# IEIM 2015-2016

## Esercitazione III
## *"Mappe di Karnaugh, array e alcune note sulle funzioni"*

Alessandro A. Nacci
alessandro.nacci@polimi.it - www.alessandronacci.it

# Cosa facciamo oggi?

- MAPPE DI KARNAUGH
- UN ESERCIZIO SUGLI ARRAY
- L'INIZIALIZZAZIONE DEGLI ARRAY
- VARIABILI E FUNZIONI

# MAPPE DI KARNAUGH

# Concetti base per le Mappe di Karnaugh

Tra le proprietà dell'algebra di Boole, le seguenti consentono di semplificare notevolmente le espressioni booleane:

$$A \cdot B + A \cdot \overline{B} = A \cdot (B + \overline{B}) = A$$

$$A \cdot (\overline{B} \cdot \overline{C} + \overline{B} \cdot C + B \cdot C + B \cdot \overline{C}) = A$$

Le mappe di Karnaugh sono una particolare forma di tabella di verità, che consente di individuare immediatamente la possibilità di fare queste semplificazioni.

# Concetti base per le Mappe di Karnaugh

Ad esempio, la seguente tabella di verità della funzione **Y = Y(A,B,C)**

| A | B | C | Y |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 1 |

può essere ridisegnata così:

| A | 0 | 0 | 1 | 1 |
|---|---|---|---|---|
| B | 0 | 1 | 1 | 0 |
| C | | | | |
| 0 | 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 1 | 1 |

Mappa di Karnaugh della funzione **Y**

Dalla tabella di verità o dalla mappa di Karnaugh è immediato ottenere l'espressione booleana della funzione **Y** come "somma" di "prodotti", cioè come OR di tanti termini AND quante sono le caselle in cui la funzione vale 1; ciascuno di questi termini AND (detti *minterm*) è costituito dall'AND delle variabili di ingresso, negate oppure no a seconda che il valore della variabile associato a quella casella sia 0 oppure 1.

$$Y = A \cdot B \cdot \overline{C} + \overline{A} \cdot B \cdot C + A \cdot B \cdot C + A \cdot \overline{B} \cdot C$$

# Concetti base per le Mappe di Karnaugh

Nel caso di funzioni di 4 variabili, ad es. **Z = Z(A,B,C,D)**, la mappa di Karnaugh ha 4 righe e quattro colonne:

| CD \ AB | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 00 | 0 | 0 | 1 | 0 |
| 01 | 1 | 0 | 1 | 1 |
| 11 | 1 | 1 | 1 | 1 |
| 10 | 1 | 1 | 1 | 0 |

Mappa di Karnaugh della funzione **Z**

Nelle mappe di Karnaugh i valori della funzione **Y** sono scritti dentro le caselle.

I valori delle variabili **A,B,C,D** sono indicati come “coordinate” delle caselle. Esaminando queste “coordinate, si constata che le coppie di valori di **A** e **B** (di **C** e **D**) associate alle colonne (alle righe) sono ordinate in modo che tra due caselle adiacenti (della medesima riga o della medesima colonna) cambia il valore di una sola delle variabili, mentre quello di tutte le altre rimane lo stesso; questa proprietà vale anche tra le caselle estreme di ciascuna riga e di ciascuna colonna (che, sotto questo aspetto, possono quindi essere considerate “adiacenti”, in senso circolare).

# Concetti base per le Mappe di Karnaugh

Nel caso di funzioni di 4 variabili, ad es. **Z = Z(A,B,C,D)**, la mappa di Karnaugh ha 4 righe e quattro colonne:

| CD \ AB | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 00 | 0 | 0 | 1 | 0 |
| 01 | 1 | 0 | 1 | 1 |
| 11 | 1 | 1 | 1 | 1 |
| 10 | 1 | 1 | 1 | 0 |

Mappa di Karnaugh della funzione **Z**

Nelle mappe di Karnaugh i valori della funzione **Y** sono scritti dentro le caselle.

I valori delle variabili **A,B,C,D** sono indicati come “coordinate” delle caselle. Esaminando queste “coordinate, si constata che le coppie di valori di **A** e **B** (di **C** e **D**) associate alle colonne (alle righe) sono ordinate in modo che tra due caselle adiacenti (della medesima riga o della medesima colonna) cambia il valore di una sola delle variabili, mentre quello di tutte le altre rimane lo stesso; questa proprietà vale anche tra le caselle estreme di ciascuna riga e di ciascuna colonna (che, sotto questo aspetto, possono quindi essere considerate “adiacenti”, in senso circolare).

