

Fondamenti di Informatica e Laboratorio  
a.a. 2020

---

# Lab 07

## Passaggio per riferimento

# Esempio visto a lezione

---

## Esempio: "Restituire" Valori Multipli

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

void sin_cos(float x, float *s, float *c) {
    *s = sin(x); // riempio *s
    *c = cos(x); // riempio *c
}

int main() {
    float x = 3.14159, ys, yc;
    sin_cos(x, &ys, &yc);
    printf("sin: %f, cos: %f\n", ys, yc);
}
```

# Esercizio 1

---

## Somma di due numeri complessi

Realizzare una procedura che riceva in ingresso due numeri complessi

- Un numero complesso è dato da una coppia rappresentante la parte reale e la parte immaginaria

La procedura deve restituire la somma di tali valori (ancora una coppia)

```
void somma_complex(  
    float reA, float imA, float reB, float imB,  
    float * reResult, float * imResult);
```

Realizzare anche un main di esempio

# Esercizio 1 - Soluzione

---

```
void somma_complex(
    float reA, float imA, float reB, float imB,
    float * reResult, float * imResult)
{
    *reResult = reA + reB;
    *imResult = imA + imB;
    return;
}

int main() {
    float reResult, imResult;
    somma_complex(1.0, 1.0, 2.0, 2.0, &reResult, &imResult);
    printf("Parte reale: %f\n
           Parte Immaginaria: %f\n", reResult, imResult);
}
```

# Esercizio 2

---

Un sistema di cronometraggio per la Formula 1 registra i tempi in millisecondi. Tuttavia tali tempi devono essere presentati in termini di minuti, secondi e millisecc.

Creare una procedura che, ricevuti in ingresso un tempo dato in millisecondi, restituisca l'equivalente in termini di minuti, secondi, millisecc (passati per riferimento).

```
void fromMillisec(int millisec, int * mm, int * sec, int * min);
```

Si realizzi un main che invoca la funzione, e che dopo aver chiesto all'utente un valore indicante una durata in millisecondi, stampi a video il tempo nel formato min:sec.millisec

```
secondi = millisecondi / 1000
secondi = secondi % 60
minuti = secondi / 60;
millisecondi = millisecondi % 1000
```

**Esempio:**  
Inserisci un tempo in millisec.: 347829  
Tempo: 5:47.829

# Esercizio 2 - Soluzione

---

```
#include <stdio.h>
#include <stdlib.h>

void fromMillisec(int millisec, int * mm, int * sec, int * min) {
    *mm = millisec % 1000;
    *sec = millisec / 1000;
    *min = *sec / 60;
    *sec = *sec % 60;
    return;
}

int main(void) {
    int millisec, mm, sec, min;

    printf("Inserisci un tempo in millisec.: ");
    scanf("%d", &millisec);
    fromMillisec(millisec, &mm, &sec, &min);
    printf("Tempo: %d:%d.%d\n", min, sec, mm);

    return (0);
}
```

# Esercizio 3

---

Realizzare una procedura che, ricevuti in ingresso un vettore di interi e la sua dimensione, e due interi passati per riferimento di nome “pari” e “dispari”, restituisca il numero di interi pari e di interi dispari presenti nell’array.

```
void contaPariDisp(int vet[], int dim, int * pari, int *  
    disp) ;
```

Si realizzi un main che, utilizzando una appropriata funzione, legga dall’utente una sequenza di al più 10 numeri (terminati da zero), e utilizzando la procedura di cui al punto precedente, stampi a video quanti numeri pari e dispari sono stati inseriti.

# Esercizio 3 - Soluzione

---

```
#include <stdio.h>
#include <stdlib.h>

int leggi(int vet[], int dim) {
    int i, num;
    i=0;
    do {
        printf("Inserisci numero: ");
        scanf("%d", &num);
        if (num != 0) {
            vet[i] = num;
            i++;
        }
    } while (num!=0 && i<dim);
    return i;
}

void contaPariDisp(int vet[], int dim, int * pari, int * disp) {
    int i;

    *pari = 0;
    *disp = 0;
    for (i=0; i<dim; i++) {
        if ( (vet[i]%2)==0)
            (*pari)++;
        else
            (*disp)++;
    }
}
```

# Esercizio 3 - Soluzione

---

...

```
int main(void)
{
    int vet[10], pari, disp, dim;

    dim = leggi(vet, 10);

    contaPariDisp(vet, dim, &pari, &disp);

    printf ("l'array contiene %d numeri pari e %d dispari", pari, disp);

    return (0);
}
```

# Esercizio 4

---

Si vuole realizzare una funzione che, dati un array di valori interi, ordinati non ripetuti, e due valori estremi, restituisca il sotto-array compreso tra i due estremi.

Tale funzione quindi riceverà in ingresso un vettore di interi e la sua dimensione; due interi di nome “first” e “last”; un intero dim passato per riferimento. La funzione dovrà restituire un puntatore all’elemento dell’array pari a first, se presente, e tramite dim la dimensione logica del sotto-array

