

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

PROVA SCRITTA – 20 ottobre 2005

Avvertenze:

- * Consegnare **solo fogli formato A4 scritti 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 **13:45**
- * La consegna delle **fotocopie** dei compiti avverrà al termine della correzione
- * La **prova orale** si terrà **giovedì 27 ottobre** alle ore **10:30** 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. (15 punti)

Si definisca una funzione C che, dati due vettori di interi, entrambi ordinati ma di dimensioni qualsiasi, restituisce il vettore ordinato con tutti gli interi dei due vettori ricevuti in input.

Nota: se si usano dimensioni predefinite per i vettori solo 5 punti.

2. (15 punti)

Si scriva una funzione C che inserisce una stringa di lunghezza variabile su un lista ordinata in base al suo valore (somma dei codici ASCII dei caratteri costituenti la stringa stessa).

1. (12 punti)

```
int *merge(int *V1, int n1, int *V2, int n2) {
    int n=n1 + n2;
    int *V=(int *)malloc(n);
    while (n1 && n2) //esistono elementi non processati
        if (V1[n1-1] > V2[n2-1])
            V[--n]=V1[--n1];
        else V[--n]=V2[--n2];
    while (n1) V[--n]=V1[--n1];
    while (n2) V[--n]=V2[--n2];
    return V;
}
```

2. (18 punti)

```
typedef struct list_tag {    //3 punti
    char *str; int val; struct list_tag *next;
} list;

int str_value(char *str) {
    int value=0, i=0;
    while (str[i]!='\0') value+=str[i++];
    return value;
}

void insert(list **lpp, char *str) {
    int i, value=str_value(str);
    list *tmp_lp;
    while (*lpp!=NULL && str_value((*lpp)->str) < value)
        lpp=&((*lpp)->next);    //posizionamento
    //inserimento
    tmp_lp=*lpp;
    *lpp=(list *)malloc(sizeof(list));
    (*lpp)->str=(char *)malloc(sizeof(str)+1);
    for (i=0; i<=sizeof(str); i++) (*lpp)->str[i]=str[i];
    //oppure strcpy((*lpp)->str,str);
    (*lpp)->val=value;
    (*lpp)->next=tmp_lp;
}
```