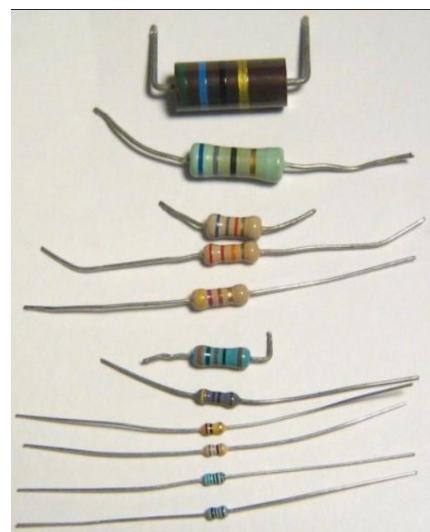


**Dispense di Laboratorio
ELETTRONICA ED AUTOMAZIONE**

ARGOMENTO: CODICE DEI COLORI - VALORE NOMINALE - TOLLERANZA

Si riportano i simboli elettrici che identificano il componente elettrico “Resistenza” secondo due normative tecniche la ANSI e la IEC. Nella realizzazione degli schemi elettrici andremo ad indicare quale normativa impiegare e tutti i simboli saranno coerenti alla normativa scelta.

Symbol (ANSI Y32.2)	Symbol (IEC 60617)
	



L’unità di misura della resistenza elettrica sono gli OHM indicati con il seguente simbolo: Ω

Nella realizzazione di circuiti elettronici si utilizzano:

Ohm Ω

sopra i 1000Ω si passa ai multipli.

KiloOhm $K\Omega$

MegaOhm $M\Omega$

La resistenza non ha polarità cioè è priva di un verso. Nel montaggio dei circuiti è indifferente come la si installa.

METODO DELLE BANDE DI COLORE:

Tale metodo ci consente di determinare il Valore nominale di Resistenza e il valore della sua Tolleranza.

Il valore reale della resistenza (ottenuta misurandola) sarà entro una tolleranza riferita al valore nominale.

Determinazione della resistenza nominale:

- 1) Orientare la resistenza posizionando la banda “Tolleranza” alla vostra destra.
Tutte le bande tranne quella di tolleranza sono tra loro equidistanti.
Nel caso di resistenza di piccole dimensioni aiutarsi consultando la tabella con i colori.
La lettura avviene da sinistra verso destra.
- 2) Annotare nel quaderno il colore delle bande.
Associare ad ogni banda il numero riportato nell'abaco:

resistenza con 4 bande

1°CIFRA - 2°CIFRA - MOLTIPLICATORE - TOLLERANZA %

resistenza con 5 bande

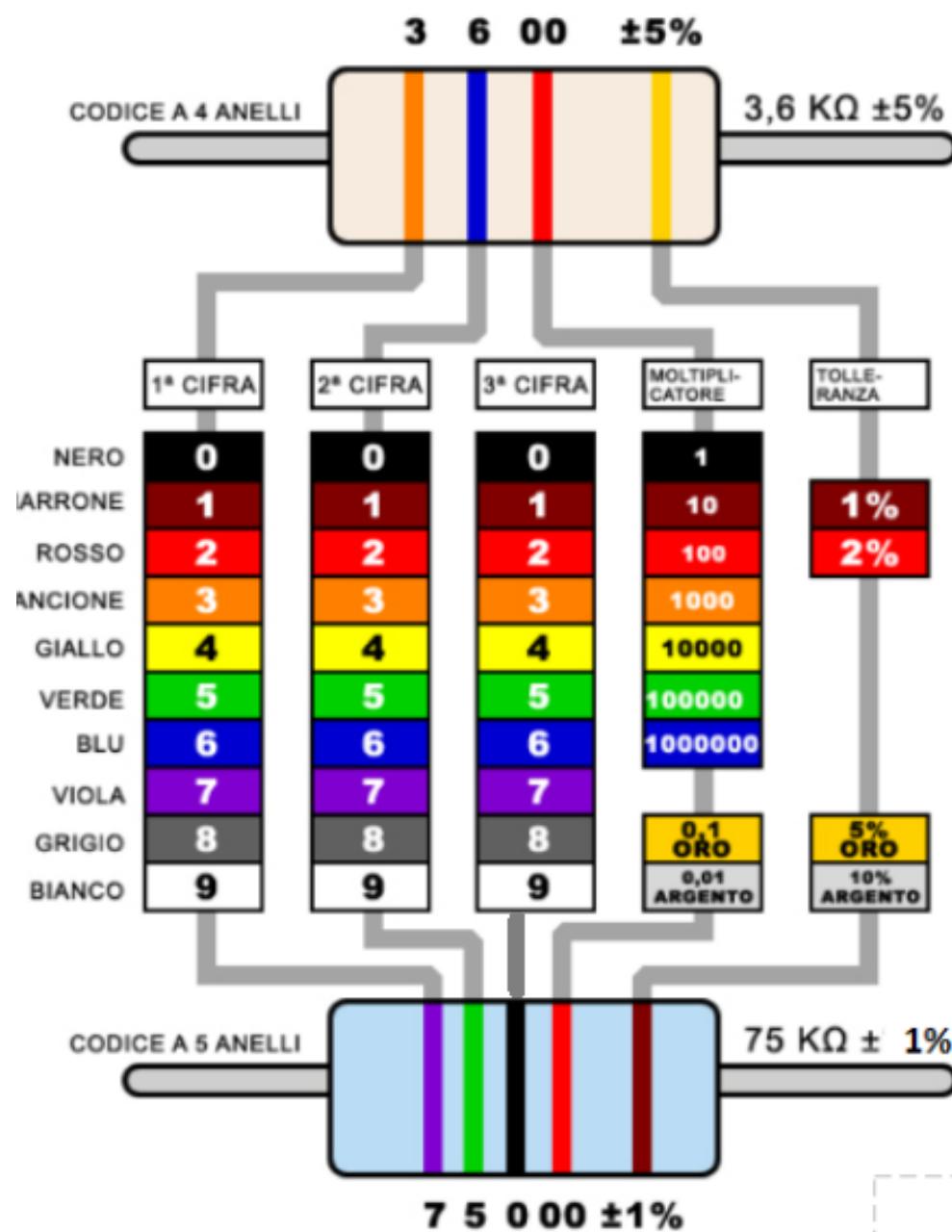
1°CIFRA - 2°CIFRA - 3°CIFRA - MOLTIPLICATORE - TOLLERANZA%

- Il valore nominale della resistenza si determina:

Unire la 1 e 2 cifra (eventualmente la 3) e moltiplicare il numero ottenuto per il moltiplicatore

- Il valore della tolleranza si determina:
Moltiplicando la tolleranza% per il valore nominale.

ABACO DELLE BANDE DEI COLORI



Esempio con 4 bande:

numero anelli: 4

colore del primo anello = arancio = 1° cifra = 3

colore del secondo anello = blu = 2° cifra = 6

colore del terzo anello= rosso = moltiplicatore= 100

colore del quarto anello= oro = tolleranza percentuale = 5%

Per determinare il valore nominale della resistenza “Rnominale”:

- unire le cifre del 1°, 2° anello = 36
- moltiplicare il numero ottenuto per il valore del moltiplicatore

$$R_{\text{nominale}} = 36 \times 100 = 3600 \Omega = 3,6 \text{ k}\Omega$$

Per determinare il valore della tolleranza:

$$\text{Tolleranza} = \text{tolleranza percentuale} \times R_{\text{nominale}} =$$

$$5\% \times 3600\Omega = 5 \times 3600/100 = 180\Omega$$

Definizione: Il valore reale della resistenza è quello effettivo ottenuto tramite misure con lo strumento.

Il valore reale deve essere compreso entro la tolleranza riferita al valore nominale.

$$R_{\min} < R_{\text{reale}} < R_{\max}$$

$$R_{\max} = R_{\text{nom.}} + \text{tolleranza} = 3600 + 180 = 3780 \Omega = 3,78 \text{ K}\Omega$$

$$R_{\min} = R_{\text{nom.}} - \text{tolleranza} = 3600 - 180 = 3420 \Omega = 3,42 \text{ K}\Omega$$

$$3,42 \text{ K}\Omega < R_{\text{reale}} < 3,78 \text{ K}\Omega$$

Esempio con 5 bande:

numero anelli: 5

colore del primo anello = marrone = 1° cifra = 1

colore del secondo anello = rosso = 2° cifra = 2

colore del terzo anello = nero = 3° cifra = 0

colore del quarto anello= marrone = moltiplicatore= 10

colore del quinto anello= oro = tolleranza percentuale = 5%

Per determinare il valore nominale della resistenza “Rnominale”:

- unire le cifre del 1°, 2°, 3° anello = 120
- moltiplicare il numero ottenuto per il valore del moltiplicatore

$$R_{\text{nominale}} = 120 \times 10 = 1200 \Omega = 1,2 \text{ k}\Omega$$

Per determinare il valore della tolleranza:

$$\text{Tolleranza} = \text{tolleranza percentuale} \times R_{\text{nominale}} =$$

$$5\% \times 1200 \Omega = 5 \times 1200 / 100 = 60 \Omega$$

$$R_{\text{min}} < R_{\text{reale}} < R_{\text{max}}$$

$$R_{\text{max}} = R_{\text{nom.}} + \text{tolleranza} = 1200 + 60 = 1260 \Omega$$

$$R_{\text{min}} = R_{\text{nom.}} - \text{tolleranza} = 1200 - 60 = 1140 \Omega$$

$$1140 \Omega < R_{\text{reale}} < 1260 \Omega$$

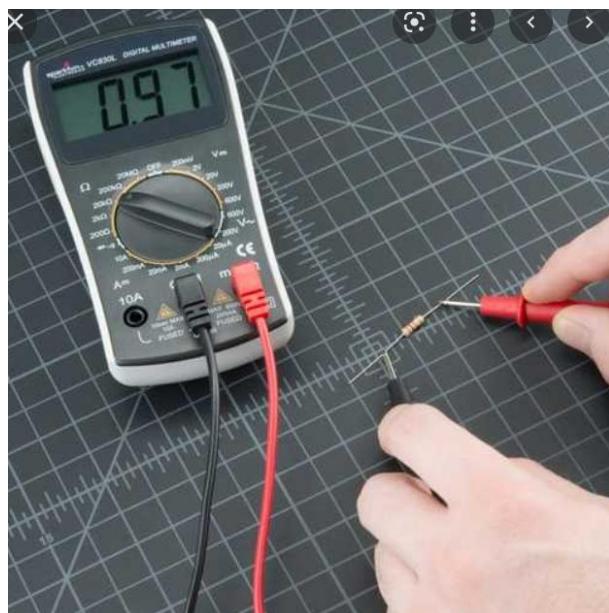
ESERCITAZIONE N°1: MISURA DIRETTA DELLA RESISTENZA

Definizione: La misura diretta del valore reale della resistenza consiste semplicemente nel misurare la resistenza tramite l'impiego del multmetro impostato nella funzionalità Ohmetro.

Il multmetro è uno strumento che compie misure di tensione, corrente e resistenza ed altre.

Se impostato per misurare la resistenza è denominato Ohmetro, per misurare le tensioni Voltmetro e per la misura di correnti Amperometro.

- 1) Assegnate le resistenze, con il **codice dei colori**, determinare il valore nominale, il valore minimo e massimo della resistenza.
- 2) Con il metodo denominato “misura diretta della resistenza” ossia tramite misura con multmetro determinare il valore reale della resistenza e verificare che tale valore sia compreso entro la tolleranza dichiarata dal fornitore del componente elettrico.



Resistenza n° _____ consegnata dal prof.

numero anelli: _____

colore del primo anello = _____ = 1° cifra = _____

colore del secondo anello = _____ = 2° cifra = _____

colore del terzo anello = _____ = 3° cifra = _____

colore del quarto anello= _____ = moltiplicatore= _____

colore del quinto anello= _____ = tolleranza percentuale = _____

Calcolo del valore nominale della resistenza “Rnominale”:

- unire le cifre del 1°, 2° ed eventualmente del 3° anello _____
- moltiplicare il numero per il valore del moltiplicatore _____

Calcolo della tolleranza:

