LIA - DEIS - Facoltà di Ingegneria - Università di Bologna

Elementi di informatica LA

Corso di laurea in Ingegneria Elettrica - Anno Accademico 2007-2008

Progetti su più File Array e Funzioni

ESERCIZIO 1

Scrivere un modulo in C, vet_io.c che:

- contenga una procedura leggi_ivet che legge da tastiera un vettore di interi
- contenga una procedura stampa_ivet che stampa su video un vettore di interi
- contenga una procedura leggi_dvet che legge da tastiera un vettore di double
- contenga una procedura stampa_dvet che stampa su video un vettore di double

Scrivere inoltre un main di prova che legge un vettore di interi *lambda[]* ed uno di reali *v[]* e li visualizza. Deve inoltre definire un un terzo vettore di reali, *alfa[]*,i cui elementi sono il prodotto degli elementi omologhi dei vettori *lambda[]* e *v[]* . Il programma visualizzi infine il vettore *alfa[]*. Si strutturi il programma come progetto costituito da moduli su più file.

Soluzione:

Il progetto si compone di tre file:

- *vet_io.c*: file che contiene la definizione delle procedure
- *vet_io.h*: header file del modulo *vet_io.c*
- *scalare.c*: file contenente il main.

vet_io.c

```
#include <stdio.h>

void leggi_ivet(int v[], int n)
{
 int i;
 for (i=0; i<n; i++) {
  printf("v[%d] = ",i);
  scanf("%d",&v[i]);
 }
```

```
}

void stampa_ivet(int v[], int n)
{
 int i;
 for (i=0; i<n; i++) printf("%d ",v[i]);
 printf("\n");
}


void leggi_dvet(double v[], int n)
{
 int i;
 for (i=0; i<n; i++) {
  printf("v[%d] = ",i);
  scanf("%lf",&v[i]);
 }
}

void stampa_dvet(double v[], int n)
{
 int i;
 for (i=0; i<n; i++) printf("%lf ",v[i]);
 printf("\n");
}
```

vet_io.h

```
void leggi_ivet(int v[], int n);

void stampa_ivet(int v[], int n);

void leggi_dvet(double v[], int n);

void stampa_dvet(double v[], int n);
```

scalare.c

```
#include "vet_io.h" /* vet_io.h e' nella stessa directory del main
*/

#define N 5

main()
{
 int lambda[N];
 double v[N],alfa[N];
 int i;
```

```
 leggi_ivet(lambda,N);
 stampa_ivet(lambda,N);
 leggi_dvet(v,N);
 stampa_dvet(v,N);

 for (i=0; i<N; i++) alfa[i] = lambda[i]*v[i];

 stampa_dvet(alfa,N);
}
```

ESERCIZIO 2

```
Si scriva un progetto su più file in linguaggio C, che Legga una sequenza di
caratteri in ingresso con al centro un '.' e
verifica se e` palindroma, ignorando gli spazi bianchi.

#include <stdio.h>


int palindroma(void)
  /* Verifica se una sequenza di caratteri con '.' al centro e` palindroma */
{
  char prima, dopo;
  int pal;             /* risultato della verifica sulla sequenza di caratteri
                          tranne il primo e l'ultimo */

  do {                          /* ciclo per saltare spazi bianchi e a capo */
    prima = getchar();
  } while (prima == ' ' || prima == '\n');

  if (prima == '.')
    return 1;                   /* la sequenza "." e` palindroma */
  else {
    pal = palindroma();         /* verifica se la sequenza di caratteri
                                   tranne il primo e l'ultimo e` palindroma */

    do {                        /* ciclo per saltare spazi bianchi e a capo */
      dopo = getchar();
    } while (dopo == ' ' || dopo == '\n');

    return pal && prima == dopo;
  }
}  /* palindroma */


int main(void)
{
  printf("Immetti una sequenza di caratteri (con un '.' centrale)!\n");

  if (palindroma())
    printf("E` palindroma\n");
  else
```

```
    printf("Non e` palindroma\n");

  return 0;
}  /* main */
```

ESERCIZIO 3

Si scriva un programma in linguaggio C, che legga da tastiera i risultati (double) di 20 esperimenti.
Stampi il numero d'ordine ed il risultato di quegli esperimenti per i quali il risultato è minore del 50% della media dei 20 risultati.

```
#include <stdio.h>

#define DIM 4


int main(void)
{
  double ris[DIM];
  double media;
  int i;

  /* inserimento dei valori */
  printf("Inserire i %d risultati dell'esperimento:\n", DIM);
  for (i = 0; i < DIM; i++) {
    printf("Inserire risultato n. %d: ", i);
    scanf("%lg", &ris[i]);
  }

  /* calcolo della media */
  media = 0;
  for (i = 0; i < DIM; i++)
    media = media + ris[i];
  media = media/DIM;
  printf("Valore medio: %lg\n", media);

  /* stampa dei valori minori di media*0.5 */
  printf("Stampa dei valori minori di media*0.5:\n");
  for (i = 0; i < DIM; i++)
    if (ris[i] < media * 0.5)
      printf("Risultato n. %d: %lg\n", i, ris[i]);

  return 0;
}
```

