Fondamenti di Informatica T-1 (A.A. 2009/2010) - Ingegneria Informatica
Prof.ssa Mello
Prova Parziale d'Esame di Lunedì 22 Dicembre 2009 – durata 1h
Totale 12 punti, sufficienza con 7
Compito A

ESERCIZIO 1 (6 punti)

Si supponga di avere a disposizione, già definite:

- l'ADT lista per interi (denominato **`list_int`**, con relative primitive **`emptylist_int(...)`**, **`cons_int(...)`**, **`head_int(...)`**, etc.).
- l'ADT lista per float (denominato **`list_float`**, con relative primitive **`emptylist_float(...)`**, **`cons_float(...)`**, **`head_float(...)`**, etc.).

Il candidato definisca una funzione *ricorsiva*:

```
list_float frequenze(list_int l1, list_int l2);
```

La funzione deve restituire una nuova lista contenente le frequenze con cui ognuno dei valori contenuti in **`l1`** compare nella lista **`l2`**. Per "frequenza" si intende il numero di volte che un elemento **`el`** compare in **`l2`** diviso il numero totale di elementi memorizzati in **`l2`**. Si presti attenzione al fatto che se un elemento **`el`** non compare mai in **`l2`**, la sua frequenza è **`0`**. Si tenga conto anche del possibile caso in cui **`l2`** sia una lista vuota: in tal caso la frequenza di un qualunque elemento è sempre **`0`**. La funzione deve essere implementata utilizzando le sole primitive dell'ADT lista.

Al fine di semplificare la funzione di cui sopra, il candidato realizzi una funzione *iterativa*:

```
float calcolaFrequenza(int el, list_int l);
```

che calcola la frequenza con cui l'elemento **`el`** compare in **`l`**. Tale funzione deve essere implementata accedendo alle liste tramite la notazione a puntatore, e senza fare ricorso alle primitive dell'ADT.

Ad esempio, se **`l1 = {1, 4, 2}`**, ed **`l2 = {1, 3, 4, 1, 3, 1}`**, la funzione **`frequenze(...)`** restituirà la lista **`{0.5, 0.166666, 0}`**, poiché il valore 1 compare 3 volte su un totale di 6 elementi, il valore 4 compare compare 1 volta su 6 elementi (1/6 = 0.166666), ed il valore 2 invece non compare affatto.

ESERCIZIO 2 (2 punti)

Un elaboratore rappresenta i numeri interi su 8 bit tramite la notazione in complemento a 2. Indicare come viene svolta la seguente operazione aritmetica calcolandone il risultato secondo la rappresentazione binaria in complemento a 2 (si trasli anche il risultato in decimale per verificare la correttezza dell'operazione):

```
45 + (-58)
```

ESERCIZIO 3 (3 punti)

Il seguente programma C compila correttamente? In caso affermativo, quali sono i valori stampati a tempo di esecuzione? (si motivi opportunamente la risposta data)

```
#include <stdio.h>
#include <stdlib.h>
int temp = 1;

int blip(char * v, char * p, int step){
      int i = step/2 - 1;
      int j = step/2;
      for (;i>=0 && j<step;) {
            p[i] = v[j];
            p[j] = v[i];
            i--, j++;
      }

      *(p+step) = '\0';
    return i;
}

int main(){
      char * p;
      char str[] = "stressed";
      int dim=0;
      int temp;

      do {
            if (str[dim])
                  dim++;
      } while (str[dim]!= '\0');
      p = (char *) malloc(sizeof(char)*(dim +1));
      temp = blip(str, p, dim);
      printf("%s %d\n", p, temp);
      free(p);
      return (0);
}
```

ESERCIZIO 4 (1 punto)

Illustrare in C come avviene il passaggio dei parametri, e cosa si intendo con l'espressione "passaggio per riferimento".

Soluzioni

ESERCIZIO 1

```
float calcolaFrequenza(int el, list_int l) {
      int dim = 0;
      int cont = 0;

      while (l != NULL) {
            if (l->value == el)
                  cont++;
            dim++;
            l = l->next;
      }
      if (dim == 0)
            return 0;
      else
            return ((float) cont)/dim;
}

list_float frequenze(list_int l1, list_int l2) {
      if (empty_int(l1))
            return emptylist_float();
      else
            return cons_float(
                        calcolaFrequenza(head_int(l1), l2),
                        frequenze(tail_int(l1), l2)
                  );
}
```

ESERCIZIO 2

```
45    ->    00101101                    00101101 +  (45)
+58   ->    00111010                    11000110 =  (-58)
            11000101                    ----------------
-58   ->    11000110                    11110011    (-13)

                                        11110011
                                        00001100
                                        ----------------
                                        00001101 -> (+13)
```

ESERCIZIO 3

Il programma è corretto sintatticamente e la sua esecuzione produce la stampa:
desserts -1

La funzione `main()` dapprima calcola la lunghezza della stringa `str`, e poi alloca memoria sufficente per contenere una stringa di lunghezza uguale (si noti che la dimensione viene aumentata di uno al fine di avere spazio a sufficienza anche per il termiantore). Quindi viene invacata la funzione `blip(…)`.

Tale funzione inizializza gli indici i e j ai due indici centrali dell'array v, grazie alla dimensione step passata come parametro. Nel caso particolare di una stringa di 8 caratteri, indici validi da 0 a 7, gli indici dei due caratteri centrali sono 3 e 4. La funzione procede poi ad assegnare al vettore p i caratteri di v, scambiandoli pero' di posizione tramite i e j: il carattere all'indice 4 di v finisce all'indice 3 di p, il carattere all'indice 3 di v finisce all'indice 4 di p, e cosi' via incrementando/decrementando rispettivametne le variabili i e j. Alla fine in p viene memorizzata la stringa str ma in ordine invertito. Un opportuno terminatore di stringa viene poi messo nell'array p nell'ultima posizione disponibile.

La funzione `main()` infine stampa a video la stringa p e l'intero ottenuto dalla funzione `blip(…)`.