Struct && Array di struct

Struct

Recap: Esempio punto cartesiano

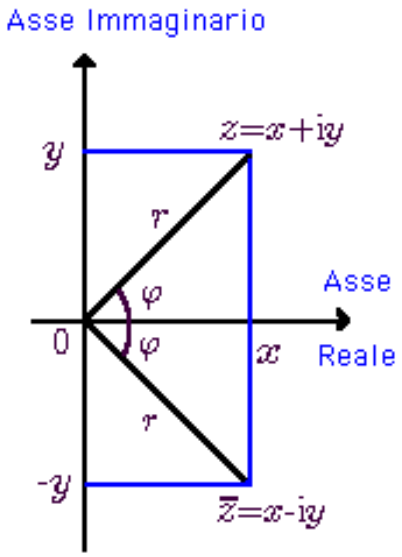
```
typedef struct CartesianPoint{
    float x;
    float y;
}CartesianPoint;
```

Da qui in avanti Complex e' un topo come gli altri...

```
int anni;
CartesianPoint p;
```

Esercizio 1

1) si scriva una opportuna *struct* del C atta a rappresentare i numeri complessi.



Un numero complesso deve avere parte Reale e parte Immaginaria.

2) si dichiari una variabile N del tipo struct appena creato.

Esercizio 1 soluzione

gianenrico.conti@polimi.it

```
Complex c1;
```

Esisterebbero atri modi...
p.es. potreste trovare uno "stile":

```
typedef struct Complex{
    float Re;
    float Im;
} t_Complex;

t_Complex c1;
```

Es 2

Dato un punto da input, si stampino modulo e fase (anomalia)

Es2:Dato un punto da input, si stampino modulo e fase (anomalia)

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

typedef struct Complex{
    float re;
    float im;
}Complex;

int main() {

   float mod, arg;
    Complex p1;
    p1.re = 30;
    p1.im = 40;

//    scanf("%f", &p1.re);
//    scanf("%f", &p1.im);


    mod = sqrt(
        p1.re * p1.re +
        p1.im * p1.im // pow?? ^ NOO
        );

    arg = atan(p1.im/ p1.re);

    printf("%f %f\n", mod, arg);


    return 0;
}
```

Es3: data una sequenza di N punti sul piano cartesiano, si stampi il piu distante dall' origine.

..

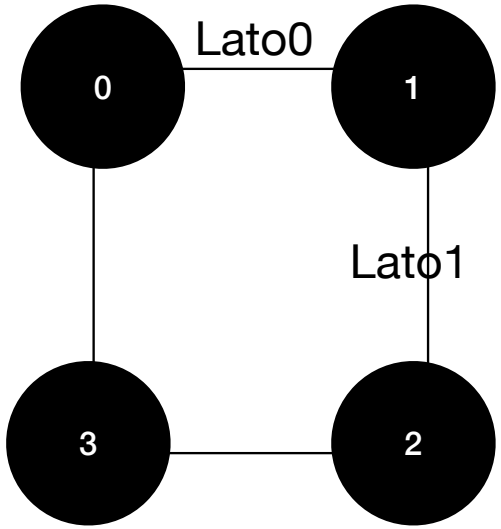
Es 3 soluzione

ES: data una sequenza di N punti sul piano cartesiano, si stampi il piu distante dall' origine.

```c
#include <stdio.h>
#define MAX 3
typedef struct CartesianPoint{
    float re;
    float im;
}CartesianPoint;

int main() {
    int i, indice_max;
    float d2, max;
    CartesianPoint  points[MAX];
    for (i = 0; i < MAX; i++) {
        printf("dammi RE: ");
        scanf("%f", &points[i].re);

        printf("dammi IM: ");
        scanf("%f", &points[i].im);
    }

    indice_max = 0;
    // calcolo il quadrato della distanza ..
    max = 0;
    for (i = 0; i < MAX; i++) {
        d2 = points[i].re * points[i].re +
                points[i].im * points[i].im;
        if (d2>max) {
            max = d2;
            indice_max = i;
        }
    }

    printf("\ntrovato in posizione %d\n", indice_max);
    printf("%f i%f\n", points[indice_max].re, points[indice_max].im);
    return 0;
}
```

Es 4

Data una sequenza di punti sul piano cartesiano, che rappresenti un poligono, si calcoli
Il perimetro



Es 4

Data una sequenza di punti sul piano cartesiano, che rappresenti un poligono, si calcoli
Il perimetro

ES: Data una sequenza di punti sul piano cartesiano, che rappresenti un poligono, si calcoli Il perimetro

```c
//  main.c
//  cartesian
//
//  Created by ing.conti on 20/10/24.
//
#include <stdio.h>
#include <math.h>

#define MAX 30

typedef struct CartesianPoint{
    float re;
    float im;
}CartesianPoint;

int main() {
    int i,k,quanti;
    float lato, perimetro, b, h;
    CartesianPoint points[MAX], pk, pksucc;

    printf("quanti punti: ");
    scanf("%d", &quanti);

    // (1,1) (4,5) 4,1)
    for (i = 0; i < quanti && quanti<MAX; ++i) {
        printf("dammi re di [%d]", i+1);
        scanf("%f", &points[i].re);

        printf("dammi im di [%d]", i+1);
        scanf("%f", &points[i].im);

    }

perimetro = 0;
    for (k = 0; k < i-1; k++) {
        pk = points[k];
        pksucc = points[k+1];
        b = pk.re - pksucc.re;
        h = pk.im - pksucc.im;
        lato = sqrt(b*b + h*h);
        perimetro = perimetro+lato;
    }


// CHIUDO VERSO PUNTO INIZIALE:

    pk = points[0];
    b = pk.re - pksucc.re;
    h = pk.im - pksucc.im;
    lato = sqrt(b*b + h*h);
    perimetro = perimetro+lato;


        return 0;
}
```