```c
/* un secondo esempio di uso di liste: qui elimino le parole ripetute */

#include <stdio.h>
#include <math.h>

#define MY_MAX 100

#define SUCCESSO 0
#define ROVINA -9

FILE *f_input;
char *f_nome_input= "leopardi2.dat";

int my_end;
char my_string[MY_MAX];

void my_start(void);
void apri_file(void);
void leggi_una_parola(void);
void costruisci_lista_inversa(void);
void stampa_lista_inversa(void);

/* creo la lista e visto che all'inizio e' vuoto la inizializzo a NULL */
struct lista_parole{
  char *p_stringa;
  struct lista_parole *prossima_parola;
} *lista_parole_1=NULL;

/*************************************************************
**/
int main(void)
```

```
{
  int i;

  my_start();
  apri_file();

  my_end=0;
  for(i=0;;i++){
    leggi_una_parola();
    costruisci_lista_inversa();
    if(my_end==1)break;
  }
  fclose(f_input);
  stampa_lista_inversa();
}

/*****************************************************************
**/
void my_start(void)
{
  printf("# Costruiremo una lista inversa senza ripetizioni\n");
}

/*****************************************************************
**/
void apri_file(void)
{
  f_input = fopen(f_nome_input,"r");
  if(f_input==NULL){
    printf("abort: impossibile aprire il file di input in lettura: %s\n",
           f_nome_input);
    exit(-9);
```

```
  }
  printf("# E' stato aperto il file di input: %s\n",
         f_nome_input);
}

/*****************************************************************
**/
void leggi_una_parola(void)
{
  int j;
  for(j=0; j<MY_MAX; j++){
    my_string[j] = fgetc(f_input);
    if(my_string[j]=='\n'){j--; continue;}/*per ignorare il fine riga*/
    if(my_string[j]==EOF){
      my_end=1;
      my_string[j]='\0';
      break;}
    if(my_string[j]==' '){
      my_string[j]='\0';
      break;
    }
  }
  /* per sicurezza mi sto riservando un carattere in piu' oltre
    al terminatore \0 */
  if(j>=MY_MAX-1){
    printf("Interruzione del programma: la parola era troppo lunga.\n");
    printf("Ricompilare con un valore di MY_MAX piu grande, era %d.\n",
           MY_MAX);
    exit(-9);
  }
  printf("Parola di lunghezza %d: %s\n",strlen(my_string),my_string);
}
```

```
/*****************************************************************
**/
void costruisci_lista_inversa(void)
{
  char *p_scratch;
  struct lista_parole *sp_scratch;
  struct lista_parole *sp_check;

  p_scratch = (char *)malloc(strlen(my_string)+1);
  if(p_scratch==NULL){
    printf("Interruzione del programma: fallita malloc 1 in c_l_i\n");
    exit(-9);
  }

  /* Qui controlliamo che la parola sia nuova */

  for(sp_check=lista_parole_1; sp_check!=NULL;){
    //printf("CHECKING... %s\n",sp_check->p_stringa);fflush(stdout);
    //printf("%d\n",strcmp(sp_check->p_stringa,my_string));
    if(strcmp(sp_check->p_stringa,my_string)==0){
      return;
    }
    sp_check = sp_check->prossima_parola;
  }

  strcpy(p_scratch,my_string);
  sp_scratch = lista_parole_1;
  lista_parole_1 =
    (struct lista_parole *) malloc(sizeof(struct lista_parole));
  if(lista_parole_1==NULL){
    printf("Interruzione del programma: fallita malloc 2 in c_l_i\n");
```

```
    exit(-9);
  }

  lista_parole_1->p_stringa = p_scratch;
  lista_parole_1->prossima_parola = sp_scratch;
}

/*****************************************************************
**/
void stampa_lista_inversa(void)
{
  for(;lista_parole_1!=NULL;){
    printf("%s\n",lista_parole_1->p_stringa);
    lista_parole_1 = lista_parole_1->prossima_parola;
  }
}
```