Fondamenti di Informatica L-A (A.A. 2003/2004) - Ingegneria Informatica
Prof.ssa Mello & Prof. Bellavista – Seconda Prova Intermedia del 09/12/2003 - durata 2h
COMPITO B

ESERCIZIO 1 (12 punti)

Si scriva una funzione **`rec()`** che riceva come parametri di ingresso il nome di un file binario **`nome_file`**, un vettore **`spedizione`** di strutture **`camion`** (definite nel seguito), la dimensione fisica **`max`** del vettore `prezzi`, e un puntatore a intero **`num`** che rappresenta la dimensione logica del vettore. La funzione deve aprire il file binario `nome_file` (in caso di errore nell'apertura di tale file la funzione deve restituire immediatamente il valore –1); la funzione deve poi leggere dal file binario al più `max` strutture di tipo `camion`, che devono essere memorizzate nel vettore `spedizione`; la dimensione logica del vettore deve essere restituita tramite il parametro `num` (passato appunto per riferimento). Al termine della lettura dei dati la funzione deve restituire il valore 0. **(6 punti)**.

```
int rec (char nome_file[], camion spedizione[], int max, int * num);
```

È dato un file binario **`record.dat`**, contenente strutture di tipo **`camion`**, composte da una stringa **`nome`** (di al più 20 caratteri, indicante il nome del conducente) e due campi **`peso_vuoto`** e **`peso_pieno`**, entrambi di tipo `float`, indicanti rispettivamente il peso a vuoto del camion e il peso dello stesso quando carico di merce. Il file contiene al più 100 strutture di tipo `camion` (potrebbero essere anche in numero minore).

Si scriva un programma C che inserisca in un vettore **`V`** (supposto di dimensione massima **`DIM 100`**) tutti i dati presenti nel file `record.dat`; il programma quindi allochi dinamicamente un secondo vettore **`valore`** di `float`, la cui dimensione fisica deve essere pari ala dimensione logica di `V`. Il programma assegni ad ogni elemento di `valore` il valore della merce trasportata da ogni camion i cui dati sono nel corrispondente elemento di `V`, e ne stampi anche il risultato a video. Si ricorda che il valore del camion in posizione i-esima è così calcolato: **(6 punti)**

$$\text{valore[i]} = \left(peso\_pieno_i - peso\_vuoto_i\right) * 12€$$

ESERCIZIO 2 (9 punti)

Si scriva una funzione ricorsiva **`tre()`** che data in ingresso una lista di interi (non ordinati), restituisca in uscita una lista contenente solo gli elementi multipli di 3 della lista di origine, ordinati in maniera crescente. A tal scopo si supponga di possedere il tipo di dato astratto **`list`** e le sole operazioni PRIMITIVE (che quindi possono NON essere riportate nella soluzione): si consideri quindi attentamente il fatto che funzioni più elaborate come `member()` o `insord()` invece NON sono disponibili.
Si scriva poi una seconda funzione **`min()`** che, data in ingresso una lista di interi, determini il valore minimo contenuto nella lista, utilizzando la notazione a puntatori. La funzione può essere realizzata in maniera sia iterativa che ricorsiva.

ESERCIZIO 3 (8 punti)

Nel caso in cui il programma sorgente C seguente compili ed esegua correttamente, se ne indichino i valori stampati a tempo di esecuzione, motivando la risposta data. In caso di errori di compilazione o errori runtime, si descriva invece nel dettaglio la motivazione di tali errori.
Si indichino inoltre quali sono i blocchi in cui sono visibili gli identificatori **`N`** e **`L`**, sempre motivando la risposta.

```
#include <stdio.h>
#include <stdlib.h>
#define L 4

void G(int v[], int pos) {
    int N=L;
    N--;
    *(v+pos) = *(v+pos) - *(v+pos-1);
    return;
}

main () {
    int N=L;
    int * v;
    int  i;

    {
      ++N;
    }

    v = (int *) malloc((N) * sizeof(int));
    for (i=0; i<N; i++)
      v[i]=(i+1)*(i+1);

    for (i=L; i>0; i--)
      if (v[i]>0) G( v, i);

    printf("Adesso N vale: %d\n", N);

    for (i=N+1; (i--)>2?i:0; )
      printf("%d\n", v[i-1]);
}
```

Esercizio 4 (3 punti)
Dire se il seguente codice viene compilato correttamente. In caso positivo spiegare anche cosa fa e se è semanticamente corretto; in caso contrario spiegare perché il codice non viene compilato.

```
      int * V = (int *) malloc(sizeof(int) * 5);
      printf("%d\n", *(V+6));
```

Soluzione Compito B

Esercizio 1

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

#define DIM 20
#define MAX_RECORD 100

typedef struct {
    char nome[DIM];
    float peso_vuoto;
    float peso_pieno;
} camion;

int rec (char nome_file[], camion spedizione[], int max, int * num) {
    FILE * f;
    int i;

    f = fopen(nome_file, "rb");
    if (f == NULL)
      return(-1);

    *num = fread(spedizione, sizeof(camion), max, f);
    fclose(f);
    return 0;
}


int main() {
    camion V[MAX_RECORD];
    float * valore;
    int num, i, result;

    result  = rec("record.dat", V, MAX_RECORD, &num);

    if (result !=0) {
      printf("Problemi durante la lettura...\n");
      exit(-1);
    }

    valore = (float *) malloc(sizeof(float) * num);

    for (i=0; i < num; i++) {
      valore[i] = (V[i].peso_pieno - V[i].peso_vuoto)*12;
      printf("%6.2f\n", valore[i]);
    }
    return 0;
}
```

Esercizio 2

```
list  tre(list l) {
      if empty(l) return emptylist();
      else
            if (!(head(l)%3))
                  return insord(head(l), tre(tail(l)));
            else
                  return tre(tail(l));
}

list insord(element el, list l) {
      if (empty(l)) return cons(el,l);
      else if (el <= head(l)) return cons(el,l);
            else return cons(head(l), insord(el,tail(l)));
}

int min(list l) {
      int temp;
      if (empty(l))
            abort() ;
      else
            if (l->next == NULL) return l->value;
            else {
                  temp = min(l->next);
                  return (l->value < temp ? l->value : temp);
            }
}
```

Esercizio 3

Il programma stampa:
Adesso N vale: 5
9
7
5
3

Innanzitutto N viene pre-incrementato all'interno del blocco, e quindi vale 5. Viene poi allocato dinamicamente un vettore di interi di dimensione N, cioè 5, e viene riempito con i quadrati dei numeri da 1 a 5. Per ognuno di questi elementi è invocata la funzione G, che sovrascrive il valore in posizione pos. Il programma principale quindi stampa a video il valore di N, e poi stampa ogni elemento del vettore v.
La variabile N è dichiarata sia nel blocco main, sia nella funzione `G()`. N non è visibile globalmente, ma solo nei singoli blocchi. L è invece una costante simbolica visibile in ogni blocco.

Esercizio 4

Il codice viene compilato correttamente, ma non è corretto. Il puntatore V infatti punta ad una zona di memoria allocata dinamicamente per lo spazio di 5 interi; la seconda istruzione però cerca di stampare un elemento al di fuori della memoria allocata. Il codice presentato quindi tenterebbe di stampare a video una zona di memoria non allocata, il cui contenuto non è noto a priori.