Tolleranza percentuale = _____

Tolleranza = tolleranza percentuale x Rnominale = _____

Rmax= Rnom. + tolleranza= _____

Rmin= Rnom. - tolleranza= _____

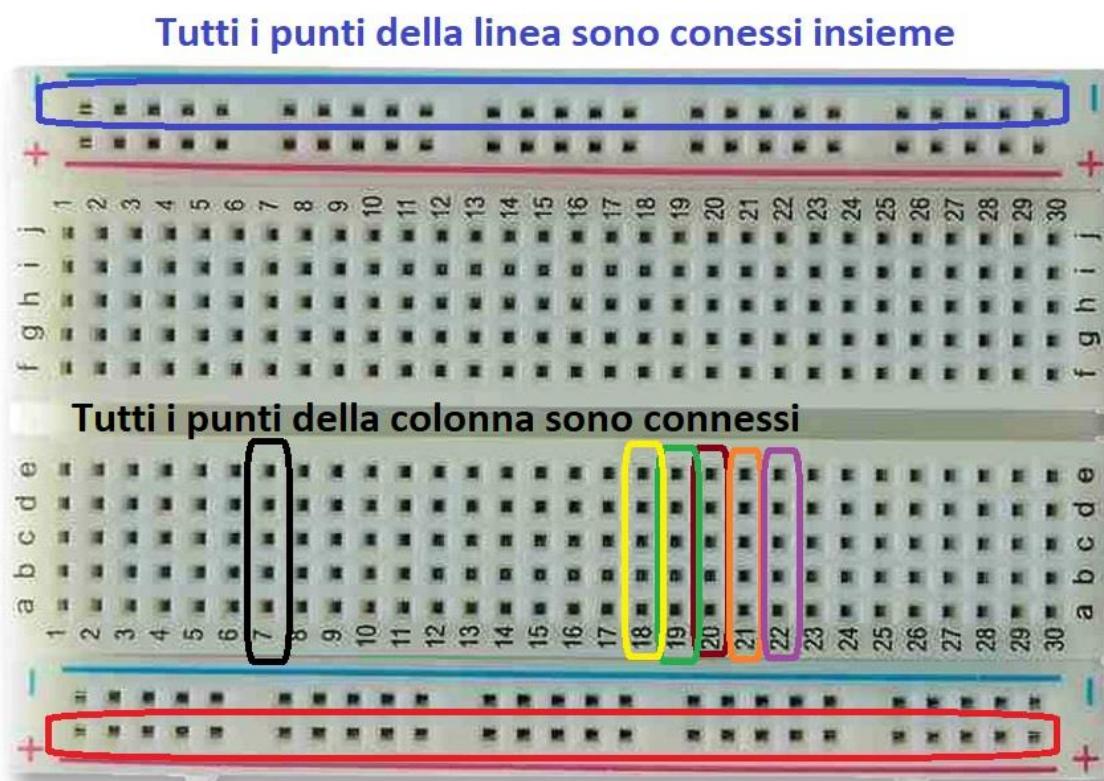
Reale (misurato) = _____

Reale è compreso tra Rmin e Rmax? SI NO

LA BREADBOARD

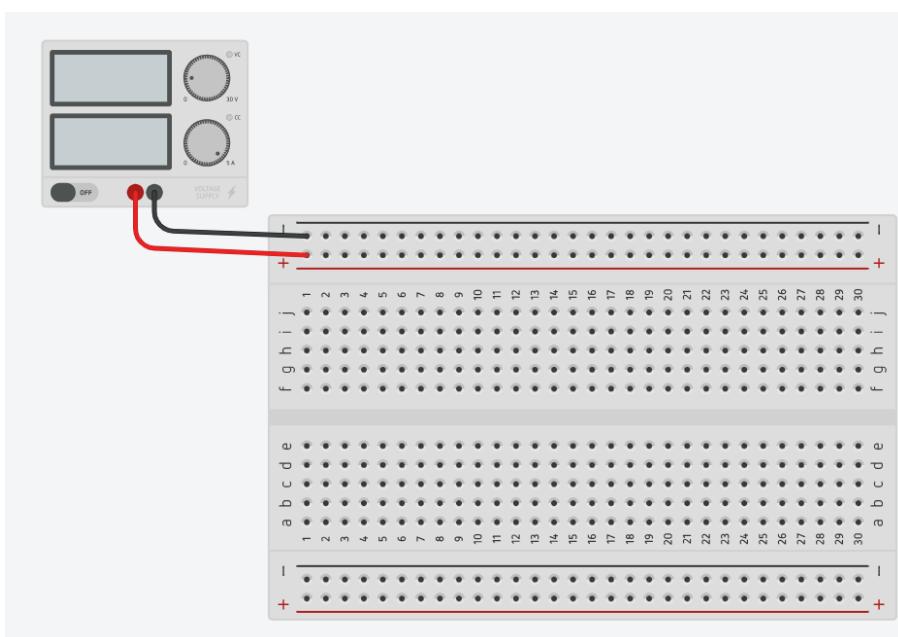
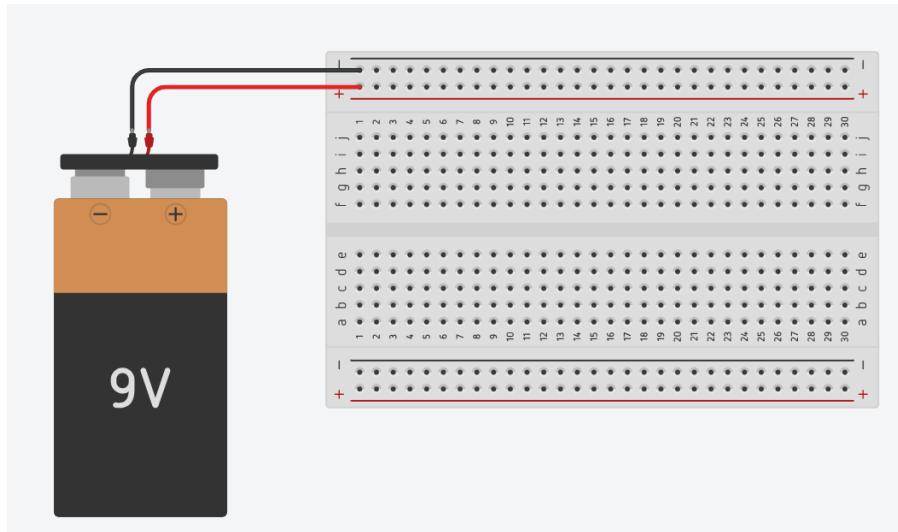
1) Descrizione della breadboard:

La basetta breadboard è un supporto molto comodo per la realizzazione di prototipi. Si tratta di una basetta con diverse serie di fori, separate da una scanalatura isolante. La scanalatura centrale isola elettricamente la parte sopra dalla parte sotto della breadboard. I fori sono disposti orizzontalmente e verticalmente e collegati all'interno da elementi conduttrici, che assicurano il collegamento elettrico.



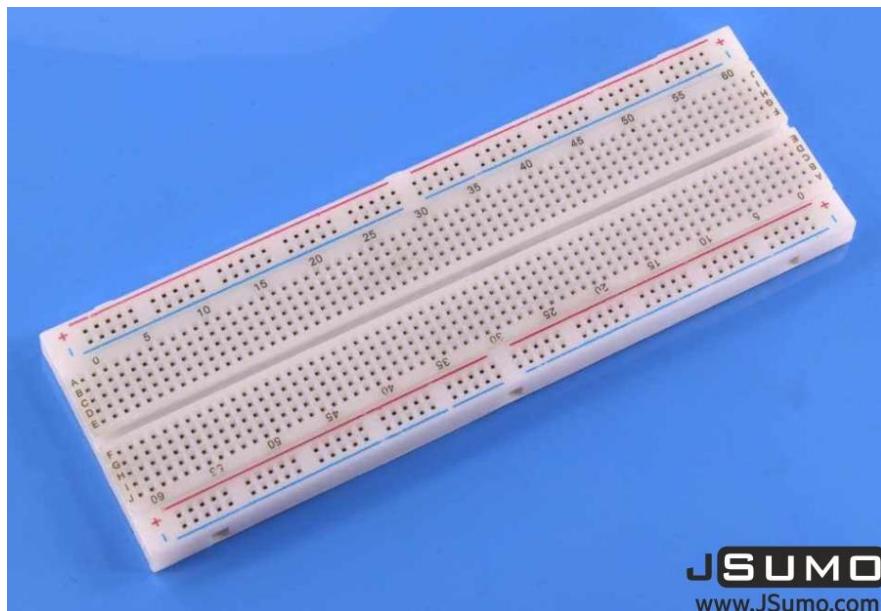
2) Linee di alimentazione rosse e blu della breadboard

L'alimentazione al circuito è collegata alle linee blu e rosse.



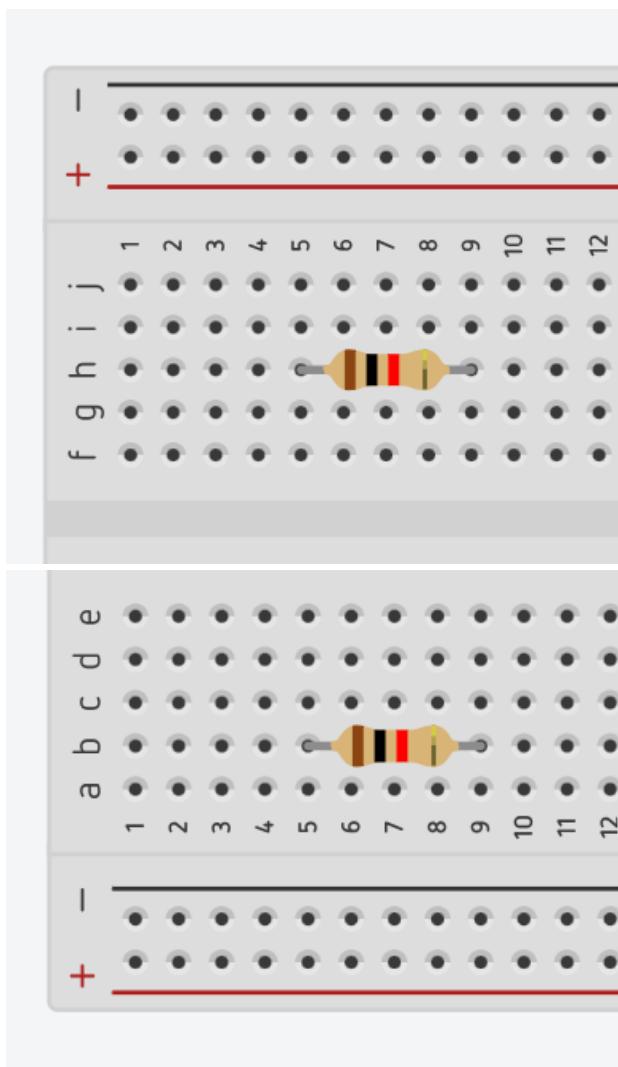
Attenzione in commercio ci sono breadboard con linee di

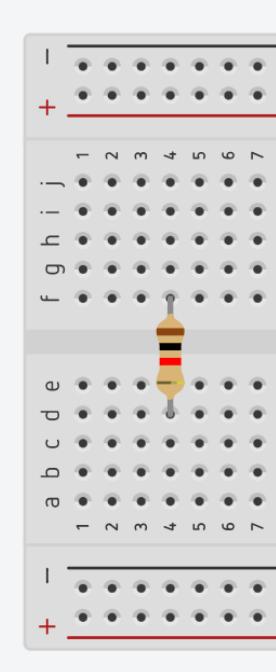
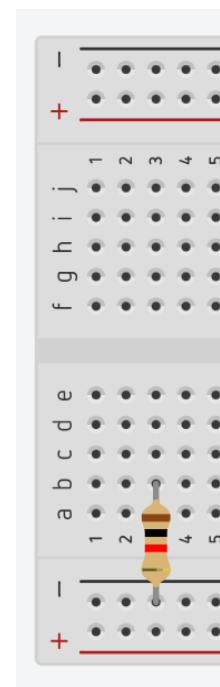
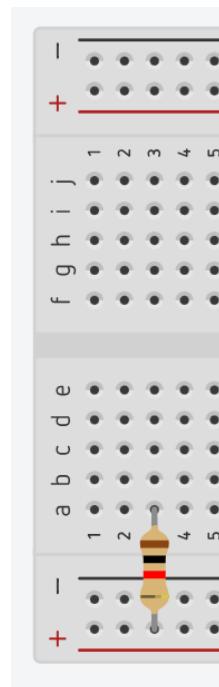
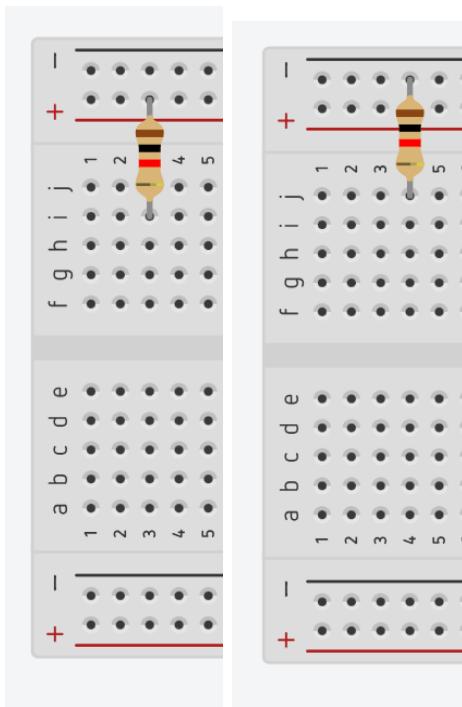
alimentazione interrotte (vedere figura). Per riportarsi alla condizione di continuità elettrica si impiegano quattro cavetti elettrici che collegano le quattro linee.



3) Posizionare correttamente le resistenze sulla breadboard:

IN ORIZZONTALE SUL CAMPO SUPERIORE O INFERIORE

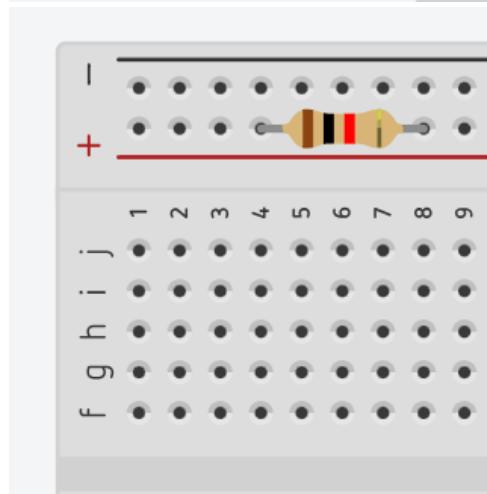
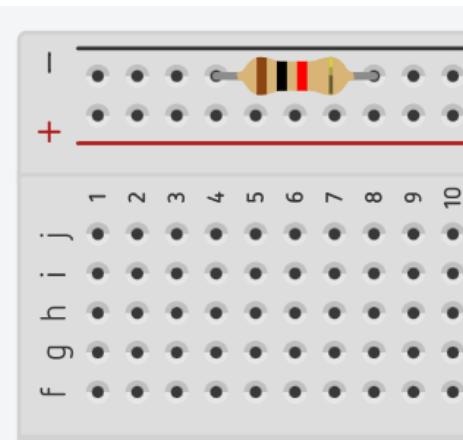
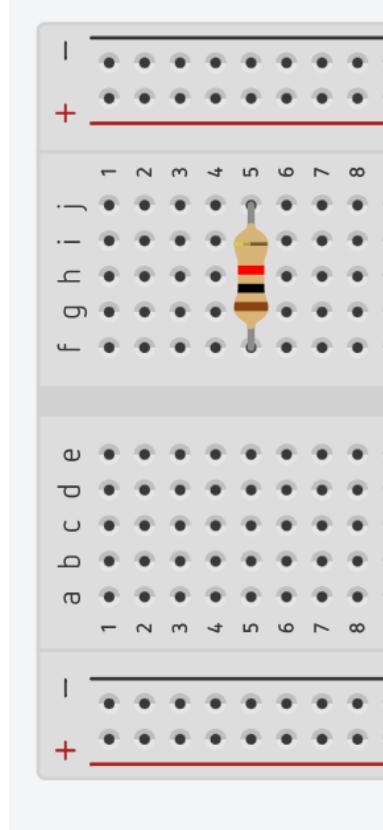


IN VERTICALE CHE COLLEGA IL CAMPO SUPERIORE CON IL CAMPO INFERIORE**IN VERTICALE CHE COLLEGA LA LINEA DI ALIMENTAZIONE POSITIVA O NEGATIVA CON IL CAMPO**

4) Configurazioni Errate e pericolose!!!!!!

Le configurazioni pericolose sono quelle che vedono la resistenza in parallelo con i collegamenti elettrici interni.

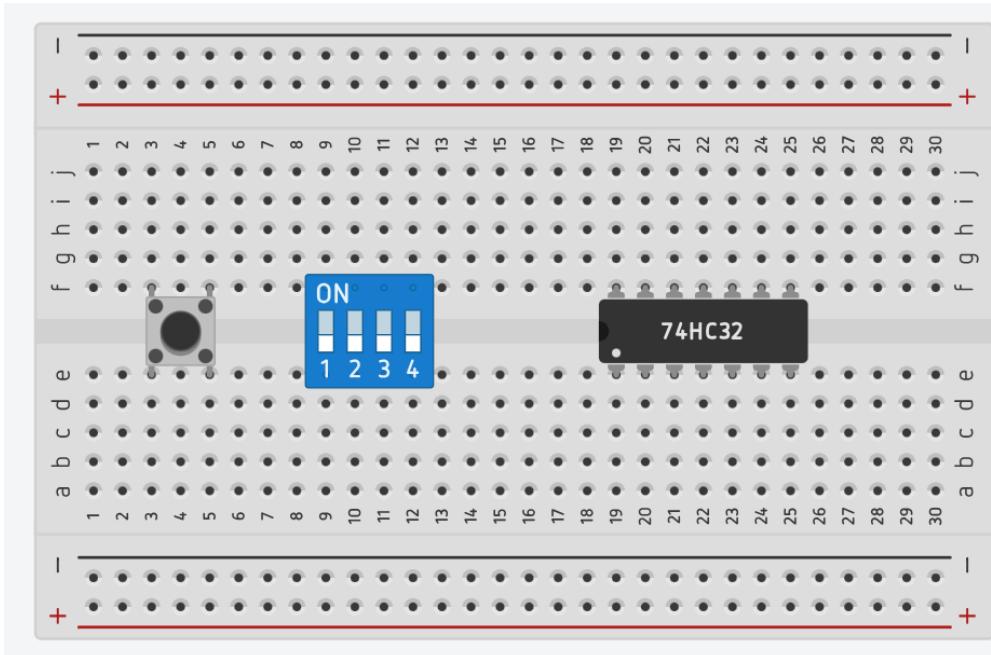
IN VERTICALE SULLO STESSO CAMPO e IN ORIZZONTALE LUNGO LE LINEE DI ALIMENTAZIONE



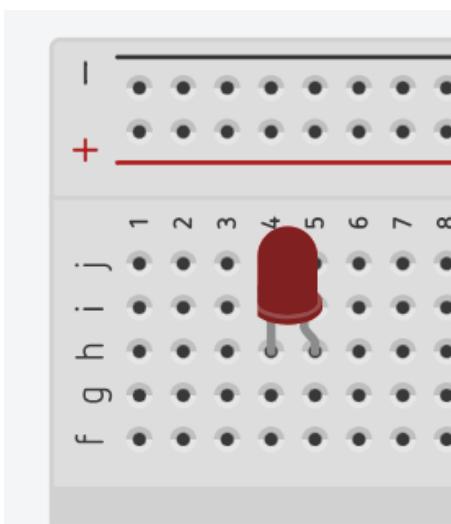
5) Altri componenti:

La distanza tra i fori è unificata e pari a 1/10 di pollice, per poter inserire i componenti elettronici dotati di piedini.

