

## Fondamenti di Informatica I

PROVA SCRITTA – 3 luglio 2002

### 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
- \* La **correzione** di riferimento per l'autovalutazione verrà effettuata in questa stessa aula alle ore **00:00**
- \* La consegna delle **fotocopie** dei compiti avverrà presso l'Istituto di Informatica al termine della correzione
- \* La **prova orale** si terrà **lunedì 8 luglio** alle ore **9:30** presso l'Istituto di Informatica



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. (8 punti)

Si definisca la funzione C che ordina un vettore di dimensione assegnata usando l'algoritmo di mergesort.

### 2. (8 punti)

Sia dato un insieme di numeri float rappresentato su una lista. L'insieme non contiene istanze multiple di uno stesso valore. Si definisca la funzione C che, ricevuto un valore, lo inserisce nell'insieme se il valore non è già presente. La funzione deve restituire FALSE se il valore da inserire era già presente nell'insieme e TRUE se invece non era presente.

### 3. (8 punti)

Si definisca la funzione C che restituisce in un vettore, di dimensione pari alle lettere dell'alfabeto, la posizione (indice) in cui ciascuna lettera dell'alfabeto è stata incontrata nel file di testo ricevuto in input (se la lettera non esiste nel file mettere -1).

Nota: usare la seguente funzione di libreria per ritornare all'inizio del file: void rewind(FILE \*file);

### 4. (6 punti)

Si descrivano le seguenti dichiarazioni di variabili e funzioni (prototipi) in linguaggio C:

```
int (*A[3]) [5];
struct { int a[5]; char *b; } B[30];
int * (* C) [3];
char * (* D [ ]) ();
```

## Esercizio 1

```
void mergesort(int *V, int size) {
    int *AUX;
    AUX=(int*)malloc(size*sizeof(int));
    r_mergesort(V,AUX,first,size)
}

void r_mergesort(int * V, int * AUX, int first, int size)
{
    if (size>1) {
        r_mergesort(V,AUX,first,size/2);
        r_mergesort(V,AUX,first+size/2,size/2)
        merge(V,AUX,first,size);
    }
}

void merge(int * V, int * AUX, int first, int size) {
    int count, left=0, right=0;

    for (count=0; count<size/2; count++)
AUX[first+count]=V[first+count];
    while (left<size/2)
        if (right<size/2) {
            if (V[first+left]<AUX[first+size/2+right]) {
                V[first+left+right]=AUX[first+left];
                left++;
            } else {
                V[first+left+right]=V[first+size/2+right];
                right++;
            }
        } else {
            V[first+left+right]=AUX[first+left];
            left++;
        }
}
```

### Esercizio 2

```
struct list { float value; struct list *next_ptr; };

Boolean insert(struct list **list_ptrptr, float value) {
    Boolean not_found=TRUE;

    while(*list_ptrptr!=NULL && not_found==TRUE)
        list_ptrptr=&(*list_ptrptr)->next_ptr;
    if(not_found==TRUE) {
        *list_ptrptr=(struct list *)malloc(sizeof(struct list));
        (*list_ptrptr)->value=value;
        (*list_ptrptr)->next_ptr=NULL;
    }
    return not_found;
}
```

### Esercizio 3

```
void indice(char *file_in, int *v) {
    FILE *in;
    int ch, ch_L='a', ch_H='A';
    int i, index;

    if ((in=fopen(file_in, "rt"))!=NULL) { //apertura file ok
        for (i='a'; i<='z'; i++) {
            index=0;
            while ((ch=fgetc(in)) != EOF && ch!=ch_L && ch!=ch_H)
                index++;
            v[i-'a']=feof(in) ? -1 : index;
            ch_low++, ch_high++;
            rewind(in);
        }
        fclose(in);
    }
}
```

### Esercizio 4

- A e' un array di 3 puntatori ad array di 5 interi
- B e' un array di 30 record (strutture) composti da un array di 5 interi e un puntatore a caratteri
- C è un puntatore a un array di 3 puntatori a interi
- D e' un array di puntatori a funzioni che ritornano un puntatore a caratteri