```
----------------------------------------
buffer
----------------------------------------
#include <stdio.h>
#include <string.h>
#include <ctype.h>

int main(int argc, char * argv[])
{
    char buffer[100];
    int i;
    strcpy(buffer,"stringa di prova");
    printf("buffer prima = %s\n",buffer);
    for(i=0; *(buffer+i)!=0; i++)
        *(buffer+i) = toupper(*(buffer+i));
    printf("buffer dopo = %s\n",buffer);
}
----------------------------------------------
buffer2
----------------------------------------------
#include <stdio.h>
#include <string.h>
#include <ctype.h>

int main(int argc, char * argv[])
{
    char buffer[100];
    int i;
    strcpy(buffer,"STRINGA DI PROVA");
    printf("buffer prima = %s\n",buffer);
    for(i=0; *(buffer+i)!=0; i++)
        if(*(buffer+i) == 'A') *(buffer+i) = 'Z';
    printf("buffer dopo = %s\n",buffer);
}
```

```
-----------------------------------------------------------
cercaA
-----------------------------------------------------------
#include <stdio.h>
#include <string.h>
#include <ctype.h>

int cercaA_iterativa(char * stringa)
{
    int i, count=0;
    for(i=0; stringa[i]!='\0'; i++)
      if(stringa[i]=='A')return 1;
    return 0;
}

int cercaA_ricorsiva(char * stringa)
{
    if(stringa[0]=='\0')return 0;
    else if(stringa[0]=='A') return 1;
    else return cercaA_ricorsiva(stringa+1);
}

int main(int argc, char * argv[])
{
    char buffer[100];
    int i;
    strcpy(buffer,"STRINGA DI PROVA");
    printf("esito con algoritmo iterativo=%d\n",cercaA_iterativa(buffer));

    strcpy(buffer,"STRINGA DI PROVA");
    printf("esito con algoritmo ricorsivo=%d\n",cercaA_ricorsiva(buffer));
}
```

```
-------------------------------------------------------------
esercizio FILES
-------------------------------------------------------------
#include <stdio.h>

int main(int argc, char * argv[])
{
  FILE * infile;
  FILE * outfile;
  int count=1;
  char c;

  if(argc!=3)
    return 0;

  infile=fopen(argv[1],"r");
  if(infile == NULL)
    return 0;

  outfile=fopen(argv[2],"w");
  if(outfile == NULL)
    return 0;

  while(!feof(infile))
  {
    c=fgetc(infile);
    if(count % 2 == 1)
        fputc(c, outfile);
    count ++;
  }

  fclose(infile);
  fclose(outfile);
}
```

```
-------------------------------------------------------------
liste tema A
-----------------------------------------------
#include <stdio.h>
#include <string.h>
#include <ctype.h>
#include <stdlib.h>

struct nodo_s {
char nome[30];
char cognome[30];
char indirizzo[50];
int eta;
struct nodo_s * next;
struct nodo_s * prev;
};

struct nodo_s * head=NULL;
struct nodo_s * tail=NULL;

void stampatutto()
{
    struct nodo_s * scan=head;

    for(; scan!=NULL; scan=scan->next)
        printf("%s %s, %d, %s\n", scan->nome, scan->cognome, scan->eta, scan-
>indirizzo);
}

int main(int argc, char * argv[])
{
    int x;
    char y;

    struct nodo_s * bianchi=(struct nodo_s *)malloc(sizeof(struct nodo_s));
    struct nodo_s * gialli=(struct nodo_s *)malloc(sizeof(struct nodo_s));
    struct nodo_s * neri=(struct nodo_s *)malloc(sizeof(struct nodo_s));
    struct nodo_s * rossi=(struct nodo_s *)malloc(sizeof(struct nodo_s));

    strcpy(bianchi->nome, "MARIO");
    strcpy(bianchi->cognome, "BIANCHI");
    strcpy(bianchi->indirizzo, "VIA BIANCHI 1");
    bianchi->eta=40;

    strcpy(gialli->nome, "PAOLO");
    strcpy(gialli->cognome, "GIALLI");
    strcpy(gialli->indirizzo, "CORSO GIALLI 1");
    gialli->eta=41;

    strcpy(neri->nome, "ANTONIO");
    strcpy(neri->cognome, "NERI");
    strcpy(neri->indirizzo, "LARGO NERI 1");
    neri->eta=42;

    strcpy(rossi->nome, "GIORGIO");
    strcpy(rossi->cognome, "ROSSI");
    strcpy(rossi->indirizzo, "PIAZZA ROSSI 1");
    rossi->eta=43;

    head=bianchi;
    bianchi->next=gialli;
    bianchi->prev=NULL;
```

```
    gialli->next=neri;
    gialli->prev=bianchi;
    neri->next=rossi;
    neri->prev=gialli;
    rossi->next=NULL;
    rossi->prev=neri;
    tail=rossi;

    stampatutto();

    /* DOMANDE TEMA A */
    /* scrivere un'istruzione che assegna a x il valore dell'età di Neri.
    Non usare il puntatore head. (scandire la lista dalla coda)  */
    x=tail->prev->eta;
    printf("Eta' di Neri=%d\n",x);

    /* scrivere un'istruzione che assegna a x la lunghezza della stringa
    contenente l'indirizzo di Rossi. Non usare il puntatore tail.
    (scandire la lista dalla testa) */
    x=strlen(head->next->next->next->indirizzo);
    printf("Lunghezza dell'indirizzo di Rossi=%d\n",x);

    /* scrivere un'istruzione che assegna a y il terzo carattere del nome di
Bianchi.
