

Lezione 9

- Scrivete le seguenti funzioni:

- `strcat(s,t)` che copia la stringa `t` al termine della stringa `s` (supponete che la lunghezza del vettore di caratteri che contiene `s` sia sufficiente a contenere la concatenazione di `s` e `t`).

```
char *strcat(char *s, char *t) {
    int i, j;

    for(i=0;*(s+i)!='\0';i++);

    for(j=0;*(t+j)!='\0';j++)
        *(s+i+j)=*(t+j);

    *(s+i+j)='\0';
    return s; /* restituisce il puntatore alla stringa concatenata
               * in modo da poter utilizzare strcat direttamente
               * in una chiamata a printf in corrispondenza dello
               * specificatore di formato %s
               */
}
```

- `reverse(s)` che inverte la stringa `s` (e.g. `abc` \rightsquigarrow `cba`).

```
char *reverse(char *s) {
    int i, l=strlen(s);
    char c;

    for(i=0; i<l/2; i++) {
        c=*(s+i);
        *(s+i)=*(s+l-(i+1));
        *(s+l-(i+1))=c;
    }

    return s;
}
```

- Scrivete un programma `sort` che, se chiamato senza argomenti sulla linea di comando, ordina le linee di un testo in ordine alfabetico, se chiamato con l'opzione `-r`, le ordina in senso inverso.

Suggerimento: si sfrutti la funzione di libreria `strcmp` (il prototipo è specificato nel file header `string.h`).

```
#include <stdio.h>
#include <string.h>

#define MAXLINES 1000
#define MAXLENGTH 1000

int getline(char s[], int lim) {
```

```

int i=0;
char c;

while((c=getchar())!='\n' && c!=EOF && i<lim-1)
    s[i++]=c;

s[i]='\0';
return i;
}

void merge(char a[MAXLINES][MAXLENGTH],int lo,int hi,int order) {
    int i,j,k,m,n=hi-lo+1;
    char b[MAXLINES][MAXLENGTH];

    k=0;
    m=(lo+hi)/2;

    for(i=lo;i<=m;i++)
        strcpy(b[k++],a[i]);

    for(j=hi;j>=m+1;j--)
        strcpy(b[k++],a[j]);

    i=0; j=n-1; k=lo;

    while(i<=j)
        if(order ? strcmp(b[i],b[j])>0 : strcmp(b[i],b[j])<0)
            strcpy(a[k++],b[i++]);
        else
            strcpy(a[k++],b[j--]);
}

void mergesort(char a[MAXLINES][MAXLENGTH],int lo,int hi,int order) {
    int m;

    if (lo<hi) {
        m=(lo+hi)/2;
        mergesort(a,lo,m,order);
        mergesort(a,m+1,hi,order);
        merge(a,lo,hi,order);
    }
}

main(int argc,char **argv) {
    char linee[MAXLINES][MAXLENGTH];
    int i=0,j,order=0;

    while(getline(linee[i],MAXLENGTH)!=0 && i<MAXLINES)

```

```

    i++;

    if(argc==2 && strcmp(argv[1],"-r")==0)
        order=1;

    mergesort(linee,0,i-1,order);

    for(j=0;j<i;j++)
        printf("%s\n",linee[j]);
}

```

Nell'esempio di soluzione sopra riportato si è fatto uso del *mergesort* (funzioni *merge* e *mergesort*) per l'ordinamento delle linee; nulla vieta tuttavia di utilizzare un qualunque altro algoritmo di ordinamento.

- Scrivete un programma che legga dallo standard input dei numeri e ne stampi la somma totale.

```

#include <stdio.h>

main() {
    int n,totale=0;
    printf("Inserisci un numero intero per linea, terminando "
           "la sequenza con Ctrl-D (su una linea da solo)\n");

    while(scanf("%d",&n)!=EOF)
        totale+=n;

    printf("Totale: %d\n",totale);
}

```