# Ora proviamo con un esempio…

| | a | b | c | d |
|---|---|---|---|---|
| | 0 | 0 | 1 | 1 |
| | 0 | 1 | 0 | 0 |
| | 0 | 1 | 1 | 0 |
| | 0 | 1 | 1 | 1 |
| | 1 | 1 | 0 | 0 |
| | 1 | 1 | 0 | 1 |
| | 1 | 1 | 1 | 0 |

1. $F_{(a,b,c,d)} = \mathbf{a'b'cd+a'bc'd'+a'bcd'+a'bcd+abc'd'+abc'd+abcd'}$

# Ora proviamo con un esempio…

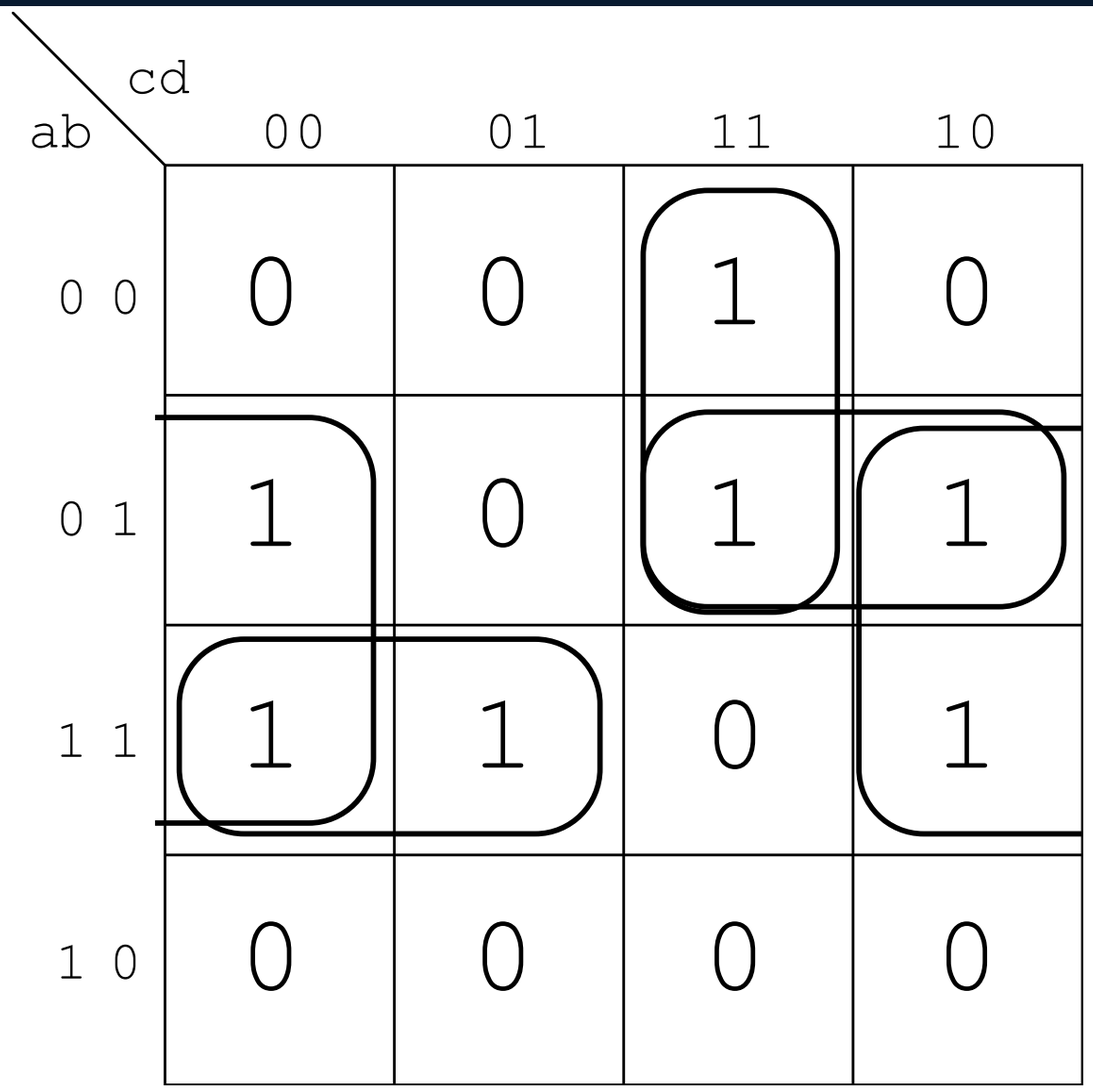
| | a | b | c | d |
|---|---|---|---|---|
| | 0 | 0 | 1 | 1 |
| | 0 | 1 | 0 | 0 |
| | 0 | 1 | 1 | 0 |
| | 0 | 1 | 1 | 1 |
| | 1 | 1 | 0 | 0 |
| | 1 | 1 | 0 | 1 |
| | 1 | 1 | 1 | 0 |

| ab \ cd | 0 0 | 0 1 | 1 1 | 1 0 |
|---|---|---|---|---|
| 0 0 | 0 | 0 | 1 | 0 |
| 0 1 | 1 | 0 | 1 | 1 |
| 1 1 | 1 | 1 | 0 | 1 |
| 1 0 | 0 | 0 | 0 | 0 |

1. $F_{(a,b,c,d)} = \mathbf{a'b'cd+a'bc'd'+a'bcd'+a'bcd+abc'd'+abc'd+abcd'}$

# Ora proviamo con un esempio…

| | a | b | c | d |
|---|---|---|---|---|
| | 0 | 0 | 1 | 1 |
| | 0 | 1 | 0 | 0 |
| | 0 | 1 | 1 | 0 |
| | 0 | 1 | 1 | 1 |
| | 1 | 1 | 0 | 0 |
| | 1 | 1 | 0 | 1 |
| | 1 | 1 | 1 | 0 |




1. $F_{(a,b,c,d)}$ = **a'b'cd+a'bc'd'+a'bcd'+a'bcd+abc'd'+abc'd+abcd'**
2. Implicanti primi: a'cd, a'bc, bd', abc'
3. Implicanti primi essenziali: a'cd, bd', abc'
4. Copertura minima: a'cd+bd'+abc'

# PROGRAMMAZIONE C

# - GLI ARRAY -

# Esercizio 1

- Scrivere un programma che
  - legga due array di interi da tastiera
  - dica quale dei due array ha valor medio più alto
  - calcoli l'array concatenato tra i due array in ingresso
  - trovi tutti i numeri primi inseriti
  - trovi il massimo ed il minimo tra tutti i valori inseriti
  - dica in quale dei due array sono presenti i due valori di massimo e minimo
  - calcoli l'array riversato

In questa soluzione, non usare le funzioni!

# Esercizio 1: dichiarazione variabili

```
int i,j;
int a[DIM];
int b[DIM];

int ab[DIM2];
int ab_r[DIM2];

float medio_a, medio_b;
float somma = 0;
bool primo = 1;              //booleana
```

# Esercizio 1: lettura array

```
// LETTURA VALORI
// ------------------------------------

// Leggo l'arrayA
for (i = 0; i < DIM; i++){
     printf("\nInserisci i valore %d per l'array a: ", i);
     scanf("%d",&a[i]);
}

// Stampo l'array A
printf("\narray a = ");
for (i = 0; i < DIM; i++)
     printf("%d ", a[i]);

// Leggo l'array B
for (i = 0; i < DIM; i++){
     printf("\nInserisci i valore %d per l'array b: ", i);
     scanf("%d",&b[i]);
}

// Stampo l'array B
printf("\narray b = ");
for (i = 0; i < DIM; i++)
     printf("%d ", b[i]);
```

# Esercizio 1: calcolo del valor medio

```
// VALOR MEDIO
// -----------------------------------

// Calcolo valor medio si A

for (i = 0; i < DIM; i++){
    somma += a[i];
}
medio_a = somma / DIM;
```

```
// Calcolo valor medio si B
somma = 0;
for (i = 0; i < DIM; i++){
    somma += b[i];
}
medio_b = somma / DIM;

if(medio_a == medio_b)
    printf("L'array a e l'array b hanno lo stesso valor medio.\n");
else if (medio_a > medio_b)
    printf("L'array a ha valor medio piu' alto dell'array b.\n");
else if (medio_a < medio_b)
    printf("L'array b ha valor medio piu' alto dell'array a.\n");
```

# Esercizio 1: concatenazione

```
// CONCATENAZIONE
// ----------------------------------

for (i=0; i < DIM; i++)
     ab[i] = a[i];

for (i=0; i < DIM; i++)
     ab[i+DIM] = b[i];


// Stampo l'array AB
printf("\narray a.b = ");
for (i = 0; i < DIM2; i++){
     printf("%d ", ab[i]);
}

printf("\n");
```