    (si assuma che tale nome sia sicuramente lungo almeno 3 caratteri)  */
    y=head->nome[2];
    printf("Terzo carattere del nome di Bianchi = '%c'\n",y);


    struct nodo_s * arrigoni;
    arrigoni=(struct nodo_s *)malloc(sizeof(struct nodo_s));
    strcpy(arrigoni->nome,"ERNESTO");
    strcpy(arrigoni->cognome,"ARRIGONI");
    strcpy(arrigoni->indirizzo,"CORSO ARRIGONI, 8");
    arrigoni->eta=45;
    /* supponendo che sia già stato creato (allocato e compilato) un nodo
    per il sig. Arrigoni, raggiungibile mediante il puntatore nuovonodo,
    scrivere le istruzioni necessarie per inserire tale nodo
    in prima posizione nella lista.  */

    head->prev=arrigoni;
    arrigoni->next=head;
    head=arrigoni;
    arrigoni->prev=NULL;

    stampatutto();

}
```

```
-------------------------------------------------------------
liste tema B
------------------------------
#include <stdio.h>
#include <string.h>
#include <ctype.h>
#include <stdlib.h>

struct nodo_s {
char nome[30];
char cognome[30];
char indirizzo[50];
int eta;
struct nodo_s * next;
struct nodo_s * prev;
};

struct nodo_s * head=NULL;
struct nodo_s * tail=NULL;

void stampatutto()
{
    struct nodo_s * scan=head;

    for(; scan!=NULL; scan=scan->next)
        printf("%s %s, %d, %s\n", scan->nome, scan->cognome, scan->eta, scan-
>indirizzo);
}

int main(int argc, char * argv[])
{
    int x;
    char y;

    struct nodo_s * bianchi=(struct nodo_s *)malloc(sizeof(struct nodo_s));
    struct nodo_s * gialli=(struct nodo_s *)malloc(sizeof(struct nodo_s));
    struct nodo_s * neri=(struct nodo_s *)malloc(sizeof(struct nodo_s));
    struct nodo_s * rossi=(struct nodo_s *)malloc(sizeof(struct nodo_s));

    strcpy(bianchi->nome, "MARIO");
    strcpy(bianchi->cognome, "BIANCHI");
    strcpy(bianchi->indirizzo, "VIA BIANCHI 1");
    bianchi->eta=40;

    strcpy(gialli->nome, "PAOLO");
    strcpy(gialli->cognome, "GIALLI");
    strcpy(gialli->indirizzo, "CORSO GIALLI 1");
    gialli->eta=41;

    strcpy(neri->nome, "ANTONIO");
    strcpy(neri->cognome, "NERI");
    strcpy(neri->indirizzo, "LARGO NERI 1");
    neri->eta=42;

    strcpy(rossi->nome, "GIORGIO");
    strcpy(rossi->cognome, "ROSSI");
    strcpy(rossi->indirizzo, "PIAZZA ROSSI 1");
    rossi->eta=43;

    head=bianchi;
    bianchi->next=gialli;
```

```
    bianchi->prev=NULL;
    gialli->next=neri;
    gialli->prev=bianchi;
    neri->next=rossi;
    neri->prev=gialli;
    rossi->next=NULL;
    rossi->prev=neri;
    tail=rossi;

    stampatutto();

    /* DOMANDE TEMA B */
    /* scrivere un'istruzione che assegna a x il valore dell'età di Rossi. Non
usare il puntatore tail. (scandire la lista dalla testa) */
    x=head->next->next->next->eta;
    printf("Eta' di Rossi=%d\n",x);

    /* scrivere un'istruzione che assegna a x la lunghezza del nome di Gialli.
Non usare il puntatore head. (scandire la lista dalla coda) */

    x=strlen(tail->prev->prev->nome);
    printf("Lunghezza del nome di Gialli=%d\n",x);

    /* scrivere un'istruzione che assegna a y il secondo carattere
dell'indirizzo di Neri. (si assuma che tale indirizzo sia sicuramente lungo
almeno 2 caratteri) */
    y=head->next->next->indirizzo[1];
    printf("Secondo carattere dell'indirizzo di Neri = '%c'\n",y);


    struct nodo_s * zefiro;
    zefiro=(struct nodo_s *)malloc(sizeof(struct nodo_s));
    strcpy(zefiro->nome,"CARLO");
    strcpy(zefiro->cognome,"ZEFIRO");
    strcpy(zefiro->indirizzo,"CORSO ZEFIRO, 10");
    zefiro->eta=44;
    /* supponendo che sia già stato creato (allocato e compilato) un nodo per il
sig. Zefiro,
    raggiungibile mediante il puntatore nuovonodo, scrivere le istruzioni
necessarie
    per inserire tale nodo in ultima posizione nella lista. */

    tail->next=zefiro;
    zefiro->prev=tail;
    tail=zefiro;
    zefiro->next=NULL;

    stampatutto();

}

-----------------------------------------------------------------
strlen
-----------------------------------------------------------------
#include <stdio.h>
#include <string.h>
#include <ctype.h>

int mystrlen_iterativa(char * stringa)
{
    int i;
    for(i=0; stringa[i]!='\0'; i++);
    return i;
```

```
}

int mystrlen_ricorsiva(char * stringa)
{
    if(stringa[0]=='\0')return 0;
    else return 1+mystrlen_ricorsiva(stringa+1);
}

int main(int argc, char * argv[])
{
    char buffer[100];
    int i;
    strcpy(buffer,"stringa di prova");
    printf("Lunghezza calcolata con algoritmo iterativo=%d\n",
        mystrlen_iterativa(buffer));
    printf("Lunghezza calcolata con algoritmo ricorsivo=%d\n",
        mystrlen_ricorsiva(buffer));
}
```