierna e mostrarlo a video; **(indicativamente fino a qui 15 punti)**

2. DATA BASE (15 punti)

SQL (9 punti)

1. CREARE LE TABELLE

Per scrivere le istruzioni SQL, utilizzate un qualsiasi editor di testo (ad esempio `Notepad`).
Si consideri la tabella `PERSONE`, contenente i seguenti attributi

- nome : 20 caratteri (chiave);
- eta : intero (3 cifre);
- reddito : intero (10 cifre);
- sesso : 'M' oppure 'F';
- residenza : 20 caratteri.

e la tabella `GENITORI`, contenente i seguenti attributi

- genitore : 20 caratteri;
- figlio : 20 caratteri;
- ordFiglio : intero (2 cifre). (1 per il primo figlio, 2 per il secondo, ect).

(Figlio,Genitore) e` chiave primaria. Inoltre, ogni figlio è una persona ed ogni genitore è una persona. Si forniscano le istruzioni SQL per creare le due tabelle.

2. EFFETTUARE INTERROGAZIONI

Fornite le interrogazioni SQL per rispondere alle seguenti domande.

1. Trovare le persone che sono genitori di almeno 2 figli;
2. Trovare l'elenco ordinato dei genitori i cui figli guadagnano tutti piu` di 30.000 euro
3. Definire una vista EtaMassimaFigli che restituisce per ogni persona l'età massima dei suoi figli.

la soluzione dovrà essere salvata su di un file denominato **esame20070327.sql**

SCHEMA CONCETTUALE: VENDITE E AFFITTI DI IMMOBILI (7 punti)

SPECIFICA

definire uno schema Entità/Relazione che descriva il seguente problema:

Occorre gestire informazioni su :

- Le transazioni (con codice, data, valore) di vendita e affitto immobili, che vengono effettuate dalle agenzie, delle quali interessa il numero identificativo, e la città di residenza. Ogni transazione è effettuata da una agenzia e riguarda un immobile.
- Gli immobili (con codice, indirizzo, città di ubicazione, metri quadrati e numero locali) oggetto delle transazioni. Alcuni immobili sono di interesse storico, e di essi interessa l'anno di costruzione. Altri sono ristrutturati, e di essi interessa la data di ultima ristrutturazione.
- Gli enti che acquistano, vendono, danno in affitto o prendono in affitto gli immobili. Degli enti interessa il codice fiscale, l'indirizzo, la città e la regione di residenza. Gli enti sono persone (dei quali interessa anche nome, cognome, professione e città di nascita) o aziende (delle quali interessa il capitale sociale e il numero di dipendenti).
- Degli affitti interessa anche il periodo di affitto.

la soluzione dovrà essere consegnata sul foglio di protocollo timbrato che viene consegnato all'inizio della prova.

SOLUZIONI

1. LINGUAGGIO C (15 punti)

funzioni.h

```
typedef enum {false,true} boolean;

typedef struct
{
      char id[8+1];
      char scade[10+1];
      char ingr[10+1];
      boolean nat;} prodotto;

typedef struct list_element
{
      prodotto value;
      struct list_element *next;} item;

typedef item *list;

typedef struct
{
```

```
      prodotto p[19];
    int i;} prodotti;


/* ADT */
list emptyList(void);
boolean isEmpty(list);
prodotto head(list);
list tail(list);
list cons(prodotto,list);

/* Funzioni non Primitive */
void showList(list);
int leggi(char*,list*);
int conta(list);
```

funzioni.c

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

list emptyList(void)
{
      return NULL;
}

boolean isEmpty(list l)
{
      if(l==NULL)
      {
            return true;
      }
      else
      {
            return false;
      }
}

prodotto head(list l)
{
      if(isEmpty(l))
      {
            abort();
      }
      else
      {
            return l->value;
      }
}

list tail(list l)
{
      if(isEmpty(l))
      {
            abort();
      }
      else
      {
```

```
            return l->next;
      }
}

list cons(prodotto e,list l)
{
      list t;
      t=(list) malloc(sizeof(item));
      t->value=e;
      t->next=l;
      return t;
}

int leggi(char* nomeFile,list *l)
{
      FILE *fp;
      prodotto p;
      int letti=0;

      list temp=emptyList();
      if((fp=fopen(nomeFile,"r"))==NULL)
      {
            return 0;
      }
      else
      {

      while(fscanf(fp,"%s%s%s%d\n",p.id,p.scade,p.ingr,&(p.nat))!=EOF
)
            {
                  temp=cons(p,temp);
                  letti++;
            }
            (*l)=temp;
            fclose(fp);
            return 1;
      }
}


int conta(list l)
{
      int i=0;
      list t=l;
      prodotto temp;
      char *data="2007/03/27";
      while(!isEmpty(t))
      {
            temp=head(t);
            if((strcmp(temp.scade,data))==0)
            {
                  ++i;
            }
            t=tail(t);
      }
      return i;
}

/* opzionale */
void showList(list l)
{
```

```
	list t=l;
	prodotto temp;

	while(!isEmpty(t))
	{
		temp=t->value;
		printf("ID: %s, scade: %s, principali ingredienti: %s\n",
temp.id,temp.scade,temp.ingr);
		t=t->next;
	}
}
```

main.c

```
#include <stdio.h>
#include "funzioni.h"

int main()
{
	int i=0;
	prodotti p;
	list l=emptyList();
	if((leggi("BIOLOGICO.txt",&l))==0)
	{
		printf("Errore nell'apertura del file");
		exit(1);
	}

