Laboratorio Informatica A Incontro 4

**Problema 1**

Scrivere un programma che richieda all'utente di riempire una matrice 3 x 3 di interi, la stampi, sostituisca nella matrice ogni numero pari con la sua metà e, infine, stampi la matrice così modificata.

// es1.c

// lab4

//

// Created by ing.conti on 17/10/20.

#include <stdio.h>

#define DIM 3

**int** main(**int** argc, **const** **char** \* argv[]) {

**int** matr[DIM][DIM];

/\* caso di test:

int matr[DIM][DIM] = {1,2,3,4,5,6,7,8,9};

rimouvere il dopo for di lettura.. deve uscire:

2' stampa:

1 1 3

2 5 3

7 4 9

\*/

**int** r,c;

**for** (r=0; r<DIM; r++) {

**for** (c=0; c<DIM; c++) {

printf("\ndammi elem %d %d:", r, c);

scanf("%d", &matr[r][c]);

}

}

printf("\n 1' stampa:\n");

**for** (r=0; r<DIM; r++) {

printf("\n"); // andiamo a capo ad ogni riga

**for** (c=0; c<DIM; c++) {

printf("%2d ", matr[r][c]); // 2%d == 2 cifre decimale

}

}

printf("\n sostituisco..\n");

**for** (r=0; r<DIM; r++) {

**for** (c=0; c<DIM; c++) {

**if** ( matr[r][c] % 2 == 0 )

matr[r][c] /= 2;

}

}

printf("\n 2' stampa:\n");

**for** (r=0; r<DIM; r++) {

printf("\n"); // andiamo a capo ad ogni riga

**for** (c=0; c<DIM; c++) {

printf("%2d ", matr[r][c]);

}

}

**return** 0;

}

**Problema 2 “warm up” puntatori**

Scrivere un programma che richieda all'utente di riempire una stringa S1 di caratteri,

cerchi al suo intero i caratteri numerici e li copi in una stringa S2 e stampi S2.

NB: va scritto usando puntatori ed aritmetica dei puntatori (Pointer arithmetic)

// es2\_warm\_up.c - lab4

// Created by ing.conti on 18th oct, 2020

#include <stdio.h>

#define DIM 128

**int** main(**int** argc, **const** **char** \* argv[]) {

**char** S1[DIM];

// caso di test:

//char S1[DIM] = "ab1 2 c d e 9";

// stampa: 129

**char** S2[DIM];

scanf("%s", S1);

**char** \* pnt1 = &S1[0];

**char** \* pnt2 = &S2[0];

**char** ch = ' '; // ch sara' il carattere corrente. lo inzializiammo per poter innescafre il for

// si puo' anche usare direttamente \* pnt

**int** fine;

**for** (fine = 0; !fine; pnt1++) {

ch = \*pnt1;

**if** (ch>='1' && ch <='9'){

\*pnt2 = ch;

pnt2++;

}**else** **if** (ch == '\0'){

fine = 1;

}

}

// terminamo correttamente s2:

\*pnt2 = '\0';

printf("\n stampa: %s\n", S2);

**return** 0;

}

**Problema 2**

Scrivere un programma che richieda all'utente di riempire una matrice 3 x 3 di interi, la stampi, calcoli e stampi la media di tutti gli elementi divisibili per 3 della matrice stessa.

L’ esercizio è SIMILE all’ es1, ma va scritto usando puntatori ed aritmetica dei puntatori (Pointer arithmetic)

// es2.c

// lab4

// Created by ing.conti on 17/10/20.

#include <stdio.h>

#define DIM 3

**int** main(**int** argc, **const** **char** \* argv[]) {

**int** matr[DIM][DIM];

/\* caso di test:

int matr[DIM][DIM] = {1,2,3,4,5,6,7,8,9};

rimouvere il dopo for di lettura.. deve uscire:

2' stampa:

1 1 3

2 5 3

7 4 9

\*/

**int** \* pnt\_inizio = &matr[0][0];

**int** \* pnt\_fine = pnt\_inizio + DIM \* DIM;

**int** \* pnt;

// per motivi didattici e per vedere riga e colonna, ricreiamo qui le v.

// con un po' di "Pointer arithmetic"

**int** offset, r, c;

**for** (pnt = pnt\_inizio; pnt < pnt\_fine; pnt++) {

offset = pnt - pnt\_inizio;

c = offset % DIM;

r = offset / DIM;

printf("\ndammi elem %d %d:", r, c);

scanf("%d", pnt);

}

printf("\n 1' stampa:\n");

**for** (pnt = pnt\_inizio; pnt < pnt\_fine; pnt++) {

offset = pnt - pnt\_inizio;

c = offset % DIM;

r = offset / DIM;

**if** (c == 0)

printf("\n"); // andiamo a capo ad ogni riga

printf("%2d ", \*pnt);// 2%d == 2 cifre decimale

}

printf("\n sostituisco..\n");

**for** (pnt = pnt\_inizio; pnt < pnt\_fine; pnt++) {

**if** (\*pnt % 2 == 0)

\*pnt /= 2;

}

printf("\n 2' stampa:\n");

**for** (pnt = pnt\_inizio; pnt < pnt\_fine; pnt++) {

offset = pnt - pnt\_inizio;

c = offset % DIM;

r = offset / DIM;

**if** (c == 0)

printf("\n"); // andiamo a capo ad ogni riga

printf("%2d ", \*pnt);// 2%d == 2 cifre decimale

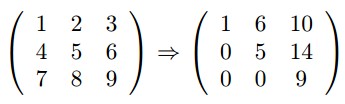
}

**return** 0;

}

**Problema 3**

Scrivere un programma che generi una matrice 3x3 inserendo dei numeri progressivi in ogni cella, la stampi e poi la trasformi in una matrice triangolare superiore avente come elementi le somme degli elementi simmetrici rispetto alla diagonale principale.



