Fondamenti di Informatica T-1 (A.A. 2009/2010) - Ingegneria Informatica
Prof.ssa Mello
Prova Parziale d'Esame di Giovedì 10 Giugno 2010 – durata 1h
Totale 12 punti, sufficienza con 7

ESERCIZIO 1 (5 punti)

Siano date una lista `l` ed un vettore `v` di uguale lunghezza, la lista ha come elementi stringhe ben formate, il vettore ha come elementi interi. Si realizzi una funzione (ricorsiva)

```
list atLeast(list l, int* v);
```

che restituisca una lista con gli elementi di `l1` che hanno un numero di **caratteri maiuscoli** almeno uguale all'intero dell'elemento corrispondente nel vettore `v`. Ad esempio se `l=["MARE", "SOle", "VAcaNZE"]` e `v=[4, 9, 3]`, la funzione `atLeast` restituisce la lista `["MARE", "VAcaNZE"]` poiché `"MARE"` ha almeno 4 caratteri maiuscoli e `"VAcaNZE"` almeno 3 caratteri maiuscoli mentre `"SOle"` ha meno di 9 caratteri maiuscoli.

La funzione `atLeast` dovrà essere implementata utilizzando le sole primitive dell'ADT lista; ogni altra funzione dovrà essere opportunamente specificata dal candidato. Si realizzi inoltre un semplice `main` di prova che invochi correttamente la funzione `atLeast` creata.

Nota: l'ordine degli elementi della lista restituita dalla funzione `atLeast` è ininfluente.

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):

```
37 - 51
```

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>

#define MAX 5

int move(int* v, char in[], int dim){
    int i, j;
    char temp[MAX];
    for(i=0; i<MAX; i++){
        if(v[i]>0){
            temp[i] = in[v[i]%MAX];
        }
        else{
            while(v[i]<0){
                v[i] += dim;
            }
            temp[i] = in[v[i]%MAX];
        }
    }
    for(j=0; j<i; j++){
        in[j] = temp[j];
    }
    return j-1;
}

int main(){
    int i, dim = 0;
    int shift[] = {2, 7, -9, 1, -6};
    char* array;
    char* temp = "ciao a tutti quanti";
    while( *(temp+dim) != '\0' ){
        dim++;
    }
    dim++;
    array = (char*) malloc (sizeof(char)*dim);
    for(i=0; i<dim; i++){
        array[i] = temp[i];
    }
    dim = move(shift, array, dim);
    printf("%d ", dim);
    while(dim>=0){
        printf("%c", array[dim]);
        dim--;
    }
    printf("\n");
    return 0;
}
```

ESERCIZIO 4 (2 punti)

Si presenti l'algoritmo di una funzione che realizza la ricerca binaria su un array di interi (eventualmente anche in forma di pseudo codice). Si discuta poi brevemente come estendere tale algoritmo a tipi di dati diversi dagli interi.

Soluzioni

ESERCIZIO 1

```
#include "list.h"
#include "element.h"

int maiuscole(char *s){
    int res = 0;
    while(*s != '\0'){
        if(*s>='A' && *s<='Z'){
            res++;
        }
        s++;
    }
    return res;
}

list atLeast(list l, int* v){
    if( empty(l) )
        return emptylist();
    else if ( maiuscole(head(l)) >= *v )
        return cons( head(l), atLeast(tail(l), v+1) );
    else
        return atLeast(tail(l), v+1 );
}

int main(){
    list l, res;
    int v[] = {4, 9, 3};
    l = cons("MARE", cons("SOle", cons("VAcaNZE", emptylist())));
    res= atLeast(l, v);
    while(!empty(res)){
        printf("%s\n", head(res));
        res = tail(res);
    }
}
```

ESERCIZIO 2

```
37    ->    00100101                    00100101+   (37)
+51   ->    00110011                    11001101=   (-51)
            11001100                    ----------------
-51   ->    11001101                    11001101    (-14)

                                        11001101
                                        00110010
                                        ----------------
                                        00001110 -> (+14)
```

ESERCIZIO 3

Il programma è corretto sintatticamente e la sua esecuzione produce la stampa:
`4  iiaa`
(tra '4' ed il primo carattere 'i' ci sono due spazi, in quanto il primo carattere stampato è un spazio)

La funzione `main()`calcola la lunghezza della stringa `temp`, alloca spazio sufficiente a contenere tutti i caratteri presenti in `temp`, li copia nello spazio di indirizzamento appena allocato (compreso il terminatore) ed invoca la funzione `move()`.

La funzione `move()` copia nell'array di caratteri `temp` gli elementi della stringa passata come parametro di ingresso; l'indice dell'elemento copiato dipende dal valore presente nell'array di interi `v` passato in ingresso. Se il valore dell'elemento dell'array di `v` è positivo, viene selezionato l'elemento di `in` con indice `v[i] % MAX`. Altrimenti il valore di `v[i]` viene incrementato di `dim` finché è negativo ed infine viene selezionato l'elemento di `in` con indice `v[i] % MAX`. Infine la funzione `move()` copia nell'array `in` i caratteri presenti in `temp` e restituisce il numero di caratteri copiati nel vettore `in`.

La funzione `main()` scrive sullo standard output il numero di caratteri modificati nell'array `shift` e poi li stampa in ordine inverso.