

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

PROVA SCRITTA – 8 luglio 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 **12:00**
- * La consegna delle **fotocopie** dei compiti avverrà al termine della correzione
- * La **prova orale** si terrà **giovedì 15 luglio** 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. (12 punti)

Definire una funzione C che costruisce un **albero binario di ricerca** con gli elementi di un vettore di interi, provvedendo ad **eliminare i valori doppi** e restituendo il **numero di nodi** creati.

2. (18 punti)

Si definisca 'valore' di una stringa la somma dei codici ASCII dei caratteri costituenti la stringa stessa. Si scriva una funzione C che inserisce una stringa su una lista dinamica (**rappresentazione collegata mediante puntatore**) di **stringhe di lunghezza variabile** in modo ordinato in base al valore della stringhe.

1. (12 punti) (10 dicembre 2002)

Definire una funzione C che costruisce un albero binario di ricerca con gli elementi di un vettore di interi, provvedendo ad eliminare i valori doppi e restituendo il numero di nodi creati.

```
typedef enum {FALSE, TRUE} Boolean;
typedef struct btree_tag {
    int value;
    struct btree_tag *left;
    struct btree_tag *right;
} btree;

void ins_ord(int V[], int dim, btree **btpp, int *nnodi) {
    int i;
    for (i=0; i<dim; i++)
        *nnodi+=btree_ord_ins(btpp, V[i]);
}

Boolean btree_ord_ins(btree **btpp, int value) {
    while (*btpp!=NULL) //posizionamento
        if ((*btpp)->value == value) return FALSE;
        else if ((*btpp)->value > value)
            btpp=&((*btpp)->left);
        else btpp=&((*btpp)->right);
    //inserimento
    *btpp=(btree *)malloc(sizeof(btree));
    (*btpp)->value=value;
    (*btpp)->left=NULL;
    (*btpp)->right=NULL;
    return TRUE;
}
```

2. (18 punti) (18 marzo 2003)

Si scriva una funzione C che inserisce una stringa su una lista dinamica (rappresentazione collegata mediante puntatore) di stringhe di lunghezza variabile in modo ordinato in base al valore della stringhe (=somma dei codici ASCII dei caratteri costituenti la stringa).

```
typedef struct list_tag { char *str; 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 ins_ordI(list **lpp, char *str) {
    list *tmp_lp;
    int i, value=str_value(str);

    while (*lpp!=NULL && str_value((*lpp)->str) < value)
        lpp=&((*lpp)->next); //posizionamento
    //inserimento
    tmp_lp=*lpp; //salvo resto della lista
    *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)->next=tmp_lp; //ricollego
}
```