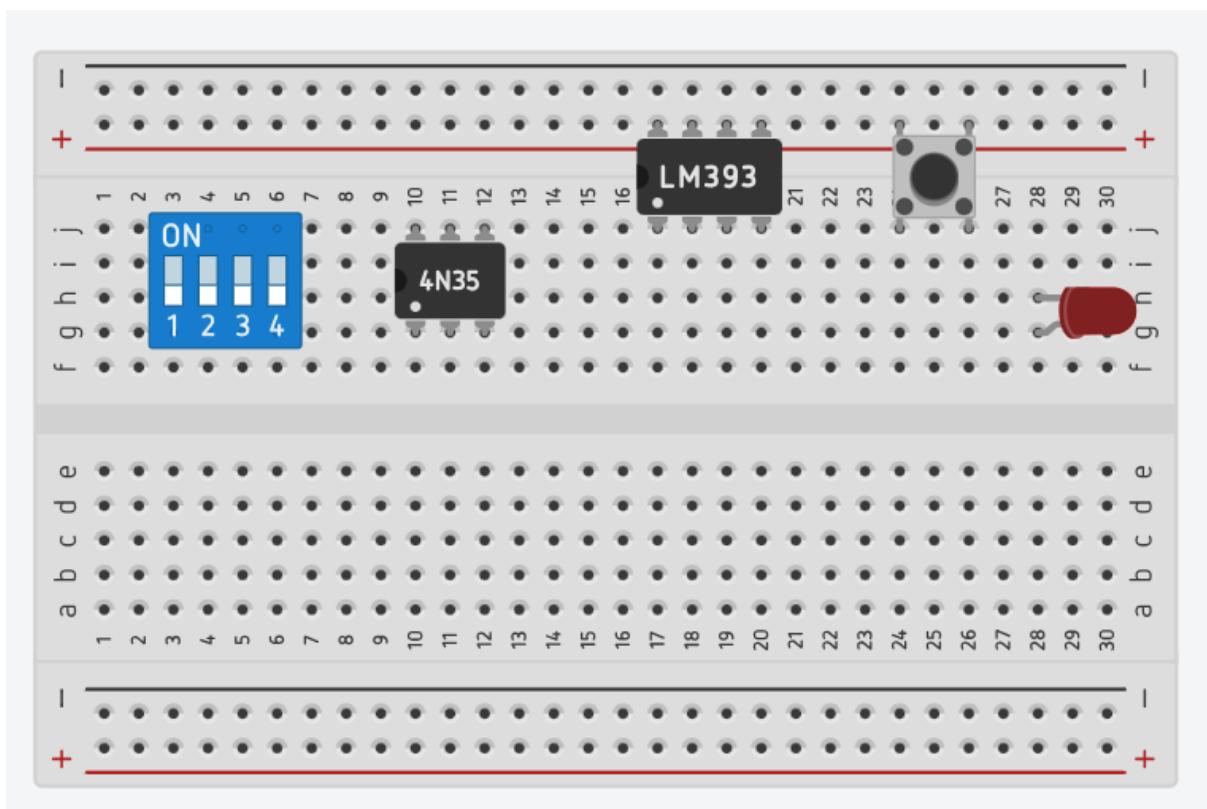
Nella figura si vedono i componenti che verranno impiegati durante l'anno scolastico, pulsanti e interruttori e circuiti integrati devono essere posizionati a scavalco tra i due campi isolati tra di loro.



Diodo led: stesse regole delle resistenze.



**FIGURA CON COMPONENTI POSIZIONATI IN
MODO ERRATO**



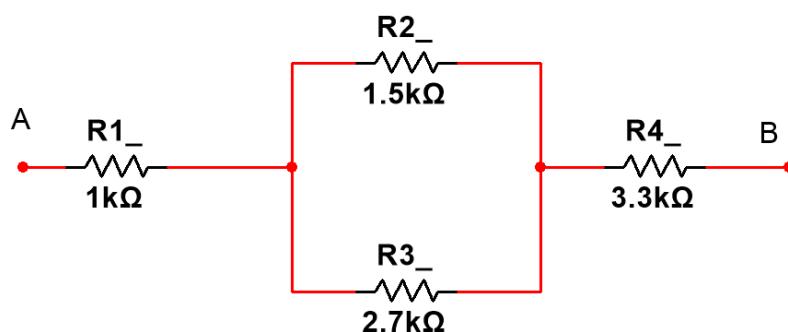
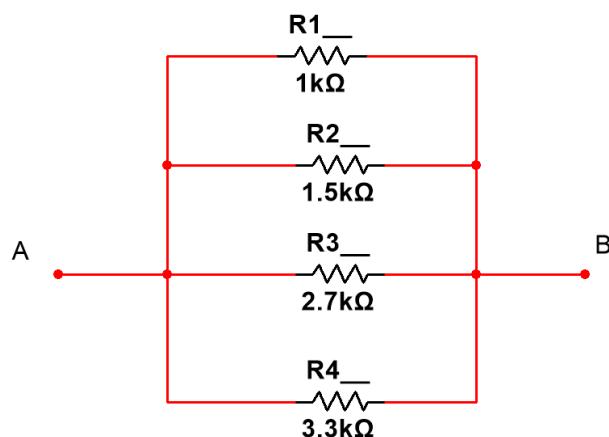
ESERCITAZIONE N°2: MISURA DIRETTA DELLA RESISTENZA PER ALCUNE CONFIGURAZIONI CIRCUITALI

- Multisim: Disegnare gli schemi sotto riportati e misurare la resistenza

Equivalente tra i morsetti AB.

- Breadboard: montare i circuiti e misurare la resistenza Equivalente con il metodo diretto.
- Nel quaderno realizzare una tabella di confronto confrontando i risultati di Multisim con quelli ottenuti con il metodo diretto.

A seguire si riportano le configurazioni da analizzare:

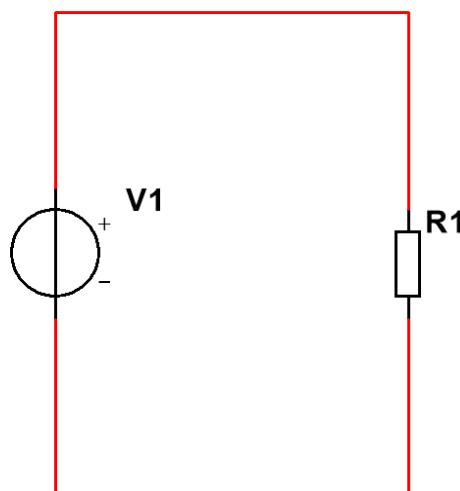


ESERCITAZIONE N°3: MISURA INDIRETTA DELLA RESISTENZA O DETTO METODO VOLTAMPEROMETRICO.

Si debba determinare la resistenza ai capi AB di un circuito. Il metodo voltamperometrico consiste nell'applicare ai capi un valore di tensione scelta dall'operatore tramite l'alimentatore e misurare la corrente uscente dall'alimentatore stesso.

Consegnata una resistenza di valore incognito seguire i seguenti passaggi.

- 1) montaggio del circuito su breadboard come da schema,
- 2) applicare una tensione V_1 al circuito di 5V tramite alimentatore
- 3) misurare la corrente uscente dal generatore
- 4) determinare la resistenza con il metodo indiretto applicando la legge di Ohm.

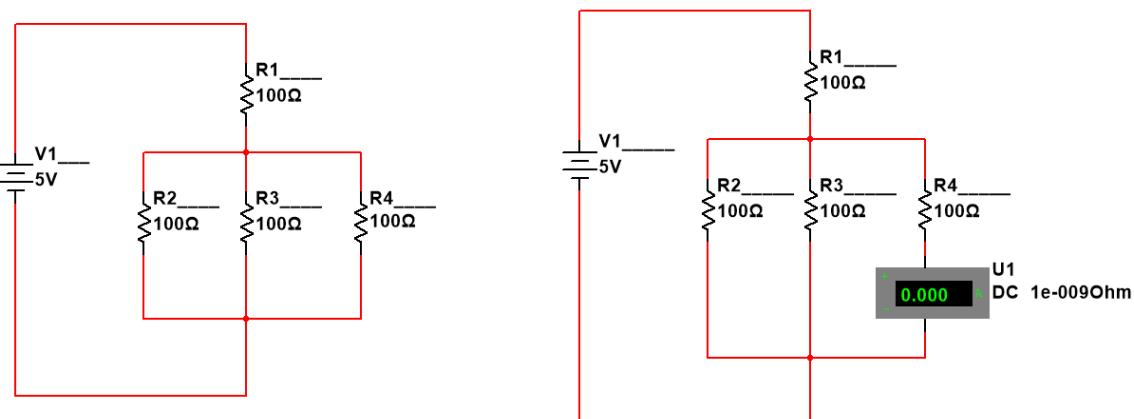


ESERCITAZIONE N°4: MISURA INDIRETTA DELLA RESISTENZA O DETTO METODO VOLTAMPEROMETRICO.

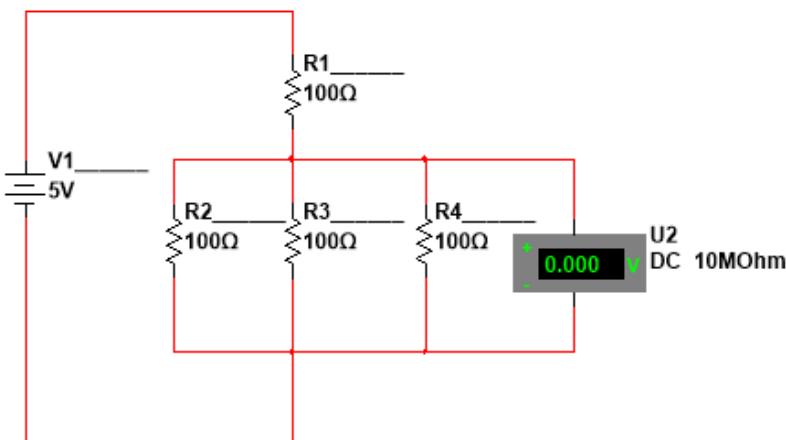
Eseguire la misura indiretta di resistenza equivalente ai capi AB per i circuiti visti nell'esercizio n°2 sempre applicando 5V. Confrontare i risultati ottenuti con il metodo per via diretta.

ESERCITAZIONE N°5: MISURE DI TENSIONE E CORRENTE

Teoria: Per compiere una misura di corrente il multmetro impostato nella funzione di Amperometro deve essere inserito in serie all'elemento attraversato da tale corrente. Esempio dato il circuito in figura si voglia misurare la corrente che attraversa R4, allora si inserisce in serie al componente l'amperometro.

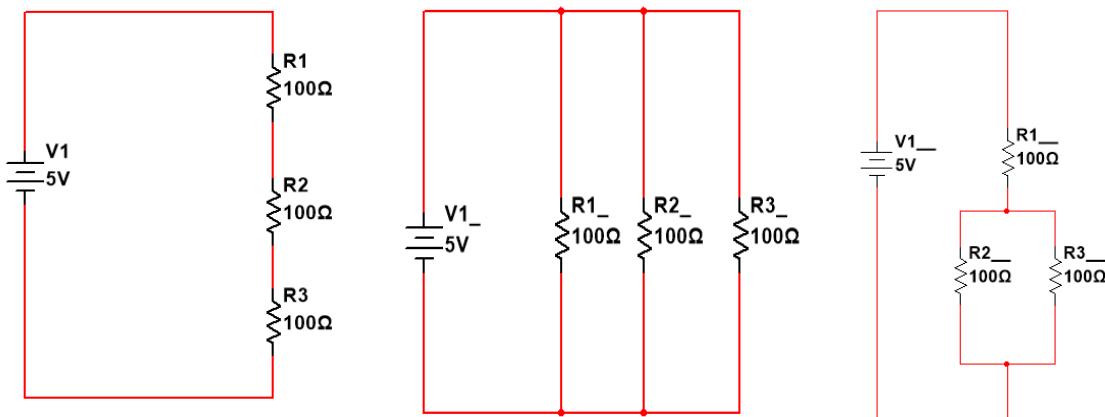


Teoria: Per compiere una misura di tensione il multmetro impostato nella funzione di Voltmetro deve essere inserito in parallelo all'elemento. Esempio dato il circuito in figura si voglia misurare la tensione ai capi di R4, allora si inserisce in parallelo al componente il voltmetro.

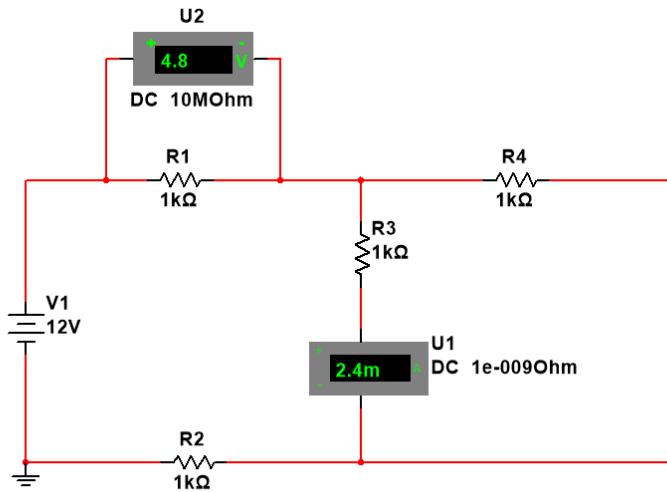


Dati i circuiti in figura:

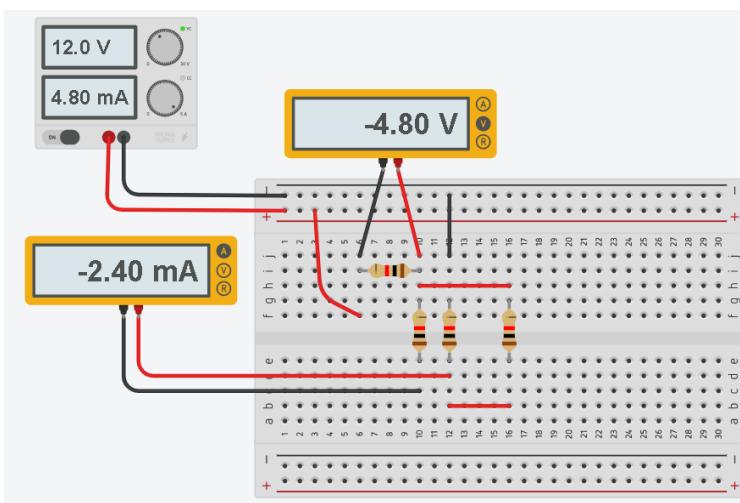
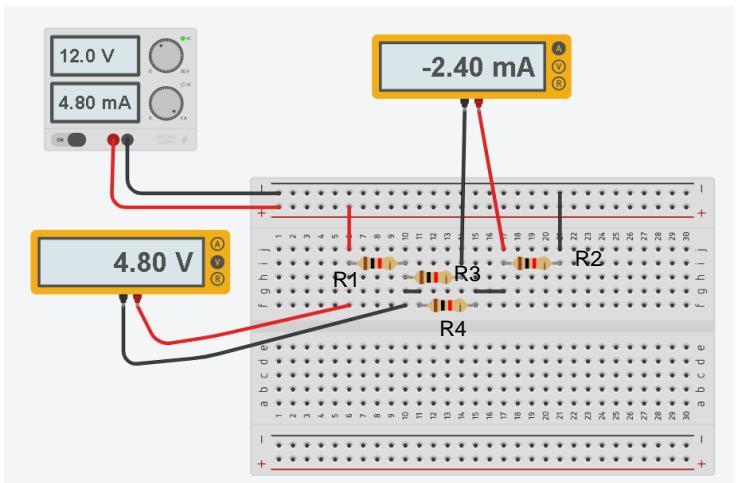
- Realizzazione in multisim degli schemi, inserendo gli strumenti per misurare la corrente che attraversa la resistenza R2, e la tensione a i capi di R1.
- Disegnare nel quaderno lo schema su breadboard.
- Montaggio dello schema su breadboard con relative misure di tensione e corrente.

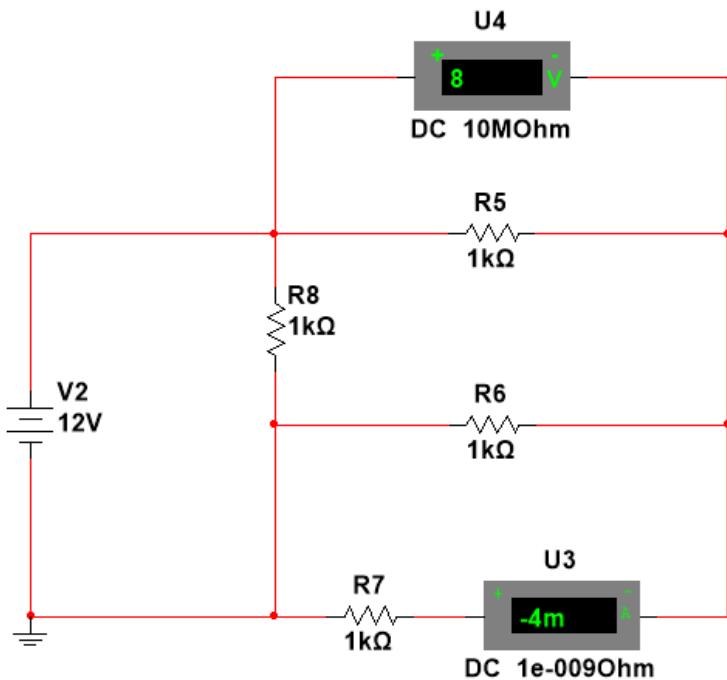


Esempi tipo compito:

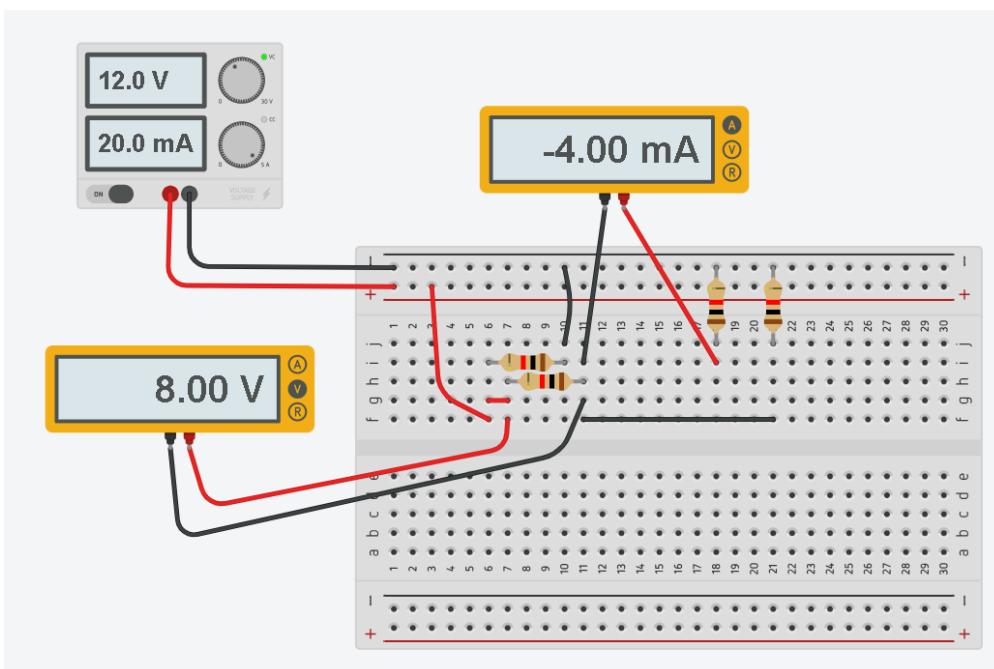


Soluzioni possibili:



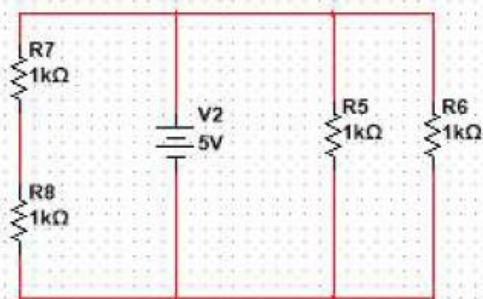


Soluzione possibile:

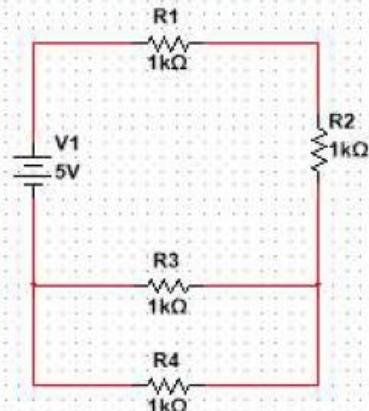


ESERCITAZIONE N°6: Montare i seguenti circuiti su Breadboard

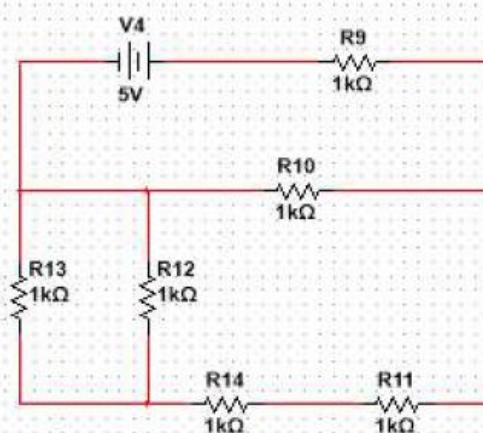
CIRCUITO N°1



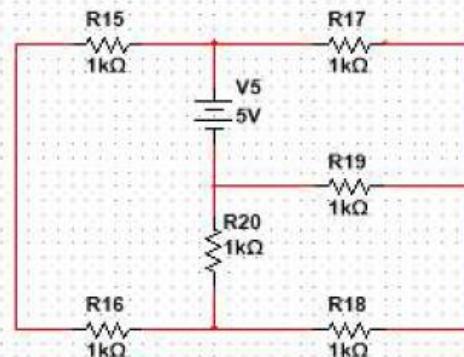
CIRCUITO N°2



CIRCUITO N°3



CIRCUITO N°4



ESERCITAZIONE N°5:

- Prova di laboratorio

Verifica del 1° Principio di Kirchhoff

Il circuito fornito presenta 2 nodi, A e B. Con i dati forniti, effettuare le seguenti operazioni:

- 1) - Attraverso propri calcoli, trovare dapprima la f.e.m di alimentazione E_1 . Nel fare questo si devono necessariamente calcolare anche le 2 correnti incognite, I_1 e I_2 , pertanto verificare al nodo A, che: $I_1 = I_2 + I_3$. (1° Principio di Kirchhoff). Tutto questo con propri calcoli sul quaderno.
- 2) Implementare al Multisim o altro software di simulazione, il circuito dando ad E_1 il valore calcolato al punto 1, e, inserendo opportunamente gli amperometri, misurare le tre correnti, verificando appunto il 1° Principio di Kirchhoff, $I_1 = I_2 + I_3$.
- 3) Dopo essersi disegnato lo schema di cablaggio, manualmente, con TinkerCad o Multisim, implementare il circuito su breadboard.
- 4) Redigere al PC una relazione, documentata con schemi vari, su tutto il lavoro svolto.

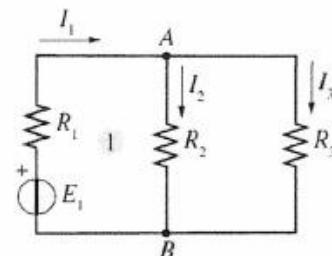
DATI

$$I_3 = 1,66 \text{ mA}$$

$$R_1 = 670 \Omega$$

$$R_2 = 2 \text{ k}\Omega$$

$$R_3 = 4 \text{ k}\Omega$$



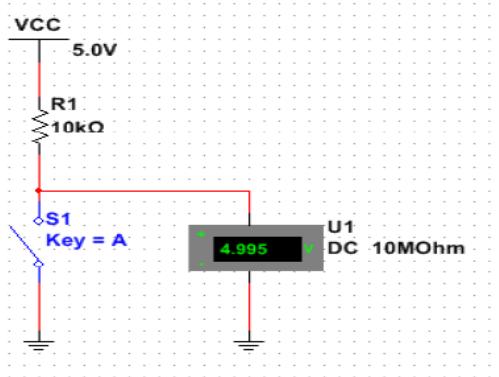
ESERCITAZIONE N°6: SEGNALI DIGITALI IN INGRESSO

Realizzare in Tinkercad e su Breadboard i seguenti due schemi.

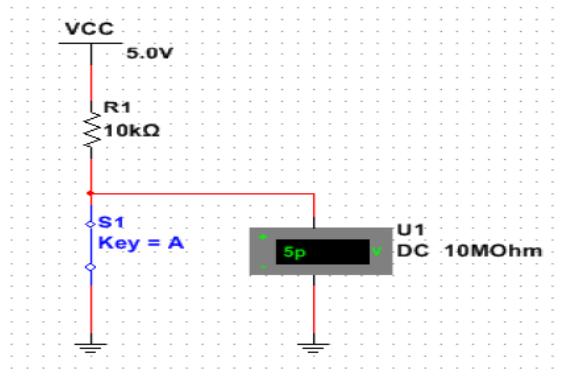
Variando la posizione dello switch tra ON e OFF, misurare la tensione al terminale come in figura. Ripetere l'esercizio sostituendo prima in Tinkercad e poi sulla Breadboard, l'interruttore con un Pulsante.

SCHEMA 1: LOGICA NEGATIVA

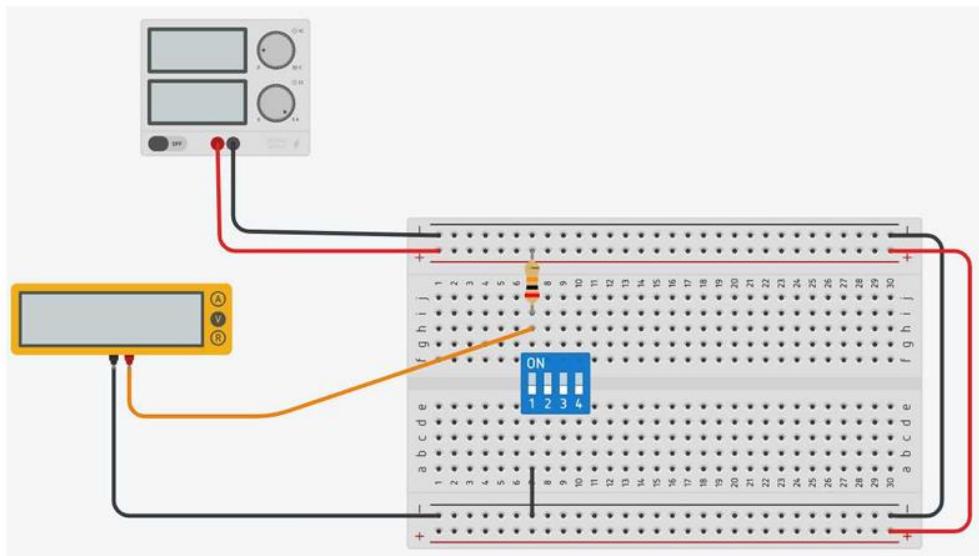
INTERRUTTORE APERTO

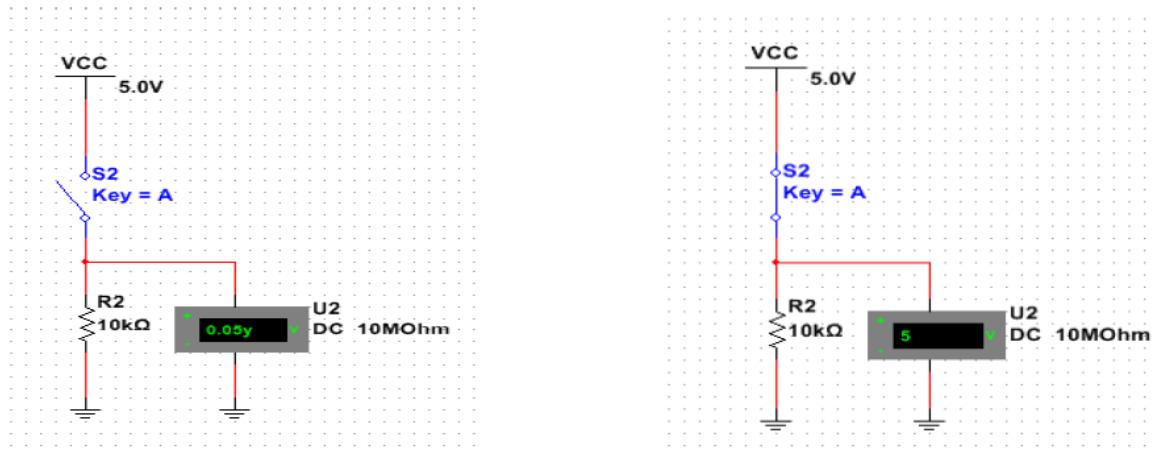
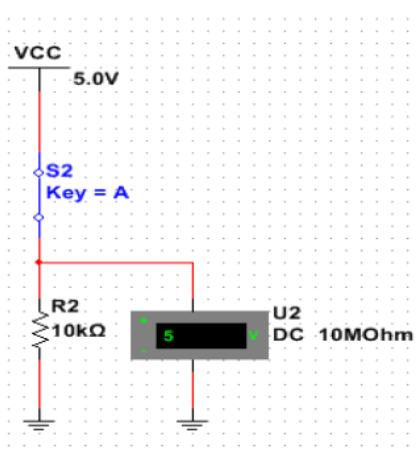
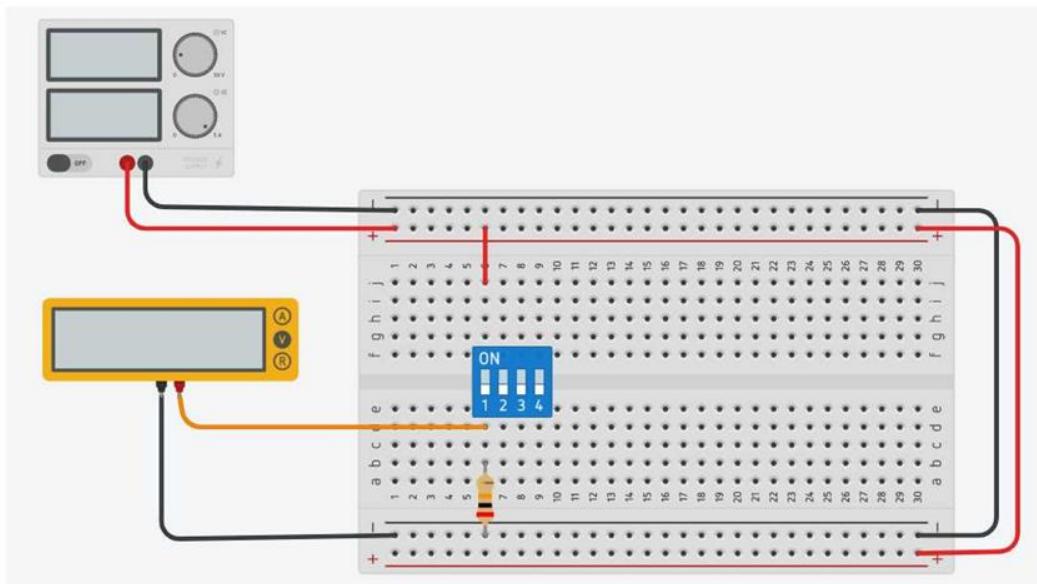


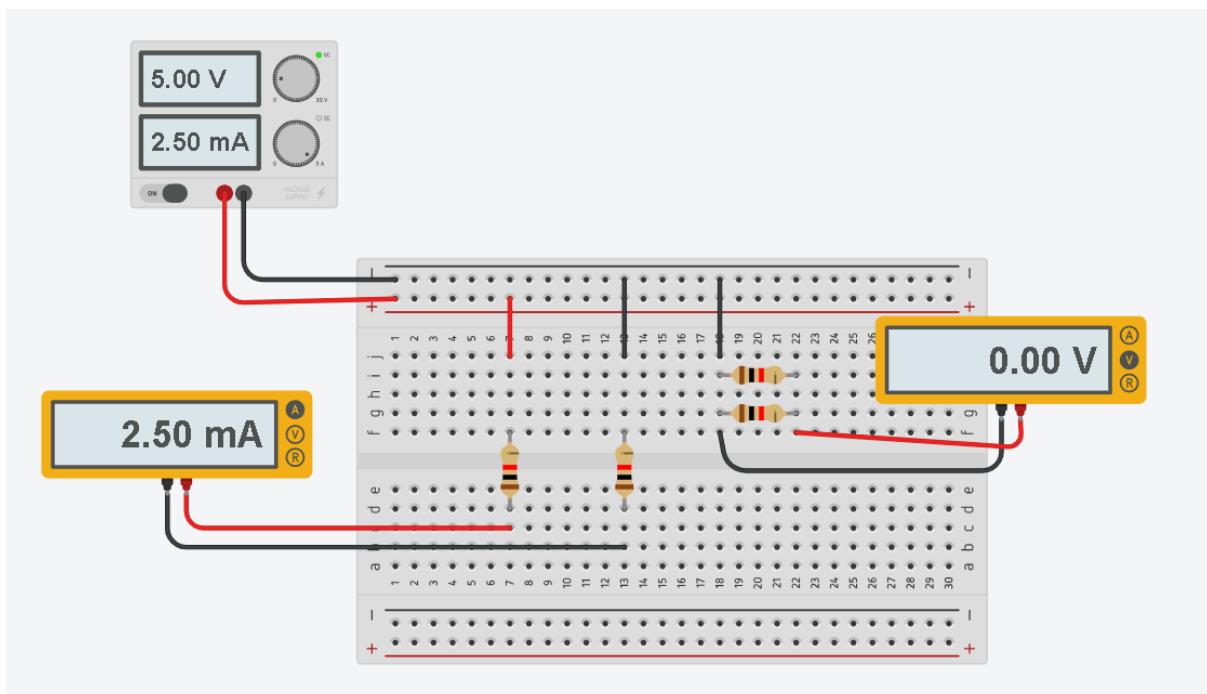
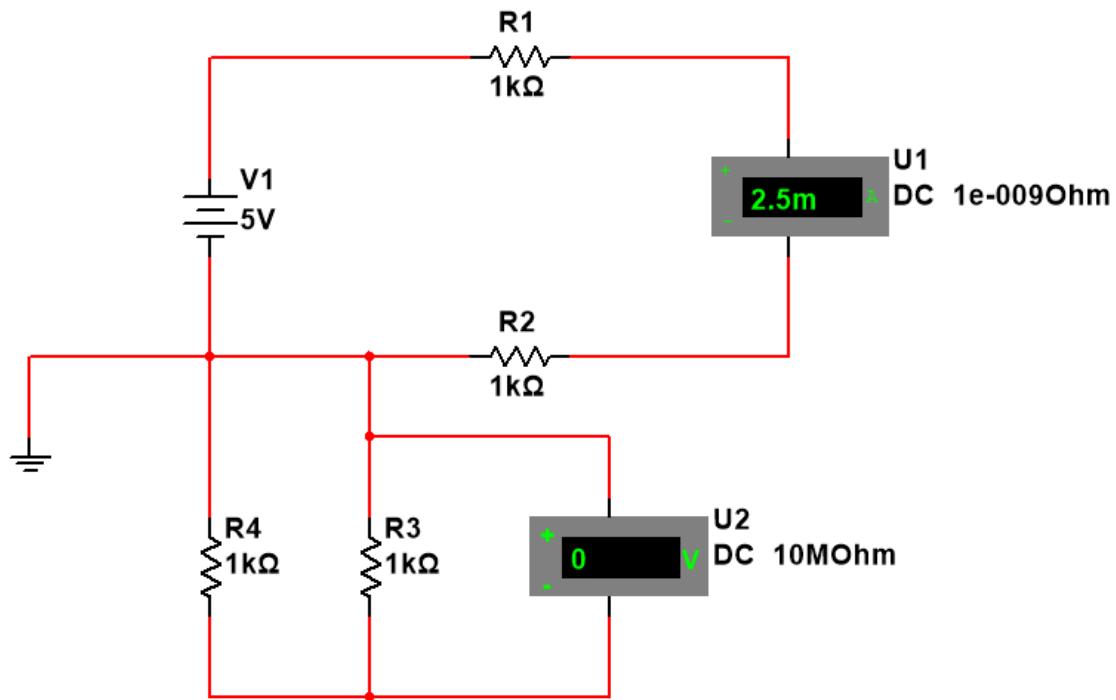
INTERRUTTORE CHIUSO

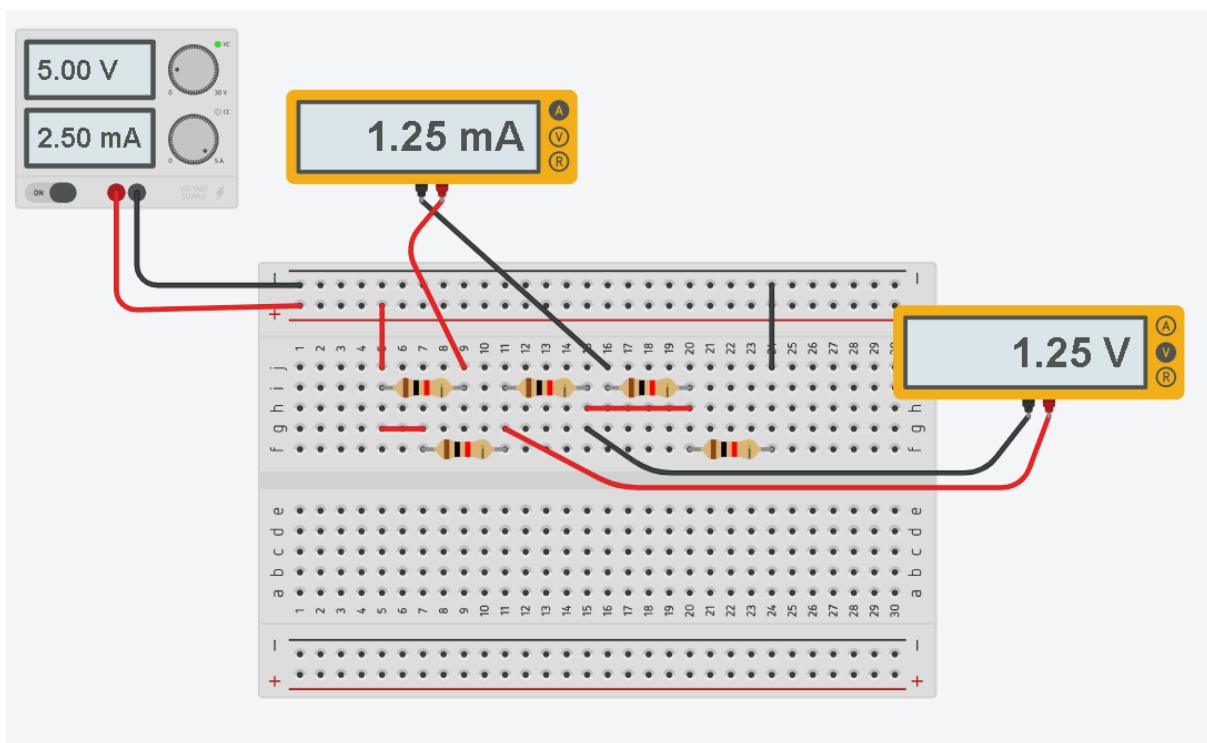
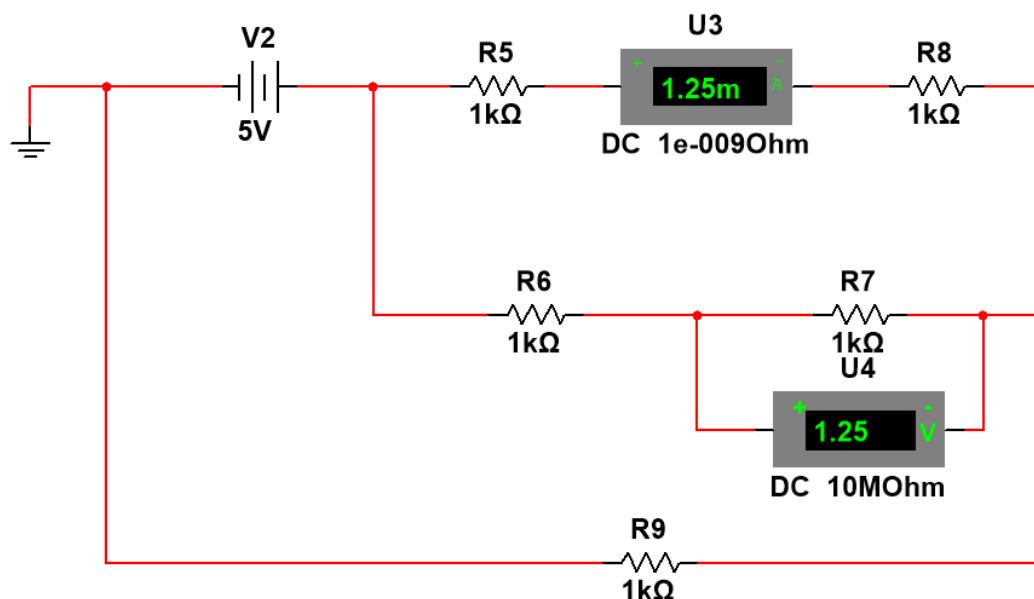


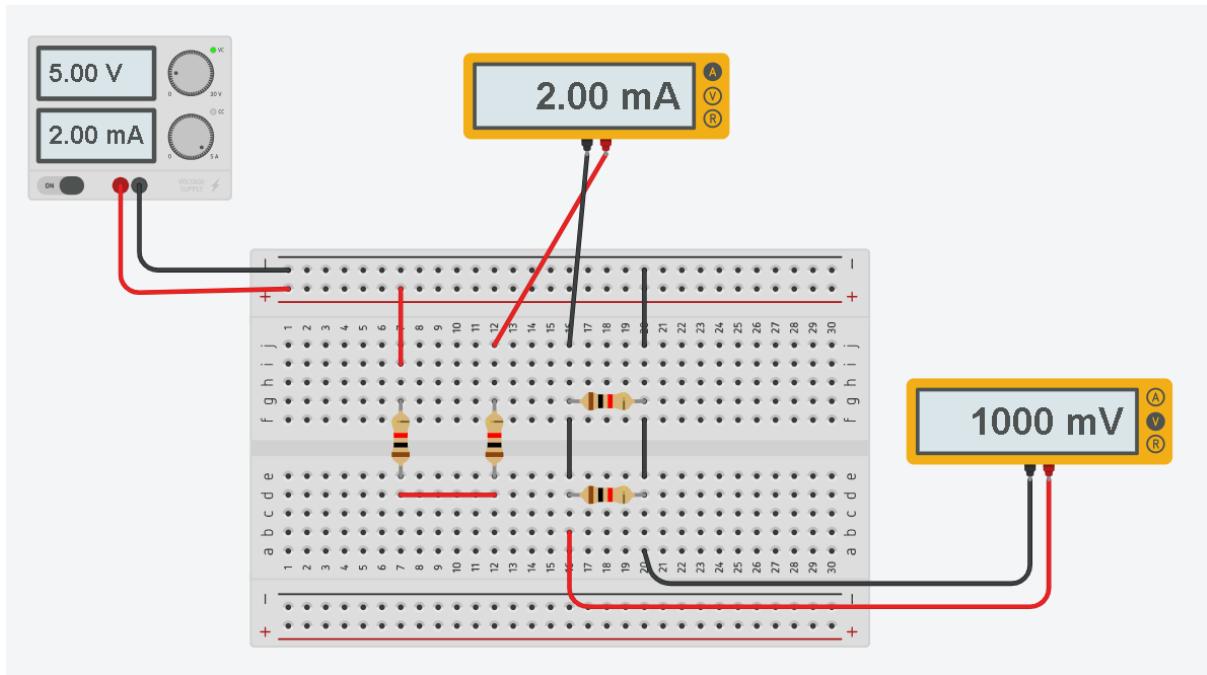
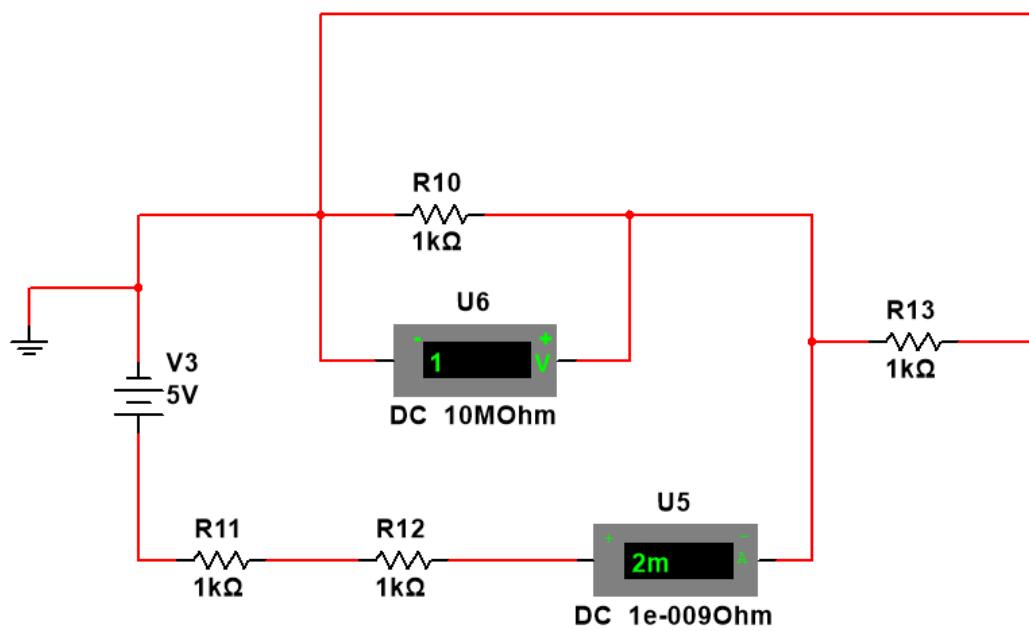
SEGNALE IN INPUT IN LOGICA NEGATA TRAMITE INTERRUTTORE