ESERCIZIO 4

Si scriva un programma in linguaggio C, che faccia uso della seguente funzione: void concat(char s1[], char s2[], char s3[]), che ponga nella stringa s3 la concatenazione delle stringhe s1 e s2. Per esempio: se s1=”prima”, s2=”vera”, al ritorno dalla chiamata a concat() si deve avere s3=”primavera”. La trasformazione deve avvenire carattere per carattere senza l’utilizzo delle librerie standard che riguardano l’uso delle stringhe.

```
#include <stdio.h>
#define N 80
```

```
void concat(char s1[],char s2[], char s3[]);

main() {
       char s1[N],s2[N],s3[N];

       scanf("%s",s1);
       scanf("%s",s2);

       concat(s1,s2,s3);

       printf("%s",s3);
}


/* PRIMA VERSIONE (con while)*/
void concat(char s1[],char s2[], char s3[])
{
 int i=0,j=0;

 while (s1[i] != '\0') s3[j++] = s1[i++];
 i = 0;
 while (s2[i] != '\0') s3[j++] = s2[i++];
 s3[j] = '\0';
}

  /* SECONDA VERSIONE (con for) */
void concat(char s1[],char s2[], char s3[])
{
 int i,j;

 for (i=0; s1[i] != '\0'; i++) s3[i] = s1[i];
 for (j=i,i=0; s2[i] != '\0'; i++,j++) s3[j] = s2[i];
 s3[j] = '\0';
}
```

ESERCIZIO 6

Si scriva un programma in linguaggio C, che legga una matrice quadrata e verifichi se è magica, ovvero le somme delle righe, delle colonne e delle due diagonali coincidono.

```
#include <stdio.h>

#define DIM 3


void leggiMatrice(int m[DIM][DIM])
{
  int i, j;

  printf("Inserire gli elementi di una matrice di interi %dx%d\n", DIM, DIM);
  for (i = 0; i < DIM; i++)
    for (j = 0; j < DIM; j++)
      scanf("%d", &m[i][j]);
}
```

```
void stampaMatrice(int m[DIM][DIM])
{
  int i, j;

  for (i = 0; i < DIM; i++) {
    for (j = 0; j < DIM; j++)
      printf("%5d", m[i][j]);
    printf("\n");
  }
}


int sommaRiga(int m[DIM][DIM], int riga)
{
  int j, somma = 0;

  for (j = 0; j < DIM; j++)
    somma += m[riga][j];

  /* printf("Somma riga %d: %d\n", riga, somma); */
  return somma;
}


int sommaColonna(int m[DIM][DIM], int colonna)
{
  int i, somma = 0;

  for (i = 0; i < DIM; i++)
    somma += m[i][colonna];

  /* printf("Somma colonna %d: %d\n", colonna, somma); */
  return somma;
}


int sommaDiagonale(int m[DIM][DIM], int principale)
  /* Calcola la somma della diagonale principale o secondaria, a seconda del
     valore di principale. */
{
  int i, somma = 0;

  if (principale)
    for (i = 0; i < DIM; i++)
      somma += m[i][i];
  else
    for (i = 0; i < DIM; i++)
      somma += m[i][DIM-i-1];

  /*
  if (principale)
    printf("Somma diagonale principale: %d\n", somma);
  else
    printf("Somma diagonale secondaria: %d\n", somma);
  */

  return somma;
}


int magica(int m[DIM][DIM])
{
```

```
  int i, somma;
  int mag = 1;     /* booleana - indica che la matrice e` magica */

  somma = sommaDiagonale(m, 1);
  mag = (somma == sommaDiagonale(m, 0));

  for (i = 0; (i < DIM && mag); i++)
    if (somma != sommaRiga(m, i) || somma != sommaColonna(m, i))
      mag = 0;

  return mag;
}


int main(void)
{
  int mat[DIM][DIM];

  leggiMatrice(mat);

  printf("La matrice\n");
  stampaMatrice(mat);

  if (magica(mat))
    printf("e` magica.\n");
  else
    printf("non e` magica.\n");

  return 0;
}
```