Fondamenti di Informatica L-A (A.A. 2005/2006) - CdS Ingegneria Informatica
Prof.ssa Mello & Prof. Bellavista – I Prova Intermedia del 02/11/2005 - durata 2h30m
COMPITO C

ESERCIZIO 1 (12 punti)

Un comune vuole calcolare la velocità media che percorrono i veicoli in una strada nell'arco di una giornata. In particolare, vuole ottenere la velocità media di tutte le autovetture la cui targa, descritta con un numero a sei cifre, è superiore o uguale a 100000 e la velocità e la targa del mezzo più veloce, includendo anche gli autoveicoli la cui targa è inferiore a 100000.
A tal scopo si realizzi:

1) una funzione

   `float mediaVel(int vel[], long targhe[], int length, int* piuVeloce, long* piuVeloceTarga)`

   che noto il numero di autoveicoli **`length`**, le velocità **`vel[]`** e le targhe **`targhe[]`**, restituisca la velocità media come **`float`** (escludendo i veicoli la cui targa è inferiore a 100000**)** e tramite i parametri **`piuVeloce`** e **`piuVeloceTarga`** rispettivamente la velocità e la targa dell'autoveicolo più veloce, includendo anche i veicoli la cui targa è inferiore a 100000. Si assuma che la velocità in **`vel[0]`** corrisponda al veicolo con targa **`targhe[0]`**, la velocità in **`vel[1]`** al veicolo con targa **`targhe[1]`** e così via; **(7 punti)**

2) un programma **`main()`** che
   a. chieda all'utente il numero di veicoli **`V`** presi in esame e che controlli che **`V`** abbia valore tra 10 e 100 compresi. In caso contrario, si richieda nuovamente il numero di veicoli;
   b. chieda all'utente di inserire **`V`** velocità e targhe, controllando che le velocità inserite abbiano valori tra 0 e 120 compresi e le targhe valori tra 0 e 1000000 (estremi non inclusi). In caso contrario, si richieda nuovamente all'utente di inserire velocità e/o targa;
   c. richiami opportunamente la funzione **`mediaVel(...)`**;
   d. stampi la velocità media restituita da **`mediaVel(...)`**, la velocità e la targa dell'autoveicolo più veloce. **(5 punti)**

ESERCIZIO 2 (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>

#define DIM 4

void fun(float *x1, float* x2, int dim){
      int i; float temp;
      for(i=0;i<dim;i++){
            if(*(x1+i)<=*(x2+dim-1-i)){
                  temp=x1[i];
                  x1[i]=x2[dim-1-i];
                  x2[dim-1-i]=temp;
            }
      } return;
}

int main(){
      int i=0;
      float f1[DIM]={0.2,3.0,8.0,-17.0};
      float f2[DIM]={0.3,2.1,4.0,15};

      fun(f1,f2,DIM);
```

```
      for(;i<DIM;i++) printf("%f ",f1[i]);
      printf("\n");
      for(;i>0;i--) printf("%f ",f2[i-1]);
      printf("\n");

      return 0;
}
```

ESERCIZIO 3 (6 punti)

Si scriva una funzione iterativa **`int fun(char *a, char *b)`** che, ricevuti come parametri in ingresso due stringhe ben formate **`a`** e **`b`**, restituisca come valore di ritorno un **`int`** rappresentante la somma totale di occorrenze di ogni carattere di **`a`** in **`b`**. Ad esempio, la chiamata **`fun("Topolino", "TittiZz")`** deve restituire 3 (1 occorrenza di 'T', 2 occorrenze di 'i').
Si proponga una possibile funzione chiamante.

ESERCIZIO 4 (4 punti)

Data la funzione:

```
int fun(int a, int* b, int c,){
      if(c<=a){
            a=a-c;
            return 1+fun(a,b,c);
      }
      else { *b=a; return 0; }
}
```

e la funzione chiamante:

```
int main(){
      int r=0;
      fun(20,&r,7);
      return 0;
}
```

mostrare la sequenza dei record di attivazione.

ESERCIZIO 5 (2 punti)

Si consideri la grammatica G con scopo S e simboli terminali {1,2,3,4,a,b,c,x,y,z}

```
S::= AA | DBC
A::= CE | DCB
B::= EBD | D
C::= FC | FE
D::= 1 | 2 | 3 | 4
E::= a | b | c
F::= x | y | z
```

La stringa “xba1zyba24” appartiene alla grammatica? Se sì se ne mostri la derivazione left-most.

ESERCIZIO 6 (2 punti)

Date le definizioni: **`int x1[]={1,7,0,-3}, * x2;`**
indicare la quantità di memoria allocata dalle due variabili sullo stack. Inoltre, spiegare se e perché il seguente assegnamento risulta corretto/scorretto:

```
x1 = x2;
```

Soluzioni

Esercizio 1

```
#define DIM 100
float mediaVel(int vel[], long targhe[], int length, int* piuVeloce, long*
piuVeloceTarga){
	int i, somma=0, macchineInSomma=0;
	*piuVeloce=-1;
	*piuVeloceTarga=-1;

	for(i=0;i<length;i++){
		if(targhe[i]>=100000){
			somma=somma+vel[i];
			macchineInSomma++;
		}
		if(vel[i]>*piuVeloce){
			*piuVeloce=vel[i];
			*piuVeloceTarga=targhe[i];
		}
	}
	return ((float)somma)/((float)macchineInSomma);
}

int main(){
	int vel[DIM];
	long targhe[DIM];
	int piuVeloce, V, i;
	long piuVeloceTarga;
	float media;

	do {	printf("Numero veicoli?\n");
		scanf("%d",&V);
	} while(V<10||V>100);

	for(i=0;i<V;i++){
		do {	printf("Velocita' veicolo %d?\n",i);
			scanf("%d",&vel[i]);
		} while ((vel[i]<0)||(vel[i]>100));
		do {	printf("Targa veicolo %d?\n",i);
			scanf("%l",&targhe[i]);
		} while((targhe[i]<=0)||(targhe[i]>=1000000));
	}
	media=mediaVel(vel, targhe, V, &piuVeloce, &piuVeloceTarga);
	printf("media %f; più veloce %d con targa %l\n",media,piuVeloce,piuVeloceTarga);
	return 0;
}
```

Esercizio 2

15.0 4.0 8.0 0.3
0.2 3.0 2.1 -17.0

La funzione fun() scorre gli array x1 e x2; il primo dall'inizio alla fine, il secondo dalla fine all'inizio. Mentre scorre i due array, fun() confronta i valori di x1 ed x2; se il valore di x1 è minore o uguale a quello di x2, inverte i valori di x1 ed x2.
La funzione main() stampa l'array f1 modificato dall'inizio alla fine e l'array f2 modificato dalla fine all'inizio.

Esercizio 3

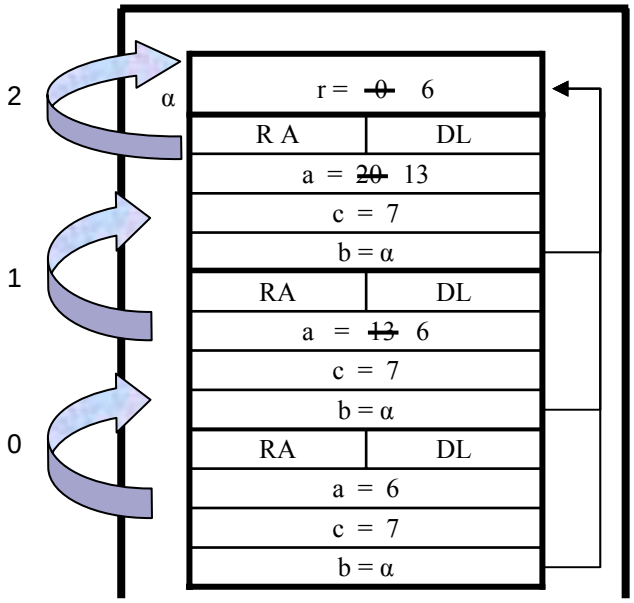
```
int fun(char* a, char* b){
	int totale=0, i=0, j;
	while(a[i]!='\0'){
		j=0;
		while(b[j]!='\0'){
```

```
                    if(a[i]==b[j]) totale++;
                    j++;
                }
                i++;
        }
        return totale;
}

int main(){
        int occorrenze=0;
        char a[]="abcdefghilmno";
        char b[]="aabbbde";
        occorrenze=fun(a,b);
        printf("Occorrenze %d\n",occorrenze);
        return 0;
}
```

Esercizio 4

La funzione effettua una divisione; come risultato restituisce 2 (risultato della divisione intera), r assume il valore 6 (resto della divisione intera).



Esercizio 5

S -> AA -> CEA -> FEEA -> xEEA -> xbEA -> xbaA -> xbaDCB -> xba1CB -> xba1FCB -> xba1zCB -> xba1zFEB -> xba1zyEB -> xba1zybB -> xba1zybEBD -> xba1zybaBD -> xba1zybaDD -> xba1zyba2D -> xba1zyba24

Esercizio 6

La variabile `x1` alloca lo spazio necessario a contenere 4 `int`.
La variabile `x2` alloca lo spazio necessario a contenere l'indirizzo di un `int`.
L'assegnamento `x1 = x2` è scorretto: anche se il tipo di dato associato a `x1` è uguale al tipo di dato associato a `x2` (per come gli array sono implementati in C), non è consentito cercare di variare il valore di `x1` (l'indirizzo denotato dal nome di un array è costante e non modificabile).