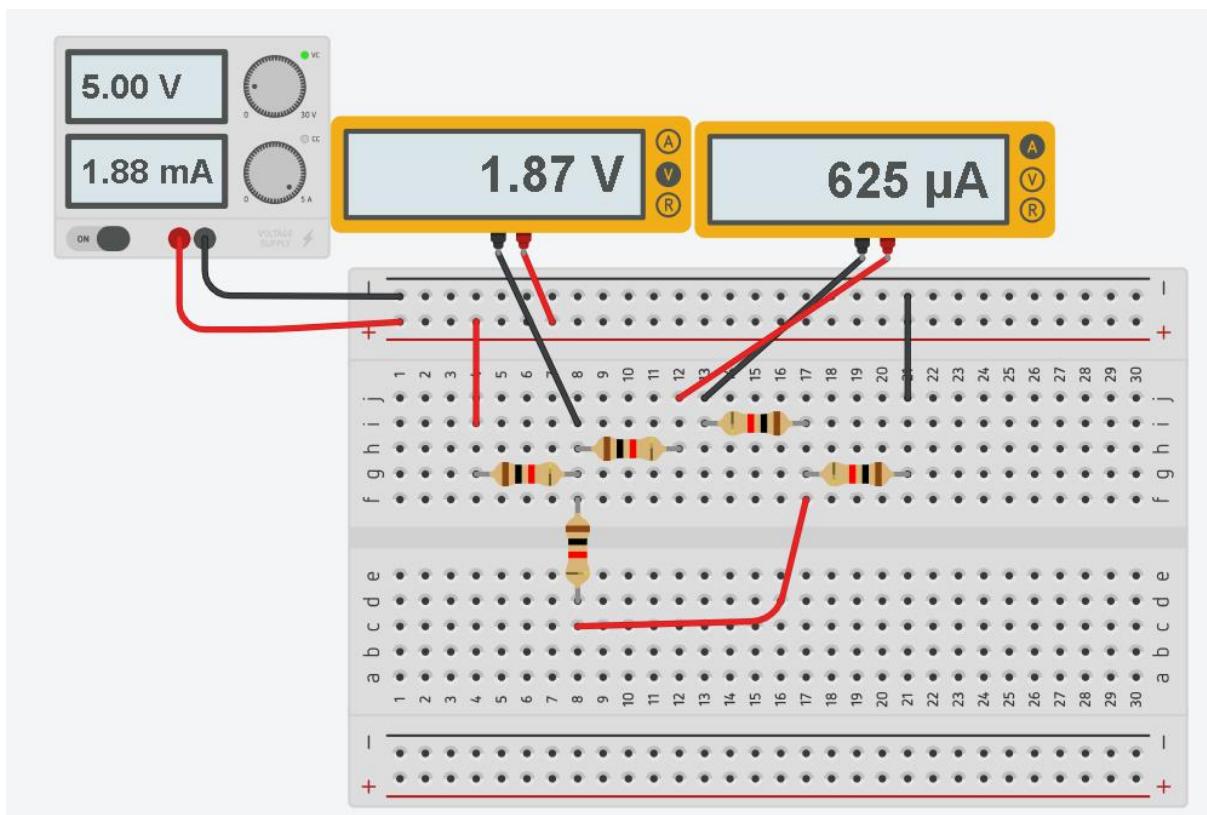
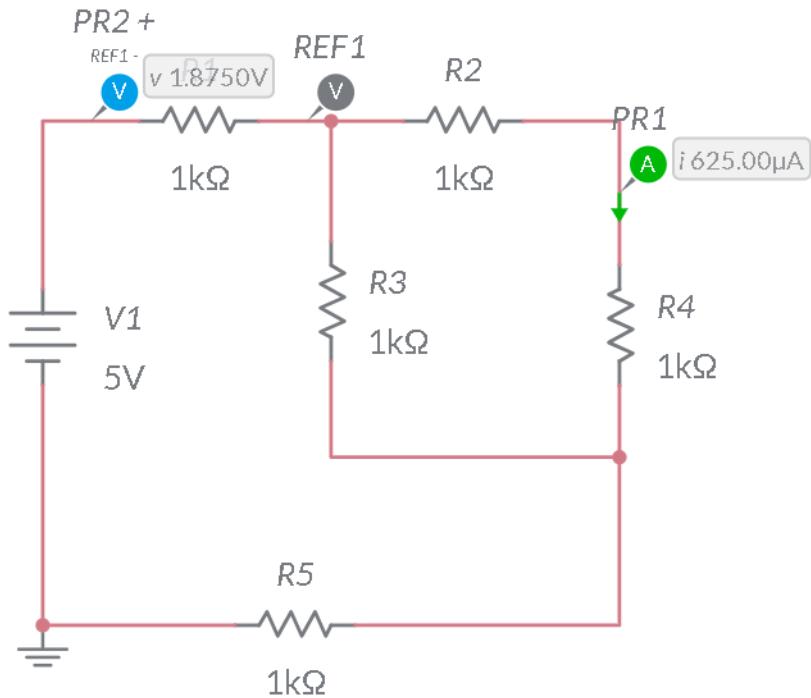


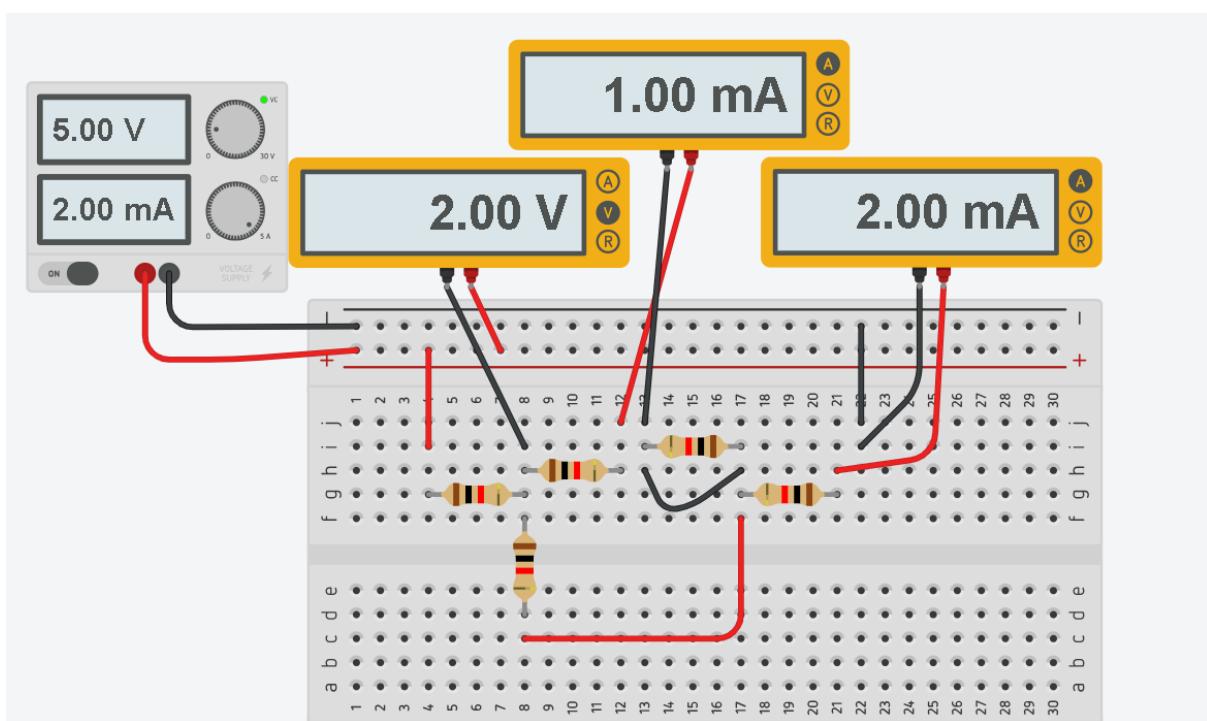
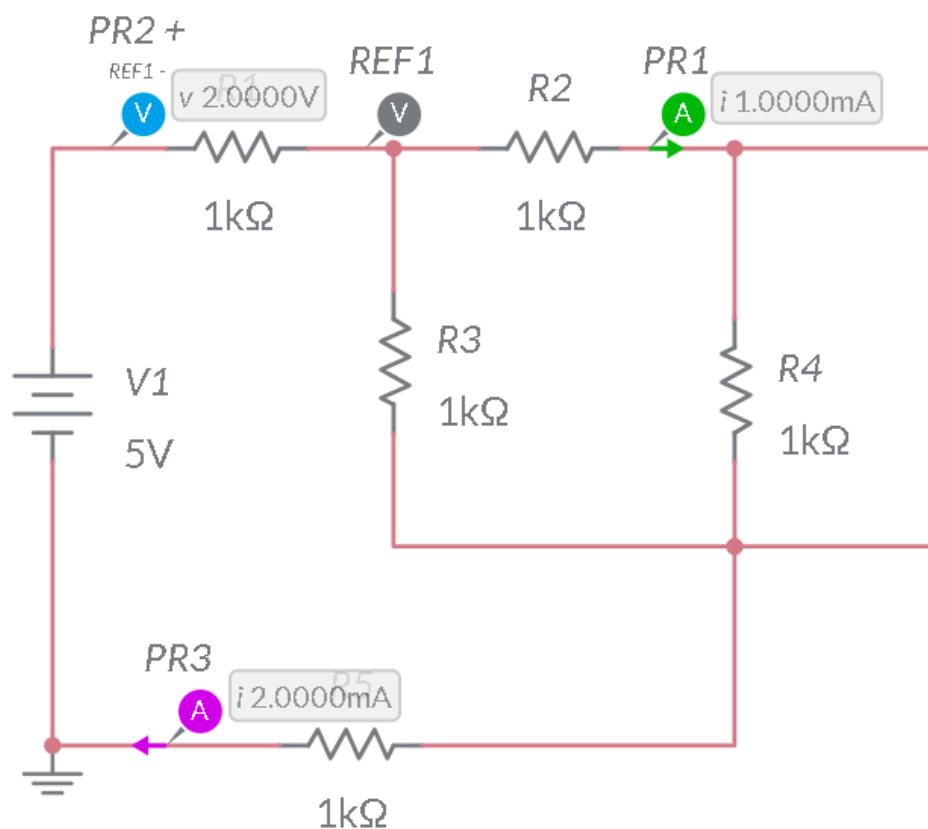
SCHEMA 2: LOGICA POSITIVA**INTERRUTTORE APERTO****INTERRUTTORE CHIUSO****SEGNALI DIGITALI:****SEGNALI IN INPUT IN LOGICA POSITIVA TRAMITE INTERRUTTORE**

ESERCIZI TIPO COMPITO:







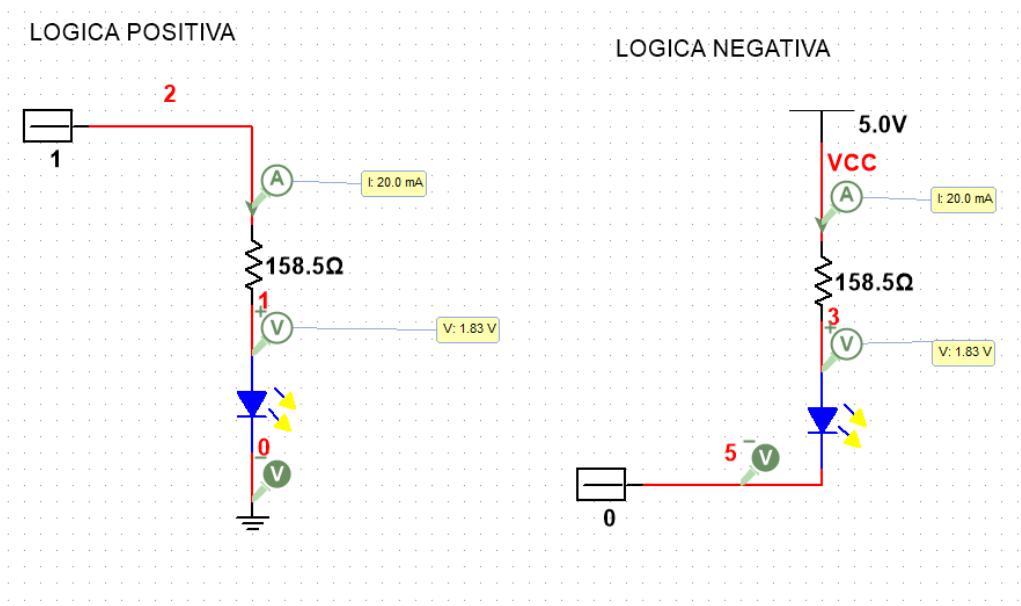


ESERCITAZIONE N°7: SEGNALE DIGITALE IN USCITA

Compiere una ricerca riguardante il Diodo Led ricercando:

- il simbolo grafico
- significato di catodo e anodo
- la curva caratteristica corrente-tensione

Realizzare i seguenti schemi, il primo dei quali in logica positiva e il secondo in logica negativa.



ESERCITAZIONE N°8: PORTE LOGICHE APPLICAZIONE SEGNALI DIGITALI IN INGRESSO E USCITA.

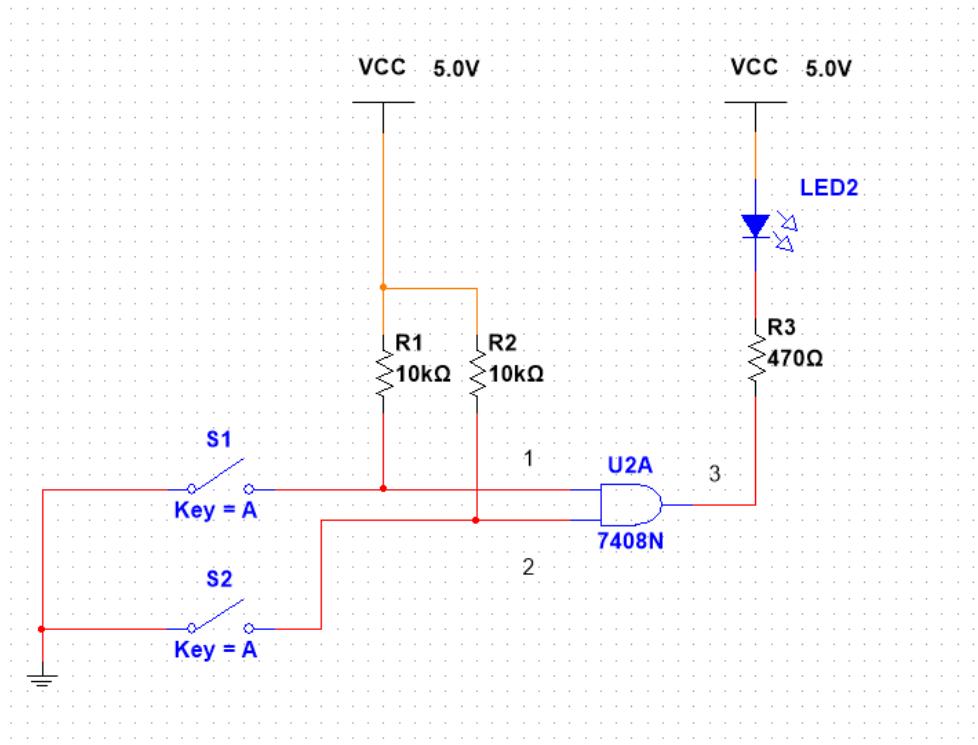
- Verificare la tabella di verità per una porta logica a scelta tra le seguenti, adottando sia per i segnali digitali di ingresso che di uscita la logica negata.

NAND 7400

AND 7408

OR 7432

NOR 7402



ESERCITAZIONE N°9: CASO DI UNA MACCHINA OPERATRICE

LEZIONE 17 Reti logiche: controllo di una macchina operatrice

UNITÀ 4 LE RETI LOGICHE

Fig. 1

Una macchina operatrice per il taglio dei metalli (fresa) è provvista di due pulsanti, ognuno dei quali, se premuto, aziona il motore della macchina utensile. Il primo pulsante (A) è posto ai piedi dell'operatore e il secondo (B) è sul lato della macchina (fig. 1).

La macchina è provvista inoltre di due **microinterruttori** usati come dispositivi di emergenza che arrestano il motore non appena uno di essi viene aperto.

Il primo microinterruttore (M_1), posto sul pannello frontale di protezione dell'apparato di taglio, si apre non appena viene aperto il pannello, arrestando così il motore della macchina se in funzione.

Il secondo microinterruttore (M_2) è posizionato sullo sportello che permette l'accesso al sistema di alimentazione della fresa: aprendolo si ha l'arresto immediato del motore se in movimento.

Per la realizzazione della rete combinatoria necessaria per il funzionamento eseguiamo ordinatamente i passi visti nella lezione precedente.

Individuazione delle variabili presenti nel sistema

Le variabili d'ingresso sono i pulsanti A e B e i microinterruttori M_1 ed M_2 ; il livello logico a loro associato è riportato nella tabella di figura 2.

Stesura della tabella della verità

La tabella della verità, costituita da 16 combinazioni facenti capo alle quattro variabili binarie indipendenti ($2^4 = 16$), è quella di figura 3: allo stato del motore della macchina indicato con Y_M (variabile d'uscita) è associato un 1 logico o uno 0 logico a seconda che il motore sia in azione o fermo.

VARIABILE BINARIA	STATO DEL SEGNALE	LIVELLO LOGICO ASSOCIATO
A	Pulsante non premuto	0
	Pulsante premuto	1
B	Pulsante non premuto	0
	Pulsante premuto	1
M_1	Microswitch 1 chiuso	1
	Microswitch 1 aperto	0
M_2	Microswitch 2 chiuso	1
	Microswitch 2 aperto	0

Fig. 2
Associazione degli stati alle variabili.