```
int * select(int v[], int length, int first, int last, int *  
dim) ;
```

Ad esempio, se invocata con  $v=\{1,2,3,5,6,8,9\}$ ,  $first=3$ ,  $last=8$ , la funzione deve restituire il puntatore all’elemento all’indice 2 ( $\&v[2]$ ), e dimensione 4.

# Esercizio 4 - Soluzione

---

```
int * select(int v[], int length, int first, int last, int * dim) {
    int i;
    int * result;

    i=0;
    while (i<length && first>v[i])
        i++;
    result = &(v[i]);

    *dim = 0;
    while (i<length && last>=v[i]) {
        i++;
        *dim = *dim + 1;
    }
    return result;
}
```

...

# Esercizio 4 - Soluzione

---

```
int main(void)
{
    int v[10], dim_v;
    int * v2;
    int dim_v2;
    int first, last, i;

    dim_v = leggi(v, 10);
    printf("Inserisci i due estremi: ");
    scanf("%d%d", &first, &last);
    v2 = select(v, dim_v, first, last, &dim_v2);
    for (i=0; i<dim_v2; i++)
        printf("%d ", v2[i]);

    system("PAUSE");

    return (0);
}
```

# Esercizio 5

---

## I numeri complessi

Data la notazione cartesiana di un numero complesso (in parte reale ed immaginaria),

Realizzare una procedura che ne restituisca la notazione polare (tramite parametri passati per riferimento)

Si usi opportunamente la funzione `atan2(float im, float re)` della libreria `math.h`

$$r = \sqrt{re^2 + im^2}$$

$$\varphi = \arctan\left(\frac{im}{re}\right)$$

La funzione `atan2` gestisce correttamente anche il caso in cui `re==0`. Se così non fosse? Si estenda la funzione di conversione in modo da controllare la correttezza dei parametri: la funzione restituisca un codice di errore se necessario.

```
void converti_complex(float re, float im,  
                     float * modulo, float * argomento)
```

# Esercizio 5 – Soluzione

---

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

void converti_complex(
    float re, float im,
    float * modulo, float * argomento)
{
    *modulo = sqrt(re*re + im*im);
    *argomento = atan2(im, re);
    return;
}

int main() {
    float modulo, argomento;
    converti_complex(1.0, 1.0, &modulo, &argomento);
    printf("Modulo: %f\n
           Argomento: %f\n", modulo, argomento);
}
```

# Esercizio 5 – Soluzione

(variante)

---

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

int converti_complex( float re, float im,
                    float * modulo, float * argomento) {
    if (re==0) return -1;
    *modulo = sqrt(re*re + im*im);
    *argomento = atan2(im, re);
    return 0;
}

int main() {
    float modulo, argomento;
    converti_complex(1.0, 1.0, &modulo, &argomento);
    printf("Modulo: %f\n
           Argomento: %f\n", modulo, argomento);
}
```

# Esercizio 6

---

Realizzare una funzione che riceva in ingresso un array di interi e la sua dimensione, un elemento da cercare ed un intero passato per riferimento.

La funzione deve restituire un valore interpretabile come “vero” se l’elemento è presente nell’array. Inoltre, tramite l’intero passato per riferimento, la funzione deve restituire anche la posizione dell’elemento nell’array

Realizzare anche un main di esempio

# Esercizio 6 - Soluzione

---

```
int trovaPos(int vet[], int dim, int el, int *pos) {  
  
    int trovato, i;  
  
    trovato = 0;  
    for (i=0; i<dim && trovato==0; i++) {  
        if (vet[i] == el) {  
            trovato = 1;  
            *pos = i;  
        }  
    }  
    return trovato;  
}
```

# Esercizio 7

---

Un sistema di gestione mp3 permette di calcolare in anticipo la durata di una compilation di brani.

Creare una procedura che, ricevuti in ingresso la durata di due pezzi musicali, in termini di ore, minuti e secondi, restituisca la durata risultante dalla somma dei due brani in termini di ore, minuti e secondi.

Si realizzi un main che chieda all'utente di inserire la durata di diversi brani musicali, e si stampi a video la durata totale (l'utente segnala il termine dei brani da inserire con un brano speciale di lunghezza 0:00.00).

# Esercizio 7 - Soluzione

---

```
#include <stdio.h>
#include <stdlib.h>

void sommaTempi(int h1, int m1, int s1, int h2, int m2, int s2, int * hr,
                int * mr, int * sr) {
    *sr = s1 + s2;
    *mr = *sr / 60;
    *sr = *sr % 60;
    *mr = *mr + m1 + m2;
    *hr = *mr / 60;
    *mr = *mr % 60;
    *hr = *hr + h1 + h2;
}

...
```

# Esercizio 7 - Soluzione

---

...

```
int main(void)
{
    int h1, h2=0;
    int m1, m2=0;
    int s1, s2=0;
    int i=1;

    do {
        printf("inserisci la durata della canzone numero %d (hh:mm:ss): ", i);
        scanf("%d%d%d", &h1, &m1, &s1);
        if (! (h1==0 && m1==0 && s1==0))
            sommaTempi(h1, m1, s1, h2, m2, s2, &h2, &m2, &s2);
        i++;
    } while ( ! (h1==0 && m1==0 && s1==0));

    printf("Durata totale: %dh:%dm:%ds\n", h2, m2, s2);

    system("PAUSE");
    return (0);
}
```