# Esercizio 1: numeri primi

```
// NUMERI PRIMI
// ------------------------------------

printf("I numeri primi inseriti sono: ");

for (i = 0; i < DIM2; i++){

    primo = 1; // Inizializzo la variabile

    for (j = 2; j < ab[i]; j++)
        if ((ab[i] % j) == 0 ){
            primo = 0;
            break;
        }

    if (primo)
        printf("%d ", ab[i]);
}

printf("\n");
```

# Esercizio 1: massimo e minimo

```
// MAX e MIN
// ---------------------------------

int max = 0;
int min = 2147483647;   //2^(n-1) - 1 :::: CPL2 max value
                              //n = 32 poiché int è rappresentato
                              //con 32 bit.

for (i = 0; i < DIM2; i++){

      if (ab[i] > max)
            max = ab[i];

      if (ab[i] < min)
            min = ab[i];

}

printf("MIN = %d\n", min);
printf("MAX = %d\n", max);
```

# Esercizio 1: massimo e minimo

```
// TROVARE DOVE SONO MAX e MIN
// -----------------------------------


bool max_in_array_a = 0;
bool min_in_array_a = 0;

for (i = 0; i < DIM; i++){
    if (a[i] == max)
        max_in_array_a = 1;

    if(a[i] == min)
        min_in_array_a = 1;
}

printf("\n");

if (max_in_array_a)
    printf("Il valore massimo %d e' nell'array a", max);
else
    printf("Il valore massimo %d e' nell'array b", max);

printf("\n");

if (min_in_array_a)
    printf("Il valore minimo %d e' nell'array a", min);
else
    printf("Il valore minimo %d e' nell'array b", min);


printf("\n");
```

# Esercizio 1: array riversato

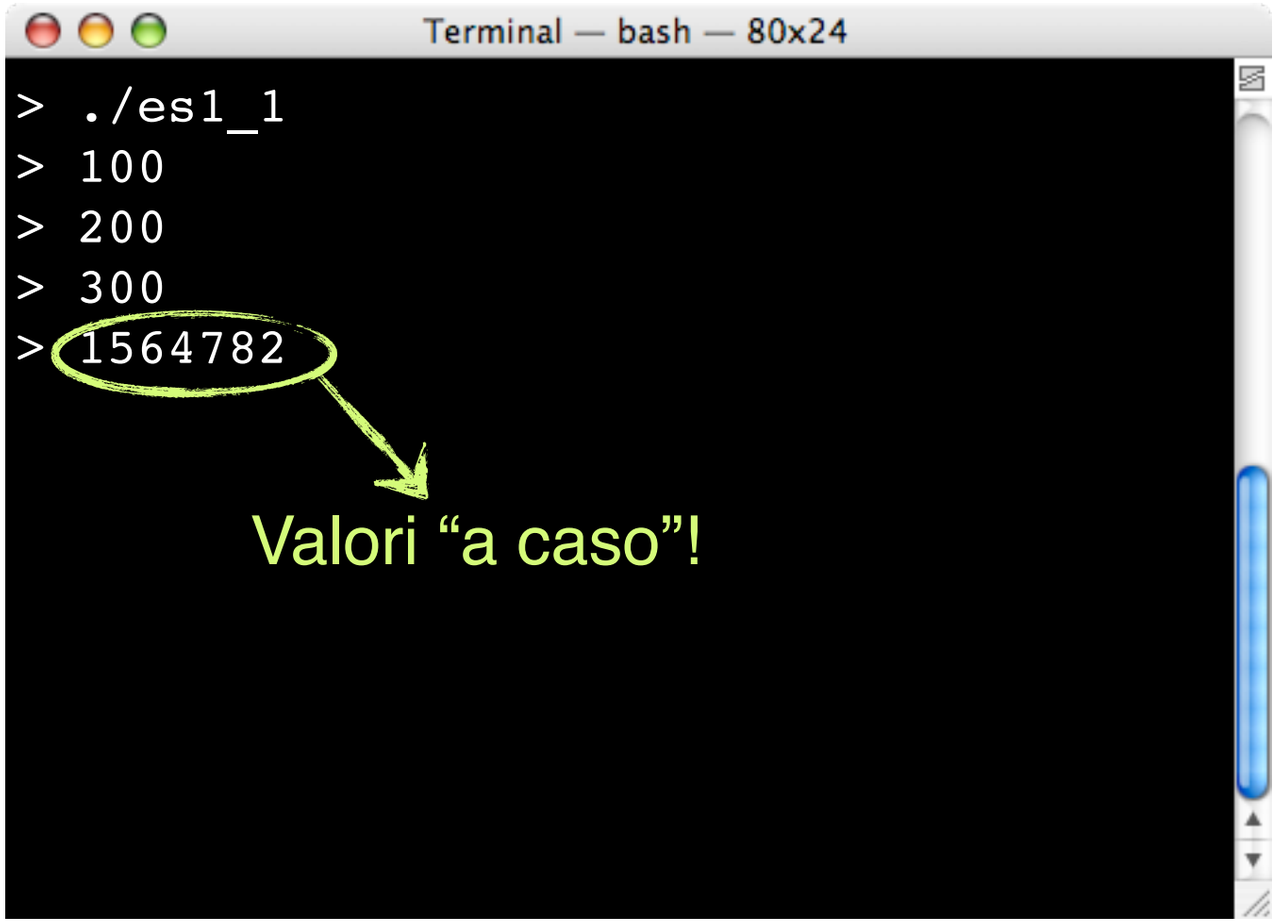
```
// ARRAY RIVERSATO
// ---------------------------------

for (i = 0; i < DIM2; i++)
     ab_r[DIM2 - i - 1] = ab[i];
```

```
printf("\narray (a.b)_r = ");
for (i = 0; i < DIM2; i++){
     printf("%d ", ab_r[i]);
}

printf("\n");
printf("\n");
```

# Esercizio 2: Array… l'inizializzazione (a)

- Quale è l'ouput del seguente codice?

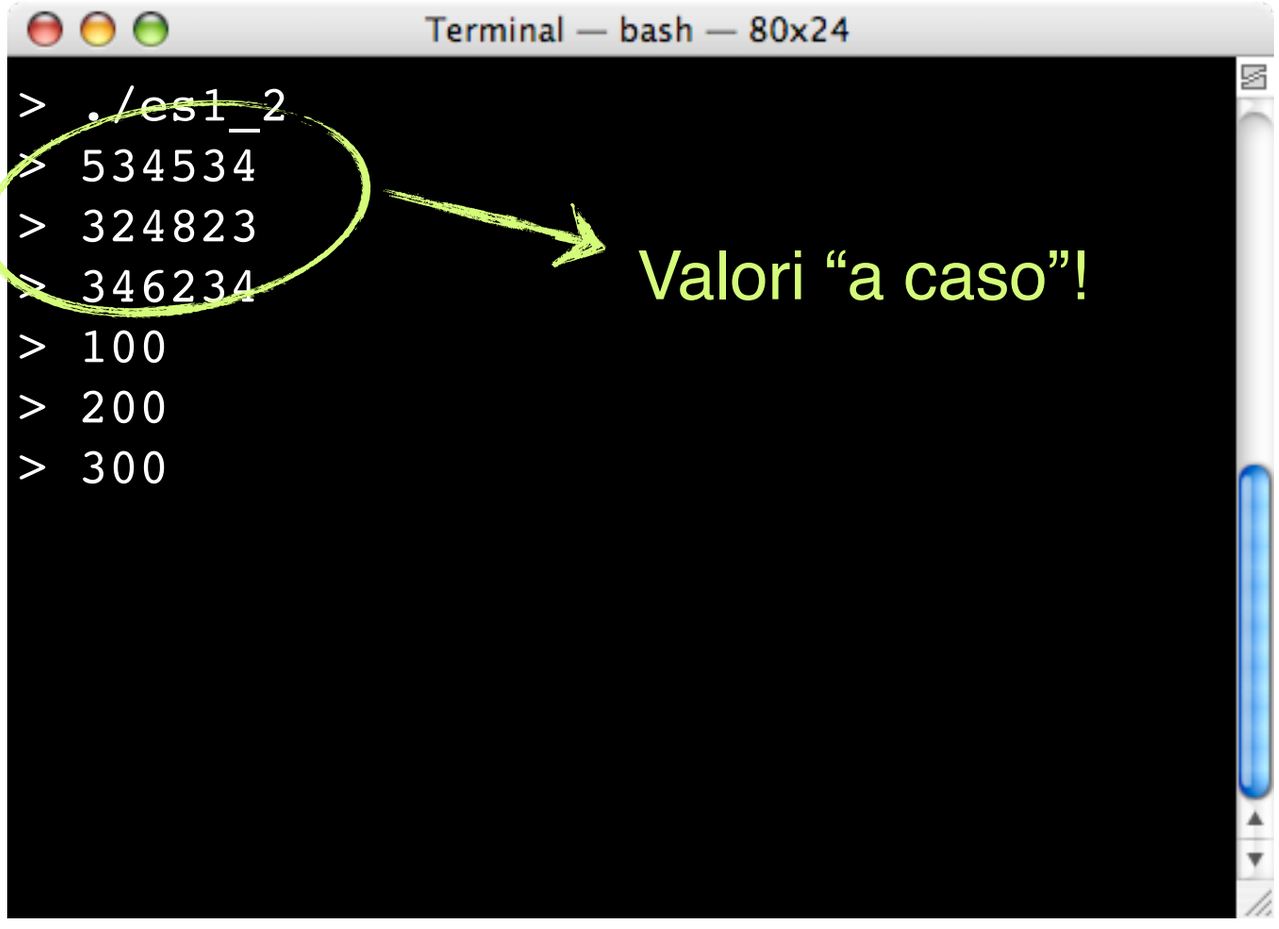
```
#include <stdio.h>

int main(){

    int arr[3] = {100,200,300};

    printf("%d\n", arr[0]);
    printf("%d\n", arr[1]);
    printf("%d\n", arr[2]);
    printf("%d\n", arr[3]);
    return 0;
}
```



# Esercizio 2: Array… l'inizializzazione (b)

- Quale è l'ouput del seguente codice?

```c
#include <stdio.h>

int main(){

    int arr[3];

    printf("%d\n", arr[0]);
    printf("%d\n", arr[1]);
    printf("%d\n", arr[2]);

    arr[0] = 100;
    arr[1] = 200;
    arr[2] = 300;

    printf("%d\n", arr[0]);
    printf("%d\n", arr[1]);
    printf("%d\n", arr[2]);

    return 0;
}
```



# Esercizio III: variabili e funzioni (a)

- Quale e' l'output del seguente programma?

```c
#include <stdio.h>

int funzione();

int main()
{

     int a = 10;
     printf("Il valore di a e' %d\n", a);
     funzione();
     printf("Ora il valore di a e' %d\n", a);

}

int funzione()
{
     a = 3;
}
```

NESSUNO!! NON COMPILA!

```
MacBook-Pro-di-Alessandro-Nacci:Desktop alessandronacci$ gcc scope.c -o scope
scope.c:3:15: error: expected ';' after top level declarator
int funzione()
              ^
              ;
scope.c:17:2: error: use of undeclared identifier 'a'
        a = 3;
        ^
2 errors generated.
MacBook-Pro-di-Alessandro-Nacci:Desktop alessandronacci$
```

# Esercizio III: variabili e funzioni (b)

- Quale e' l'output del seguente programma?

```
MacBook-Pro-di-Alessandro-Nacci:Desktop alessandronacci$ ./scope
Il valore di a e' 10
Ora il valore di a e' 10
MacBook-Pro-di-Alessandro-Nacci:Desktop alessandronacci$
```

```c
#include <stdio.h>

int funzione();

int main()
{
    int a = 10;
    printf("Il valore di a e' %d\n", a);
    funzione();
    printf("Ora il valore di a e' %d\n", a);

}

int funzione()
{
    int a;
    a = 3;
}
```

# Esercizio III: variabili e funzioni (c)

- Quale e' l'output del seguente programma?

```
#include <stdio.h>

int funzione();

int main()
{
      int a = 10;
      printf("Il valore di a e' %d\n", a);
      funzione(a);
      printf("Ora il valore di a e' %d\n", a);

}

int funzione(int a)
{
      a = 3;
}
```

```
MacBook-Pro-di-Alessandro-Nacci:Desktop alessandronacci$ ./scope
Il valore di a e' 10
Ora il valore di a e' 10
MacBook-Pro-di-Alessandro-Nacci:Desktop alessandronacci$
```

# Esercizio III: variabili e funzioni (d)

- Quale e' l'output del seguente programma?

```
#include <stdio.h>

int funzione();

int main()
{
    int a = 10;
    printf("Il valore di a e' %d\n", a);
    a = funzione(a);
    printf("Ora il valore di a e' %d\n", a);

}

int funzione(int a)
{
    a = 3;
    return a;
}
```

```
MacBook-Pro-di-Alessandro-Nacci:Desktop alessandronacci$ ./scope
Il valore di a e' 10
Ora il valore di a e' 3
MacBook-Pro-di-Alessandro-Nacci:Desktop alessandronacci$
```

**Tutte il materiale sarà disponibile sul mio sito internet!**

www.alessandronacci.it