VARIABILI IN INGRESSO				VARIABILE IN USCITA	NOTE
A	B	M_1	M_2	Y_M	
0	0	0	0	0	Motore fermo
0	0	0	1	0	Motore fermo
0	0	1	0	0	Motore fermo
0	0	1	1	0	Motore fermo
0	1	0	0	0	Motore fermo
0	1	0	1	0	Motore fermo
0	1	1	0	0	Motore fermo
0	1	1	1	1	Motore in movimento
1	0	0	0	0	Motore fermo
1	0	0	1	0	Motore fermo
1	0	1	0	0	Motore fermo
1	0	1	1	1	Motore in movimento
1	1	0	0	0	Motore fermo
1	1	0	1	0	Motore fermo
1	1	1	0	0	Motore fermo
1	1	1	1	1	Motore in movimento

Fig. 3
Tabella della verità.

Traduzione della tabella in funzione booleana

La funzione logica, considerando i mintermini che valgono 1, è:

$$Y_M = m_7 + m_{11} + m_{15} = \bar{A}\bar{B}M_1M_2 + A\bar{B}M_1M_2 + ABM_1M_2$$

Semplificazione della funzione

Applicando le regole e i teoremi dell'algebra di Boole troviamo:

$$\begin{aligned} Y_M &= \bar{A}BM_1M_2 + A\bar{B}M_1M_2 + ABM_1M_2 = M_1M_2(\bar{A}B + A\bar{B} + AB) = \\ &= M_1M_2[\bar{A}B + A(\bar{B} + B)] = M_1M_2(\bar{A}B + A) = M_1M_2(A + B) \end{aligned}$$

Realizzazione del circuito logico

Lo schema del circuito che meccanizza la funzione trovata è quello di figura 4, in cui sono necessarie due porte logiche AND e una porta logica OR, e quindi due circuiti integrati diversi (7408 e 7432 rispettivamente).

Possiamo verificare che, applicando i teoremi di De Morgan, lo stesso circuito può essere realizzato con tre porte logiche NAND e quindi con un solo circuito integrato (7410, triplo NAND a tre ingressi) anziché con due (7432-7408), risparmiando quindi sul costo.

In figura 5 è riportato lo schema di cablaggio del circuito di figura 4 montato su di una basetta sperimentale (descritta nel modulo precedente). Collegando gli ingressi, ad esempio, come mostrato dal tratteggio, ($A = 0 \text{ V}$; $B = M_1 = M_2 = +5 \text{ V}$), il motore sarà in movimento perché viene soddisfatta l'ottava riga della tabella della verità in cui $Y_M = 1$.

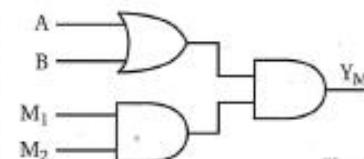
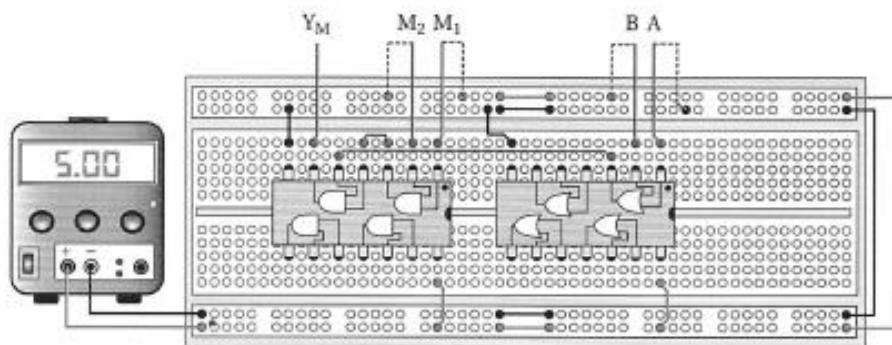
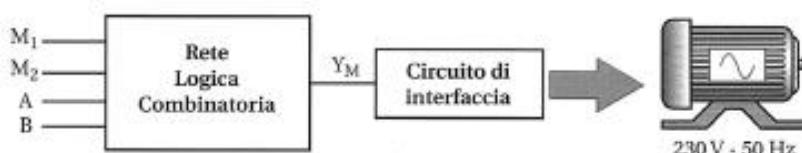


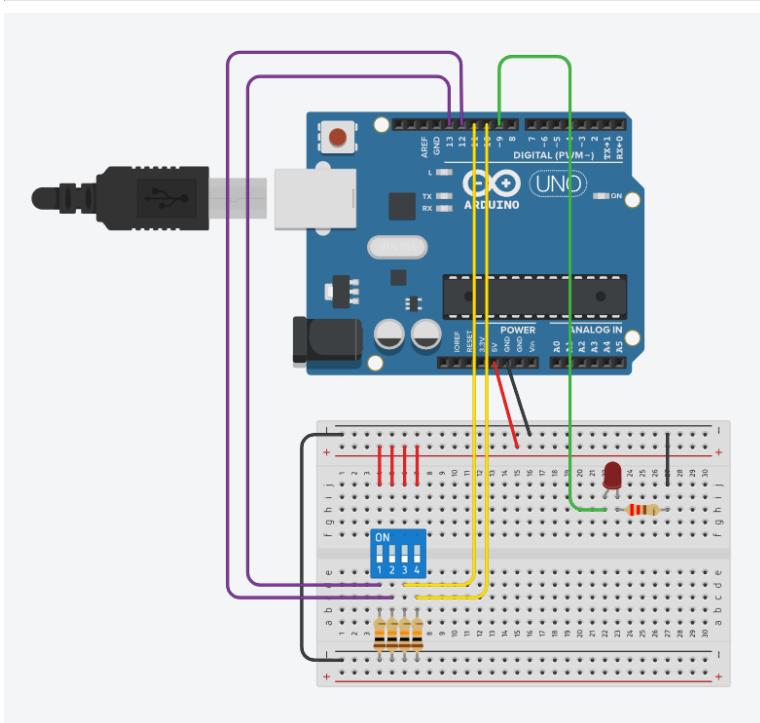
Fig. 4

Fig. 5
Schema di cablaggio.

Naturalmente quello ricavato è solo il circuito logico di controllo dell'azionamento o meno del motore.

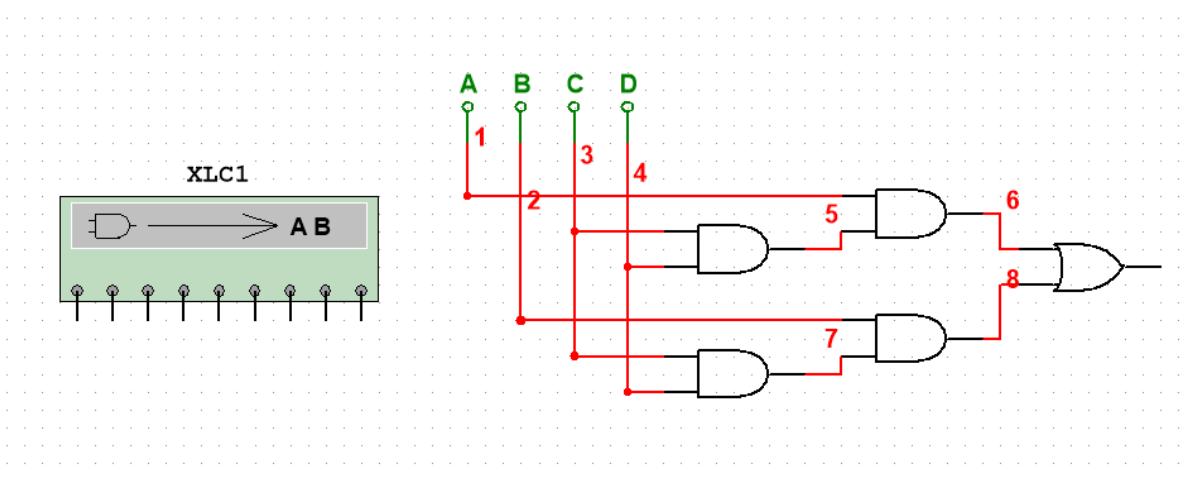
Per poter effettivamente arrestare o azionare il motore della fresa è necessario un circuito di interfaccia tra la parte logica e quella di potenza al cui ingresso viene collegata l'uscita Y_M della rete combinatoria (fig. 6). Ciò permette di azionare il motore, collegandolo all'alimentazione di rete ($230 \text{ V}-50 \text{ Hz}$) se $Y_M = 1$, o di disconnetterlo dall'alimentazione se $Y_M = 0$.

Fig. 6
Schema a blocchi in grado di azionare il motore della fresa secondo la tabella della verità di figura 3.



```
// C++ code
bool A,B,C,D,Y;
int i;
void setup() {
  for(i=10;i<=13;i=i+1){
    pinMode(i,INPUT);
  }
  pinMode(9,OUTPUT);
}

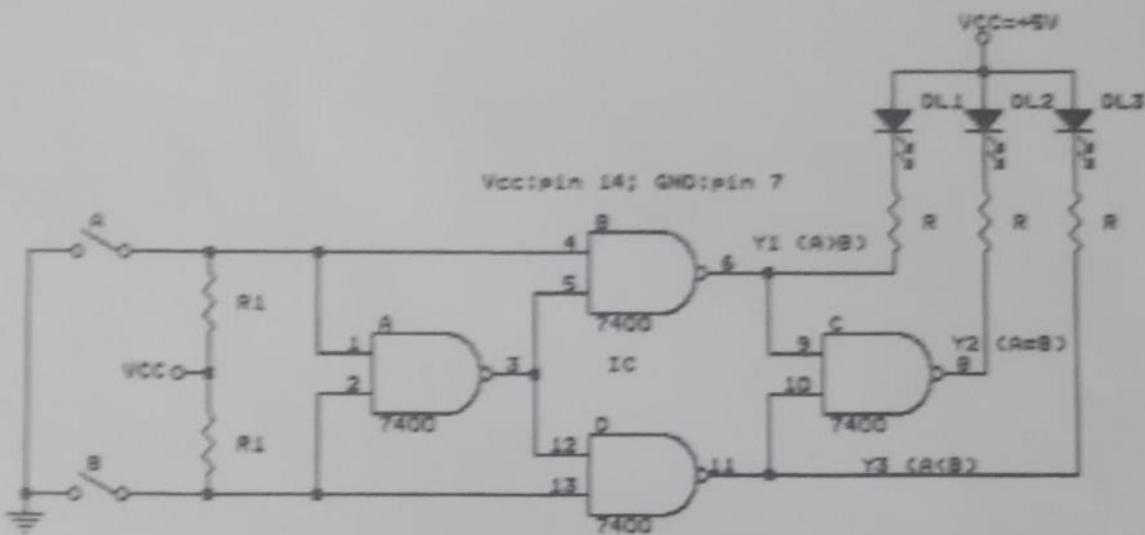
void loop(){
  A=digitalRead(13);
  B=digitalRead(12);
  C=digitalRead(11);
  D=digitalRead(10);
  Y=(A&&C&&D)+(B&&C&&D);
  digitalWrite(9,Y);
}
```



ESERCITAZIONE N°10: COMPARATORE DIGITALE AD UN BIT

Comparatore digitale ad un bit (Teoria: cap. II, par. 1).

Schema elettrico:



Componenti: $R = 470 \Omega$; $R_1 = 10 \text{ k}\Omega$; DL1-2-3 = diodi LED; IC = 7400

Prove da effettuare: applicando in successione le quattro possibili combinazioni tra gli ingressi A e B verificare, tramite l'accensione dei LED, che il circuito si comporta come comparatore digitale ad un bit con uscite in logica negativa. Poiché i LED sono anch'essi montanti in logica negativa, il verificarsi di una data condizione è segnalato dall'accensione del relativo LED.

ESERCITAZIONE N°11: MULTIPLEXER A 4 INGRESSI

ESERCITAZIONE N. 17

Multiplexer a quattro ingressi (Teoria; cap. IV, par. 4).

Schema elettrico e tabella della verità:

A ₁	A ₀	OE	Y
X	X	1	0
0	0	0	D ₀
0	1	0	D ₁
1	0	0	D ₂
1	1	0	D ₃

Elenco componenti: $C = 100 \text{ nF}$; $R_1 = 270 \Omega$; $R_2 = 330 \Omega$; $R_3 = 390 \Omega$; $R_4 = 470 \Omega$; $R = 10 \text{ k}\Omega$; sestuplo inverter a TS 7414; doppio multiplexer 74153.

Apparecchi da utilizzare: oscilloscopio.

Note: le entrate del multiplexer sono comandate da 4 generatori di onde quadre a trigger di Schmitt con periodo $T_n = 1.15 \cdot R_n \cdot C$ con $n = 1 \dots 4$.

Conduzione della prova: determinare il periodo dei quattro multivibratori astabili utilizzando l'oscilloscopio.

Collegare, poi, l'oscilloscopio all'uscita Y del multiplexer e tenere aperto l'interruttore OE (OE = 1). Si osserva un oscillogramma piatto corrispondente al livello logico 0 indipendentemente dallo stato degli interruttori A e B.

Successivamente si chiude l'interruttore OE (OE = 0) e verificare la tabella della verità confrontando il periodo dell'oscillogramma con quello dell'oscillatore collegato al multiplexer corrispondente all'indirizzo $A_1 A_0$ impostato.

ESERCITAZIONE N°12: FULL-ADDER (SOMMATORE A TRE BIT)

Esercitazioni di laboratorio

547

ESERCITAZIONE N. 18 (SOMMATORE A TRE BIT)

Full-adder con decodificatore (Teoria: cap. IV, par. 8).

Schema elettrico e tabella della verità:

The circuit diagram shows a 74138 decoder (IC1) with its enable pin (pin 14) connected to ground. Its outputs Y1 through Y7 are connected to the inputs of a 7420 four-input NAND gate (IC2A). The 7420's output is connected to one input of a second 7420 NAND gate (IC2B). The other input of IC2B is connected to the enable pin of IC1. The output of IC2B is labeled 'SOMMA'. A third 7420 NAND gate (IC2D) is used as a buffer, with its output connected to a 'RIPORTO' (carry-out) terminal. The circuit also includes four microswitches (labeled G1, G2A, G2B) connected to the enable pin of IC1 and two 370 ohm resistors connected to the 'SOMMA' and 'RIPORTO' outputs.

Elenco componenti: microswitch a 4 interruttori; decodificatore 74138; porte NAND 7420; 2 diodi LED.

Note: si osservi che il pilotaggio dei diodi LED proposto è senza resistenze di limitazione. La corrente è limitata intorno a 15 mA dalle stesse porte logiche.

Prove da effettuare: verificare che se l'interruttore di abilitazione G_2 è aperto ($G_2 = 1$) i due LED rimangono spenti anche se si modifica lo stato logico degli ingressi A , B e C .

Portare, poi, $G_2 = 0$ e applicare le 8 combinazioni possibili tra gli addendi ad un bit A e B e il riporto precedente C e rilevare gli stati dei LED SOMMA e RIPORTO verificando, così, la tabella della verità che è quella di un sommatore completo.

