

Fondamenti di Informatica L-A (A.A. 2006/2007) - Ingegneria Informatica
Prof.ssa Mello & Prof. Bellavista – Prova d’Esame di Mercoledì 13 Dicembre 2006 – durata 2h
Compito B

ESERCIZIO 1 (9 punti)

Si consideri un vettore (di dimensione nota) contenente dei valori interi maggiori di 0 ripetuti più volte. Tale vettore ha anche la caratteristica che, detto **max** il valore massimo presente, contiene almeno una volta ogni valore tra 1 e **max** compresi. Il candidato realizzi una funzione:

```
int * ripetuti(int * v, int dim, int * newDim)
```

che, ricevuti come parametri il vettore **v** e la dimensione **dim** di tale vettore, restituisca come risultato un puntatore a **int**, che rappresenta un nuovo vettore allocato dinamicamente. Tale vettore deve contenere gli elementi di **v** senza ripetizioni, esattamente nell’ordine con cui compaiono per l’ultima volta nel vettore v. Il candidato abbia cura di allocare dinamicamente solo la memoria strettamente necessaria. Tramite poi il parametro **newDim**, la funzione restituisca anche la dimensione del vettore risultato.

Ad esempio, se **v=[3,3,1,6,4,5,1,2,3,5]**, il risultato deve essere un vettore di dimensione 6, della forma **[6,4,1,2,3,5]**.

ESERCIZIO 2 (9 punti)

Il candidato supponga di avere a disposizione, già definiti, l’ADT lista per interi (denominato **list**, con relative primitive) e un nuovo ADT, denominato **list_1**, che rappresenta il concetto usuale di lista, definita però per elementi che a loro volta sono liste di interi. Il candidato supponga di possedere le usuali primitive (denominate però **empty_1(...)**, **emptylist_1(...)**, **cons_1(...)**, **head_1(...)**, **tail_1(...)**) definite opportunamente per il nuovo ADT **list_1**. Il candidato definisca una funzione

```
list_1 subList(list l1)
```

che, ricevuta in ingresso una lista **l1** di interi, che rappresenta una collezione di sequenze di valori (ogni sequenza terminata dal valore 0), restituisca in uscita una nuova lista di liste, le cui singole sotto-liste corrispondono alle singole sottosequenze (in un qualunque ordine). Ad esempio, se invocata con parametro **l1=[4,2,3,0,1,13,2,9,0,1,2,0]**, la funzione deve restituire come risultato la lista di tre liste **[[4,2,3], [1,13,2,9], [1,2]]**. A tal scopo si utilizzino solo le operazioni primitive degli ADT **list** e **list_1**.

ESERCIZIO 3 (6 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>

typedef struct item {
    char value;
    struct item * next;
} Item;
typedef Item * list;

int dim = 5;

list duplicate(list l1) {
    list temp = l1, newItem;
    while (temp->next != NULL) temp = temp->next;
    newItem = (list) malloc(sizeof(Item));
    newItem->next = l1;
    newItem->value = temp->value;
    return newItem; }

int main(void) {
    list l1, temp;
    int i;
    l1 = (list) malloc(sizeof(Item));
    l1->next = NULL;
    l1->value = 'e';
    for (i=1; i<dim; i++) {
        temp = (list) malloc(sizeof(Item));
        temp->next = l1;
        temp->value = 'e' - i;
        l1 = temp;
    }
    l1 = duplicate(l1);
    while (l1 != NULL) {
        printf("%c ", l1->value);
        l1 = l1->next; }
    return (0); }
```

ESERCIZIO 4 (5 punti)

Data la funzione:

```
int x(int a, int b){
    if ( (a-b>1) && (b>=0) )
        return x(a-b, b-1) + 1 + x(a-b, b-2);
    else return 1;
}
```

mostrare la sequenza dei record di attivazione nel caso in cui la funzione sia invocata con parametri attuali (4,2).

ESERCIZIO 5 (3 punti)

Si consideri la rappresentazione collegata usata per l'implementazione dell'ADT stack vista durante il corso. Si riporti il codice che implementa, in tale rappresentazione, un nodo dello stack e il codice delle operazioni primitive `push(...)` e `pop()`.

SOLUZIONE

ESERCIZIO 1

```
int * ripetuti(int * v, int dim, int * newDim) {
    int * result;
    int i, j, pos=0, trovato=0;

    *newDim = 0;
    for (i=0; i<dim; i++)
        if (v[i] > *newDim) *newDim = v[i];
    result = (int*) malloc(sizeof(int)* *newDim);

    for (i=0; i<dim; i++) {
        for (j=i+1; j<dim && !trovato; j++)
            if (v[i] == v[j]) trovato++;
        if (!trovato) {
            result[pos] = v[i];
            pos++;
        }
        trovato = 0;
    }
    return result;
}
```

ESERCIZIO 2

```
list_l subList(list l1)
{
    list_l result = emptylist_l();
    list temp;

    while ( !empty(l1))
    {
        temp = emptylist();
        while ( head(l1) != 0)
        {
            temp = cons(head(l1), temp);
            l1 = tail(l1);
        }
        result = cons_l(temp, result);
        l1 = tail(l1);
    }
    return result;
}
```

ESERCIZIO 3

Il programma è corretto sintatticamente e la sua esecuzione produce la stampa:
e a b c d e

Nella funzione main() viene creata una lista di caratteri, contenenti nell'ordine gli elementi a,b,c,d,e. Viene invocata la funzione duplicate(...) e la lista risultante viene poi stampata a video.

La funzione `duplicate(...)` alloca memoria per un nuovo elemento e lo concatena in testa alla lista. Quindi in tale primo elemento viene copiato il valore presente nell'ultimo elemento della lista (determinato tramite il ciclo iniziale). La funzione restituisce come lista risultante il puntatore al nuovo elemento aggiunto.

ESERCIZIO 4

Nel caso di valutazione degli addendi da sinistra verso destra:

