

Fondamenti di Informatica A-L – (Prof. P. Zingaretti)

PROVA SCRITTA – 10 dicembre 2004

Avvertenze:

- * Consegnare **solo fogli formato A4** e scrivere **su un solo lato** (no fronte-retro)
- * **In testa a ciascun foglio** scrivere: cognome, nome, numero progressivo di pagina rispetto al totale
esempio per il secondo foglio di 3 consegnati: Giuseppe Russo 2/3
- * Mantenere sul banco il **libretto o altro documento di riconoscimento** fino a controllo avvenuto
- * Nient'altro deve trovarsi sul banco: **non è consentito consultare libri, dispense, appunti, ecc.**
- * La **correzione** di riferimento per l'autovalutazione verrà effettuata in questa stessa aula alle ore **15:30**
- * La consegna delle **fotocopie** dei compiti avverrà al termine della correzione
- * La **prova orale** si terrà **venerdì 17 dicembre** alle ore **9:00** presso il D.I.I.G.A.



Si ricorda che chi si presenterà all'orale **DEVE** portare l'implementazione al computer della propria soluzione, eventualmente corretta, **corredata di tutto quanto necessario alla verifica** del corretto funzionamento.

1. (9 punti)

Si definisca una funzione C che restituisce la stringa ottenuta dalla fusione di due stringhe ordinate di lunghezza variabile, ricevute in input.

Esempio: "aaeirstt" e "abeet" devono fornire in uscita la stringa "aaabeeirstt".

2. (18 punti)

Si definisca una funzione C che **trasforma** la lista ordinata di interi ricevuta in input, nella rappresentazione collegata mediante puntatori, in una lista, sempre ordinata, contenente esattamente due elementi per ciascun valore intero diverso presente nella lista di input.

Ad esempio: la lista [1, 2, 2, 2, 2, 3, 6, 6, 6, 8, 8, 8] diventa [1, 1, 2, 2, 3, 3, 6, 6, 8, 8]

NB: per "trasforma" s'intende che **non** viene creata un'intera nuova lista, bensì occorre cancellare gli elementi superflui (nell'esempio due 2, un 6 e un 8) e occorre fare la malloc per gli elementi che vanno duplicati (nell'esempio 1 e 3).

3. (3 punti)

Si effettuino le seguenti dichiarazioni in linguaggio C:

A è un puntatore ad una funzione void con un array di interi come parametro

B è un array di 12 puntatori ad array di 5 double

C è un array di 20 stringhe

1. (9 punti)

```
char *merge_str(char *str_in1, char *str_in2) {
    int len=0, len1=0, len2=0;
    char *str_out;
    while (str_in1[len1]!='\0') len1++; //len=strlen(str_in1);
    while (str_in2[len2]!='\0') len2++; //len=strlen(str_in2);
    str_out=(char *)malloc(len1+len2+1); //len+EOLN
    len1=len2=0;
    while ((str_in1[len1]!='\0') && (str_in2[len2]!='\0'))
        if (str_in1[len1] < str_in2[len2])
            str_out[len++]=str_in1[len1++];
        else str_out[len++]=str_in2[len2++];
    while (str_in1[len1]!='\0') str_out[len++]=str_in1[len1++];
    while (str_in2[len2]!='\0') str_out[len++]=str_in2[len2++];
    str_out[len]='\0'; //termino nuova stringa
    return str_out;
}
```

2. (18 punti)

```
typedef struct list_tag {int val; struct list_tag *next;} list;

void trasforma(list *lp) {
    list *tmp;
    int rip=0, val;

    while (lp!=NULL) {
        tmp=lp->next; //salvo prossimo elemento da processare
        if (!tmp || (lp->val != tmp->val)) {
            if (!rip) { //duplico elemento
                val=lp->val; //salvo x duplicare elemento
                lp->next=(list *)malloc(sizeof(list)); //creo
                lp->next->val=val; //aggiorno
                lp->next->next=tmp; //ricollego
            } //else esistono gi... 2 elementi uguali
            else rip=0; //continuo con primo elemento diverso
        }
        else if (rip) { //esistevano già 2 elementi uguali
            lp->next=tmp->next; //salto elemento da cancellare
            free(tmp); //canello elemento
            tmp=lp; //aggiorno prossimo elemento da processare
        }
        else rip=1; //flag che esistono già 2 elementi uguali
        lp=tmp; //riposizionamento
    }
}
```

3. (3 punti)

```
void (*A)(int *, int);
double (*B[12])[5];
char *C[20];
```