Fondamenti di Informatica L-A (A.A. 2007/2008) - Ingegneria Informatica
Prof.ssa Mello & Prof. Bellavista – Prova d'Esame di Mercoledì 25 Giugno 2008 – durata 2h
Compito A

ESERCIZIO 1 (11 punti)

Un ufficio comunale vuole automatizzare la gestione delle multe notificate ogni anno ai cittadini per ritardi nel pagamento delle rate trimestrali delle tasse dei rifiuti; ciascuna multa deve essere memorizzata in un'opportuna struttura dati **`verbale`** composta dall'identificativo dell'utente (un **`int`**) e dall'entità della multa (un **`float`**). A tal fine il comune mantiene due elenchi separati. Il primo elenco contiene l'identificativo dei cittadini che hanno pagato in ritardo una rata nell'anno attuale; se un cittadino ha pagato più di una rata in ritardo, comunque il suo identificativo comparirà una sola volta. Il secondo elenco contiene l'identificativo dei cittadini recidivi che negli anni passati sono già stati multati a causa di ritardi nei pagamenti.

a) Il candidato realizzi una funzione:

```
verbale* calcolaMulta(int* idMultati, int dimMul, int* recidivi, int dimRec, int* firstRec);
```

che riceva come parametri un vettore **`idMultati`** contenente **`dimMul`** identificativi utente (rappresentante l'elenco dei cittadini che hanno pagato in ritardo una rata; ciascun identificativo può comparire al più una volta, come spiegato sopra) e un vettore **`recidivi`** contenente **`dimRec`** identificativi utente (rappresentante l'elenco dei cittadini già multati in passato).

La funzione deve calcolare l'entità della multa da verbalizzare a ciascun utente. L'entità della multa sarà di 20 euro nel caso in cui l'utente non sia recidivo (ovvero nel caso in cui sia il primo anno in cui paga rate in ritardo), sarà invece di 50 euro nel caso in cui l'utente sia recidivo. In particolare la funzione deve restituire un apposito vettore di strutture dati di tipo **`verbale`**, allocato dinamicamente, della dimensione strettamente necessaria per contenere un verbale per ciascun cittadino multato. Ciascun elemento di tale vettore deve contenere l'identificativo dell'utente e l'entità della multa verbalizzata, avendo premura di inserire nella prima parte del vettore tutti i verbali da 20 euro e nella seconda parte tutti i verbali da 50 euro. Infine la funzione deve restituire tramite **`firstRec`** l'indice del primo verbale da 50 euro.

b) Il candidato realizzi una semplice funzione **`main()`** di esempio che invochi correttamente la funzione **`calcolaMulta()`** e che stampi sullo standard output l'elenco dei soli verbali da 20 euro. A tal fine il candidato definisca due opportuni array per contenere l'elenco dei cittadini multati nell'anno attuale e l'elenco dei cittadini recidivi.

ESERCIZIO 2 (8 punti)

Si supponga di avere a disposizione, già definito, l'ADT lista per interi (denominato **`list`**, con relative primitive). Il candidato definisca una funzione *ricorsiva*

```
list noDuplicated (list l1, list l2)
```

che, ricevute in ingresso due liste **`l1`** ed **`l2`** di interi, restituisca una nuova lista contenente gli elementi della lista **`l1`** che non sono presenti nella lista **`l2`**.

ESERCIZIO 3 (7 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>

char* find(char* s1, char* s2){
      char *temp,*res;        res=NULL;         int i;
      while(*s1!='\0' && res==NULL){
            temp=s2;
            while(*temp!='\0' && *(temp+1)!='\0' && res==NULL){
                  if(*s1==*temp && *s1==*(temp+1)){
                        temp=s1;
                        for(i=1; *temp!='\0'; i++,temp++);
                        res=(char*)malloc( i*sizeof(char) );
                        temp=res;
                        while(*s1!='\0'){
                              *temp=*s1;
                              temp++;
                              s1++;
                        }
                        *temp='\0';
                  }
                  temp++;
            }
            if(res==NULL) s1++;
      }
      return res;
}

int main(){
      char *res=find("arti","soprattutto");
      if(res==NULL) printf("no match\n");
      else printf("%s\n",res);
      return 0;
}
```

ESERCIZIO 4 (3 punti)

Data la funzione:

```
      char res(int a, char c){
            if(a+c>'h') return c-1;
            else return res(a+2,c-1);
      }
```

mostrare la sequenza dei record di attivazione nel caso in cui la funzione sia invocata con parametri attuali (3.2, 'd').

ESERCIZIO 5 (3 punti)

Il candidato descriva brevemente le differenze tra la rappresentazione di un char in un file binario e in un file di testo. Inoltre, il candidato mostri tramite codice come sia possibile leggere un char dai due tipi di file.

ESERCIZIO 1

```
typedef struct{
	int idCittadino;
	float multa;
}verbale;

verbale* calcolaMulta(int* idMultati, int dimMul, int* recidivi, int dimRec, int* firstRec){
	int i, j;
	verbale *result;
	int first, last;

	result = (verbale*) malloc(dimMul * sizeof(verbale));
	first=0;
	last=dimMul-1;

	for(i=0; i<dimMul; i++){
		int giaMultato=0;
		for(j=0; j<dimRecidivi && !giaMultato; j++){
			if(idMultati[i]==recidivi[j]) giaMultato=1;
		}
		if(giaMultato) {
			result[last].idCittadino=idMultati[i];
			result[last].multa=50.0;
			last--;
		}
		else{
			result[first].idCittadino=idMultati[i];
			result[first].multa=20.0;
			first++;
		}
	}
	*firstRecidivo=last;
	return result;
}

int main(void){
	int i;
	int multati[]={2,8,4,3,5,10};
	int giaMultati[]={2,5};
	int indice;
	verbale* res;
	res=calcolaMulta(multati, 6, giaMultati, 2, &indice);
	for(i=0;i<indice;i++){
		printf("%d\t%f\n", res[i].idCittadino, res[i].multa);
	}
	return 0;
}
```

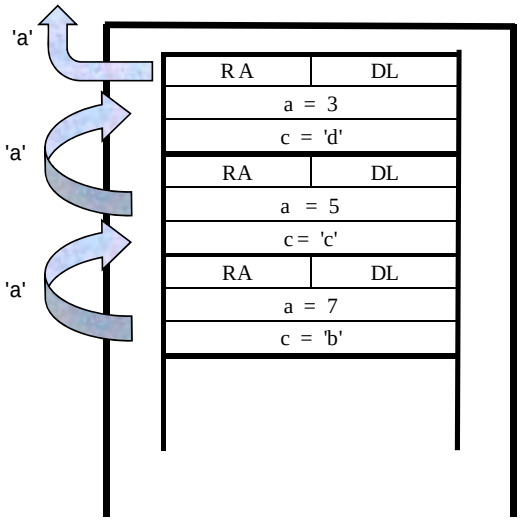
ESERCIZIO 2

```
list noDuplicated(list l1, list l2){
	if(empty(l1))return l1;
	else{
		list temp=l2;
		int trovato=0;
		while(!empty(temp) && !trovato){
			if(head(temp)==head(l1)) trovato=1;
			temp=tail(temp);
		}
		if(trovato) return noDuplicated(tail(l1),l2);
		else return cons(head(l1), noDuplicated(tail(l1),l2));
	}
}
```

ESERCIZIO 3

Il programma è corretto sintatticamente e la sua esecuzione produce la stampa:
`ti`
La funzione `main()` invoca la funzione `find()` con parametri di ingresso `"arti"` e `"soprattutto"`.
La funzione `find()` itera lungo la stringa `s1` cercando un carattere di `s1` che compaia due volte consecutivamente nella stringa `s2`. Considerando le due stringhe passate come parametri di ingresso, la condizione data è soddisfatta quando la funzione giunge al terzo carattere della stringa `"arti"`. A questo punto la funzione alloca spazio sufficiente a contenere tutti i caratteri rimanenti della stringa `s1` più il carattere di terminazione e copia in tale area di memoria i caratteri rimanenti di `s1`. Infine restituisce al chiamante un riferimento a tale area.
La funzione `main()` stampa sullo standard output la stringa restituita oppure un messaggio di errore qualora la stringa restituita non fosse valida.

ESERCIZIO 4



L'invocazione `res(3.2, 'd')` restituisce il carattere `'a'`.