// es3.c lab4

// Created by ing.conti on 17/10/20.

#include <stdio.h>

/\*

Scrivere un programma che generi una matrice 3x3 inserendo dei numeri progressivi in ogni cella, la stampi e poi la trasformi in una matrice triangolare superiore avente come elementi le somme degli elementi simmetrici rispetto alla diagonale principale.

\*/

#define DIM 3

**int** main(**int** argc, **const** **char** \* argv[]) {

**int** matr[DIM][DIM];

// parte inziale simile ad ed 2.. usiamo pointer, si puo' fare ANCHE con indici..

**int** \* pnt\_inizio = &matr[0][0];

**int** \* pnt\_fine = pnt\_inizio + DIM \* DIM;

**int** \* pnt;

**int** offset, r, c;

**for** (pnt = pnt\_inizio; pnt < pnt\_fine; pnt++) {

offset = pnt - pnt\_inizio;

\*pnt = offset+1; // caricheremo numeri da 1 a 9

}

printf("\n 1' stampa:\n");

**for** (pnt = pnt\_inizio; pnt < pnt\_fine; pnt++) {

offset = pnt - pnt\_inizio;

c = offset % DIM;

r = offset / DIM;

**if** (c == 0)

printf("\n"); // andiamo a capo ad ogni riga

printf("%5d ", \*pnt);

}

**for**(r=0;r<DIM;r++)

**for**(c=r;c<DIM;c++){

**if**(r!=c){

matr[r][c]+=matr[c][r];

matr[c][r]=0;

}

}

printf("\n 2' stampa:\n");

**for** (pnt = pnt\_inizio; pnt < pnt\_fine; pnt++) {

offset = pnt - pnt\_inizio;

c = offset % DIM;

r = offset / DIM;

**if** (c == 0)

printf("\n"); // andiamo a capo ad ogni riga

printf("%5d ", \*pnt);

}

**return** 0;

}

**Problema 4**

Scrivere un programma che esegue un inserimento controllato di una matrice ed in particolare controlla che il valore corrente non sia già stato inserito dall’utente in precedenza

***Hint:*** Si consideri come viene riempita la matrice. Tipicamente l’inserimento avviene per righe, quindi occorre controllare interamente le righe precedenti e la riga corrente fino alla colonna specificata.

A picture containing table

Description automatically generated

// es4.c - lab4

// Created by ing.conti on 17/10/20.

//

#include <stdio.h>

#define N 2

**int** main(){

**int** i,j,k,t,A[N][N],ok=1;

**for**(i=0;i<N;i++) {

**for**(j=0;j<N;j++) {

**do** {

ok=1;

printf("Inserire un valore\n"); scanf("%d", &A[i][j]);

// controlla le righe precedenti

**for** (k=0; k<i && ok; k++)

**for** (t=0; t<N && ok; t++)

**if** (A[i][j] == A[k][t]) {

ok=0;

printf("duplicato\n");}

// controlla la riga corrente

**for** (t = 0; t < j && ok; t++)

**if**(A[i][j] == A[i][t]){

ok=0;

printf("duplicato nella stessa riga\n");

}

}**while** (ok==0);

}

}

}

**Problema 5 (struct)**

Scrivere un programma che effettui somme e prodotti fra numeri complessi.

Chart

Description automatically generated

* Si definisca la struct adeguata definire un numero complesso (parte reale e parte immaginaria)
* Si scriva il codice che legga 2 numeri complessi C1 e C2 e ne calcoli la somma SC e prodotto complesso PC

Theory:

Dati a+bi and c+id, la somma e’ (a+c) + i (b+d).

Dati a+bi and c+id, il prodotto e’ (ac -bd) + i (ad+bc)

Verificare che, dati:

C1 = 1+ 1i, C2 = 1+ 3i si abbia: SC = 2 + 4i, PC = -2 + 4i

// es5.c - lab4

// Created by ing.conti on 17/10/20.

//

#include <stdio.h>

**typedef** **struct** t\_Complex{

**double** Re;

**double** Im;

} Complex;

**int** main(){

Complex C1, C2, SC, PC;

// tralasciamo input...

// eventualmente sarebbe::

//scanf("%f", C1.Re);

//scanf("%f", C1.Im);

C1.Re = 1; C1.Im = 1;

C2.Re = 1; C2.Im = 3;

SC.Re = C1.Re + C2.Re;

SC.Im = C1.Im + C2.Im;

printf("somma %f, i %f\n", SC.Re, SC.Im);

PC.Re = C1.Re \* C2.Re - C1.Im \* C2.Im; // (a\*c -b\*d)

PC.Im = C1.Re \* C2.Im + C1.Im \* C2.Re; // (a\*d+b\*c)

printf("somma %f, i %f\n", PC.Re, PC.Im);

}