

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

PROVA SCRITTA – 2 ottobre 2003

Avvertenze:

- * Consegnare **solo fogli formato A4**
- * **Scrivere su un solo lato** (no fronte-retro)
- * In ordine di preferenza usare: 1) **inchiostro nero**; 2) matita; 3) inchiostro rosso; 4) inchiostro blu
- * **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 **10:15**
- * La consegna delle **fotocopie** dei compiti avverrà presso l'Istituto di Informatica al termine della correzione
- * **Prova orale giovedì 9 ottobre** alle ore **9:00** c/o il Dip. Ing. Informatica, Gestionale e dell'Automazione.



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. (18 punti)

Si definisca una funzione C che costruisce un albero binario **bilanciato** con gli elementi di un **vettore di interi**.

2. (9 punti)

Si definisca una funzione C che verifica se una lista **ordinata** di interi contiene **almeno tre** elementi con lo stesso valore.

3. (3 punti)

Si dichiarino le seguenti variabili nel linguaggio C:

A è un array di 12 puntatori a intero

B è un array di 12 arrays di 3 puntatori a intero

C è un puntatore a puntatore a intero

1. (18 punti)

```
typedef struct btree_tag {
    int value; struct btree_tag *lx; struct btree_tag *rx;
} btree;

int get_nodi(btree *root) {
    if (root_ptr == NULL) return 0;
    else return 1 + get_nodi(root->lx) + get_nodi(root->rx);
}

void insert(btree ** root_pp, int value) {
    int left, right;
    while(*root_pp!=NULL) {
        left = get_nodi((*root_pp)->lx);
        right = get_nodi((*root_pp)->rx);
        if (left > right) root_pp = &((*root_pp)->rx);
        else root_pp = &((*root_pp)->lx);
    }
    *root_pp = (btree *)malloc(sizeof(btree));
    (*root_pp)->lx = NULL;
    (*root_pp)->rx = NULL;
    (*root_pp)->value = value;
}

void ins_bilanciato(int V[], int dim, btree **btp) {
    int i;
    for (i=0; i<dim; i++) btree_ord_ins(btp, V[i]);
}
```

2. (9 punti)

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

int search_copies( list * head) {
    list * ptr;
    int i=1;
    if (head == NULL) return 0;
    else ptr=head->next;
    while (head && ptr) {
        if (ptr->value == head->value) {
            i++;
            if (i == 3) return 1; //TRUE
        } else i=1; //reinizializzo
        ptr=ptr->next;
        head=head->next;
    }
    return 0;
}
```

3. (3 punti)

```
int *A[12];
int *B[12][3];
int **C;
```