Fondamenti di Informatica T-1 (A.A. 2010/2011) - Ingegneria Informatica
Prof.ssa Mello
Prova Parziale d'Esame di Giovedì 16 Giugno 2011 – durata 1h
Totale 12 punti, sufficienza con 7

ESERCIZIO 1 (6 punti)

Data una lista di char `listChar`, una stringa ben formata `string` ed un char `skip`, si realizzi una funzione ricorsiva

```
list merge(char* string, list listChar, char skip);
```

che restituisca una nuova lista di caratteri ordinati alfabeticamente contenente i caratteri presenti in `string` e `listChar`. Si noti che:

1) i caratteri presenti in `listChar` e `string` sono ordinati alfabeticamente;
2) la lista restituita NON deve contenere il carattere presente in `skip`.

Ad esempio se `listChar = ['a', 'b', 'd', 'g']`, `string = "cdehim"` e `skip = 'd'`, la funzione `merge()` deve restituire la lista `['a', 'b', 'c', 'e', 'g', 'h', 'i', 'm']`.

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

ESERCIZIO 2 (2 punti)

Si consideri la seguente funzione:

```
int fun(char ch, float fl){
      int diff = ch - 'a' + fl;
      if(diff < fl){
            return diff;
      }
      else{
            return diff + fun(ch-1, fl);
      }
}
```

Mostrare la sequenza dei record di attivazione ed il valore di ritorno nel caso in cui la funzione sia invocata con parametri attuali ('c', 1.9).

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>
#include <string.h>

float* avg(int* intArray){
	int dim = 0;
	float *res, *tempRes;
	int *temp;
	int count;

	temp = intArray;
	while( *temp > 0 ){
		dim++;
		temp++;
	}
	tempRes = res = (float*)malloc(sizeof(float)*(dim+1));

	while( *intArray > 0 ){
		dim = 0;
		count = 0;
		temp = intArray;
		while( temp[dim] > 0 ){
			count += temp[dim];
			dim++;
		}
		*tempRes = count/dim;
		tempRes++;
		intArray++;
	}
	*tempRes = -1;
	return res;
}

int main() {
	int v[] = {2, 9, 3, 2, -3, 4, 5};
	float* res = avg(v);
	while( *res > 0 ){
		printf("%f\n", *res);
		res++;
	}
	return 0;
}
```

ESERCIZIO 4 (1 punti)

Il candidato illustri brevemente cosa si intende e quali sono le differenze tra passaggio di parametri per valore e passaggio di parametri per riferimento.

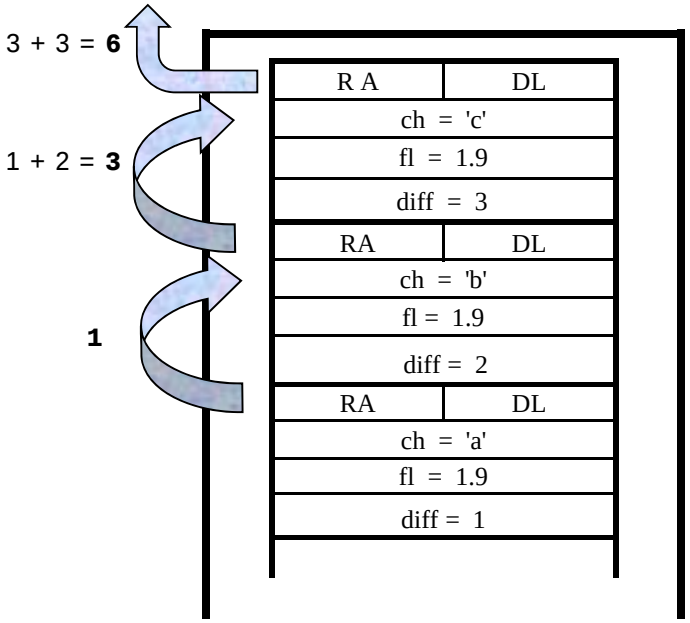
Soluzioni

ESERCIZIO 1

```
list merge(char* string, list listChar, char skip){

      if( *string!='\0' && *string == skip){
            return merge(string+1, listChar, skip);
      }
      else if( ! empty(listChar) && head(listChar) == skip){
            return merge(string, tail(listChar), skip);
      }

      if( *string == '\0' && empty(listChar) ){
            return emptylist();
      }
      else if( *string == '\0' ){
            return cons(head(listChar), merge(string, tail(listChar),skip));
      }
      else if (empty(listChar)){
            return cons(*string, merge(string+1, listChar,skip));
      }
      else if( *string < head(listChar) ){
            return cons(*string, merge(string+1, listChar,skip));
      }
      else{
            return cons(head(listChar), merge(string, tail(listChar),skip));
      }
}

int main(){
      list lista, res;
      char* stringa = "cdehim";
      char skipChar = 'd';
      lista = cons('a', cons('b', cons('d', cons('g', emptylist()))));
      res = merge(stringa, lista, skipChar);
      while( ! empty(res) ){
            printf("%c", head(res));
            res = tail(res);
      }
      printf("\n");
      return 0;
}
```

ESERCIZIO 2

La funzione restituisce il valore 6.



ESERCIZIO 3

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

```
4.00
4.00
2.00
2.00
```

La funzione `main()` inizializza l'array di int `v` e poi invoca la funzione `avg()`.

La funzione `avg()` alloca dinamicamente spazio di memoria sufficiente a contenere tanti float quanti sono gli elementi dell'array `intArray` fino al primo valore non positivo compreso. Successivamente il ciclo while itera sugli elementi dell'array `intArray` e, per ciascun valore, calcola la media della valore in considerazione e di quelli successivi fino al primo valore non positivo. Il risultato della media viene inserito nello spazio di memoria allocato dinamicamente. Da notare che la media viene troncata ad intero in quanto risultato di una divisione tra interi anche se poi successivamente viene salvato come float. Infine la funzione `avg()` aggiunge -1 come ultimo valore nello spazio di memoria allocato dinamicamente e restituisce al chiamante un riferimento alla prima cella di tale area.

La funzione `main()` scrive sullo standard output i valori presenti nell'area di memoria allocata dinamicamente restituita dalla funzione `avg()` fino al primo valore non positivo escluso.