	/* da qui */
	printf("\nInizio Elenco\n");
	showList(l);
	printf("Fine Elenco\n");
	/* a qui opzionale */

	i=conta(l);
	printf("\nOggi scadono %d prodotto/i\n",i);
	return 0;
}
```

BIOLOGICO.TXT

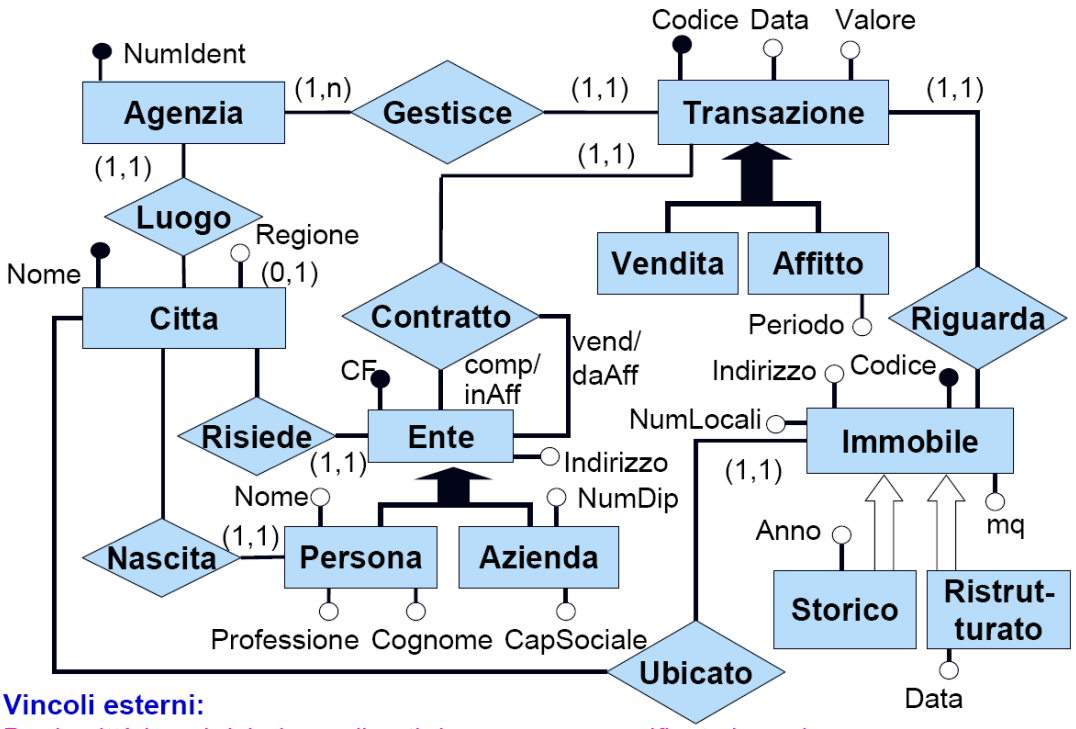
```
7FYHR8R0 2007/03/27 dfye874h4w 0
84H32NDI 2006/06/28 geyud873ur 0
HFUH334D 2006/04/30 nhxsdoiu39 1
JIUROEJD 2006/06/26 0987fgrtey 0
478EEH33 2006/07/15 cazshtyruc 1
```

2. SQL (9 punti)

```
CREATE TABLE Persone(
  nome                  CHARACTER (20)    PRIMARY KEY,
  eta                   NUMERIC(3),
  reddito               NUMERIC(10),
  sesso                 CHAR CHECK (sesso = 'M' OR sesso = 'F'),
  residenza             CHARACTER (20)
  );
```

```sql
CREATE TABLE Genitori(
  genitore              CHARACTER (20)    REFERENCES Persone(nome),
  figlio                CHARACTER (20)    REFERENCES Persone(nome),
  ordFiglio             NUMERIC(2),
                        PRIMARY KEY (figlio,genitore)
  );

-- Trovare le persone che sono genitori di almeno 2 figli.

SELECT genitore
FROM Genitori
GROUP BY genitore
HAVING COUNT(figlio) >= 2;

-- Trovare l'elenco ordinato dei genitori i cui figli guadagnano
tutti piu` di 30.000 € (con l'operatore differenza).

SELECT DISTINCT genitore
FROM Genitori
  MINUS
SELECT DISTINCT genitore
FROM Genitori, Persone
WHERE nome = figlio AND reddito <= 30.000
ORDER BY Genitore;

-- Trovare l'elenco ordinato dei genitori i cui figli guadagnano
tutti piu` di 30.000 € (senza l'operatore differenza).

SELECT DISTINCT genitore
FROM Genitori
WHERE genitore NOT IN (SELECT genitore
                       FROM Genitori, Persone
                       WHERE nome = figlio AND reddito <= 30.000)
ORDER BY genitore;

-- Definire una vista EtaMassimaFigli che restituisce per ogni
persona l'eta` massima dei suoi figli.

CREATE VIEW EtaMassimaFigli (genitore, eta) AS
SELECT genitore, MAX(eta)
FROM Genitori, Persone
WHERE figlio = nome
GROUP BY genitore;
```

SCHEMA CONCETTUALE: VENDITE E AFFITTI DI IMMOBILI (7 punti)



Vincoli esterni:

Per le città in cui risiedono gli enti deve essere specificata la regione