NOTE

① ATTENZIONE MODIFICARE
CIRCUITO DI INGRESSO CON DIP SWITCH

② ATTENZIONE MODIFICARE
CIRCUITO CON LED IN LOGICA NEGATA

TRE BIT SOMMA 2BIT			
A	B	S	R
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

SOMMA 3BIT LED LS UR				
A	B	C	S	R
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	0

Tabella della verità (Truth Table):

A	B	C	S	R
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	0

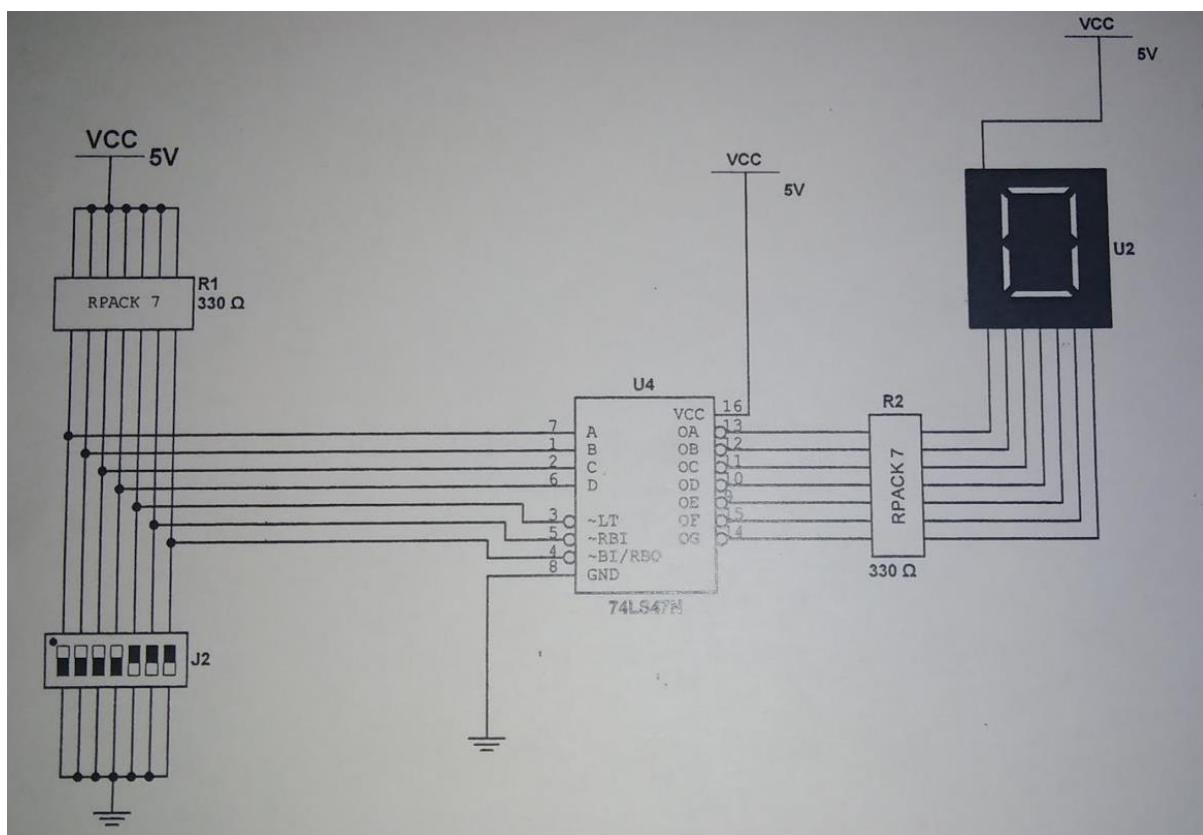
Circuit diagram of a 3-bit full adder using a 74138 decoder and a 7420 four-input NAND gate.

The circuit diagram shows a 74138 decoder (74LS138) with its enable pin (G1) connected to ground. The decoder's outputs Y1 through Y7 are connected to the inputs of a 7420 four-input NAND gate (NAND 4 INGRESSI 7420). The 7420's output is connected to one input of a second 7420 NAND gate. The other input of the second 7420 is connected to the enable pin of the 74138. The output of the second 7420 is labeled 'SOMMA'. A third 7420 NAND gate is used as a buffer, with its output connected to a 'RIPORTO' (carry-out) terminal. The circuit also includes four microswitches (G1, G2A, G2B) connected to the enable pin of the 74138 and two 370 ohm resistors connected to the 'SOMMA' and 'RIPORTO' outputs.

Segnali di ABILITAZIONE:

$$\begin{cases} G_1 = \text{HIGH} \\ G_2^* = \overline{G_2A} + \overline{G_2B} = \text{LOW} \end{cases} \Rightarrow \begin{cases} \overline{G_2A} = \text{LOW} \\ \overline{G_2B} = \text{LOW} \end{cases}$$

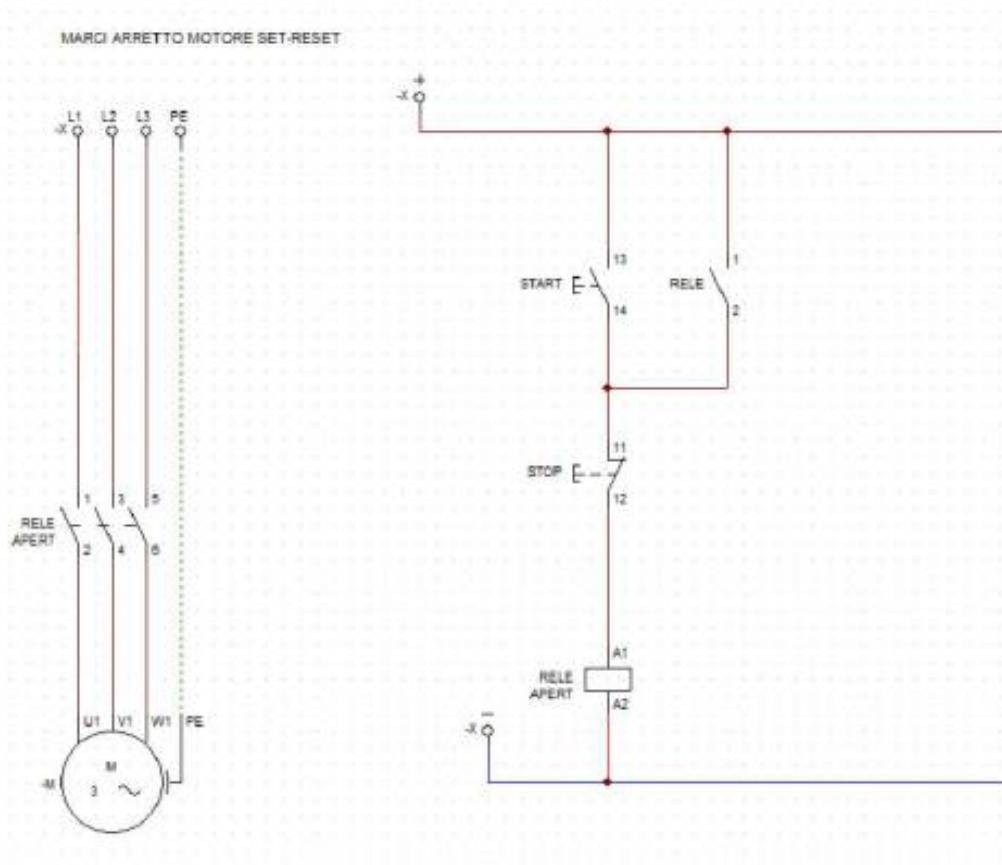
ESERCITAZIONE N°13: FUNZIONAMENTO DEL DECODER DRIVER



ESERCITAZIONE N°14: LATCH SR

Simulare l'avviamento e lo spegnimento di un motore elettrico tramite l'equivalente circuito elettronico LATCH SR.

La presenza del motore è simulata da un led e il circuito elettromeccanico sostituito dalla configurazione a porte NAND che realizzano il LATCH SR.

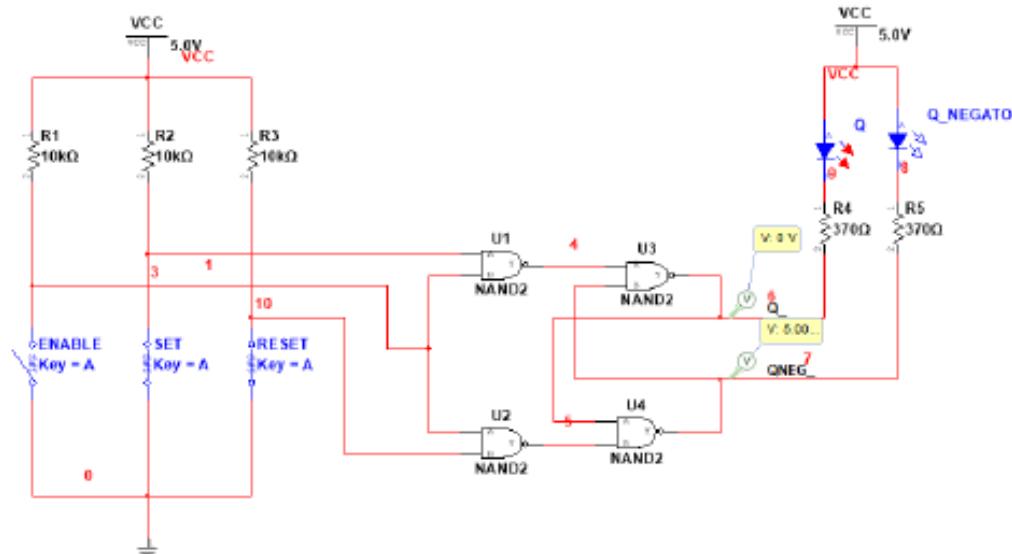


Provare a eseguire la sequenza di step temporali sotto riportata, agendo sugli interruttori.

LATCH TIPO SET-RESET CON ENABLE (LOGICA SEQUENZIALE)

RICAVARE LA TABELLA DELLA VERITÀ: AGIRE SUGLI INTERRUATORI SECONDO LA SEQUENZA E TRASCRIVERE IL VALORE DI Q

- | | | | | |
|-----|----------------------|---------------------|-----------------------|----------|
| t=0 | ENABLE HIGH (APERTO) | SET = LOW (CHIUSO) | RESET = LOW (CHIUSO) | Q=?????? |
| t=1 | ENABLE HIGH (APERTO) | SET = HIGH (APERTO) | RESET = LOW (CHIUSO) | Q=?????? |
| t=2 | ENABLE HIGH (APERTO) | SET = LOW (CHIUSO) | RESET = LOW (CHIUSO) | Q=?????? |
| t=3 | ENABLE HIGH (APERTO) | SET = LOW (CHIUSO) | RESET = HIGH (APERTO) | Q=?????? |
| t=4 | ENABLE HIGH (APERTO) | SET = LOW (CHIUSO) | RESET = LOW (CHIUSO) | Q=?????? |



LATCH SR CON ENABLE

S	SET
R	RESET
EN	ENABLE
Q	USCITA

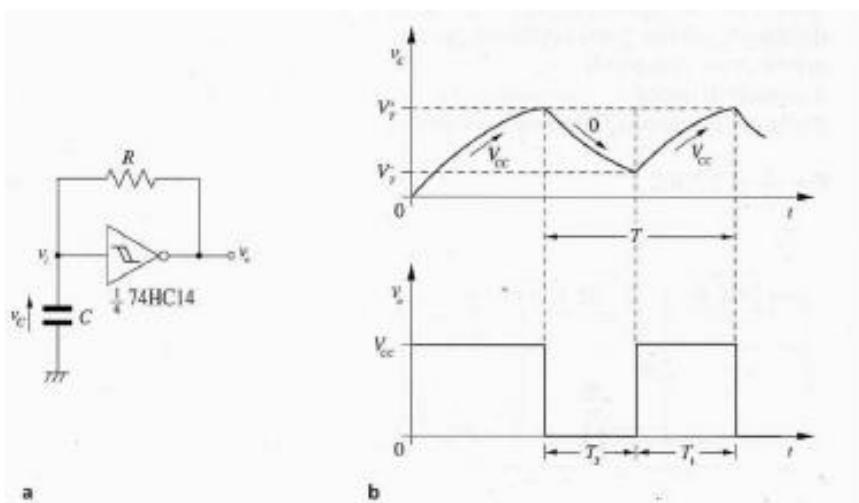
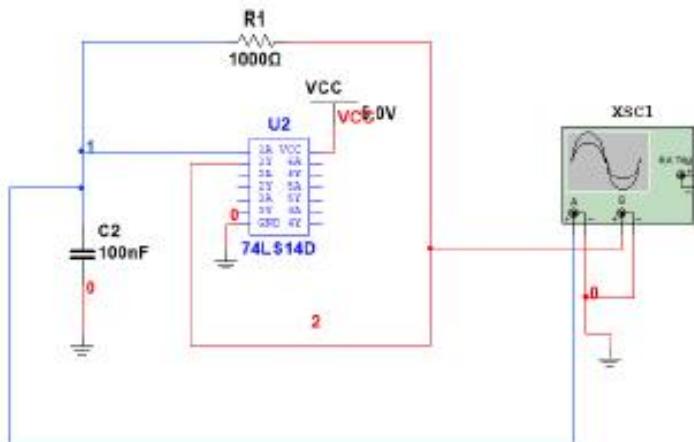
CIRCUITO COSTITUITO DA DUE SEGNAli DI INGRESSO SET E RESET
LA VERSIONE DI LATCH IN ESAME PRESENTA L'ENABLE

	Abilitazione	Set	Reset	Q ₀₊₁	
t=0	1	1	0	1	SET
t=1	1	0	0	Q ₁	MEMORIA
t=2	1	0	1	0	RESET
t=3	1	0	0	Q ₁	MEMORIA
t=4	0	X	X	Q ₁	MEMORIA
	1	1	1	NON DETERMINATO	la combinazione non è funzionale

ESERCITAZIONE N°15: TRIGGER DI SCHMITT

Foglio1

Realizzare lo schema riportato in figura, confrontare i valori ottenuti visualizzati da oscilloscopio con quelli calcolati applicando le formule spiegate in classe



$$T_1 = RC \ln \frac{V_{CC} - V_T^-}{V_{CC} - V_T^+} \quad T_2 = RC \ln \frac{0 - V_T^+}{0 - V_T^-} = RC \ln \frac{V_T^+}{V_T^-}$$

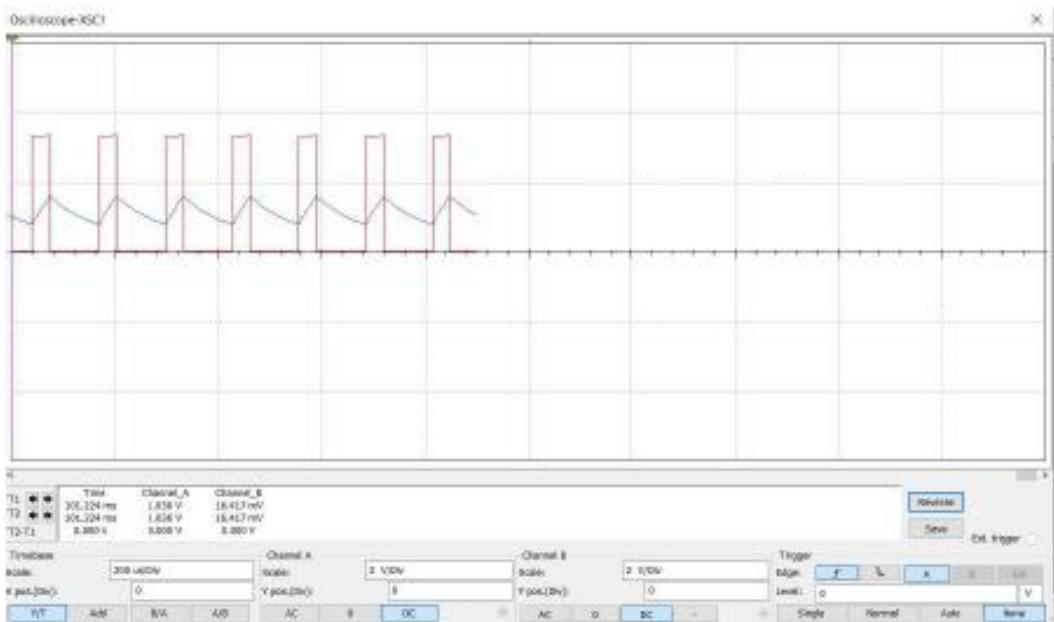
Il periodo complessivo T e la frequenza $f = 1/T$ si ricavano dall'espressione:

$$T = \frac{1}{f} = T_1 + T_2 = RC \ln \left(\frac{V_{CC} - V_T^-}{V_{CC} - V_T^+} \times \frac{V_T^+}{V_T^-} \right) \quad [11.6]$$

Foglio1

Vcc	5	V			
R	1	KΩ	1000	Ω	
C	100	nF	1E-07	F	
VT+	1,6	V			
VT-	0,8	V			

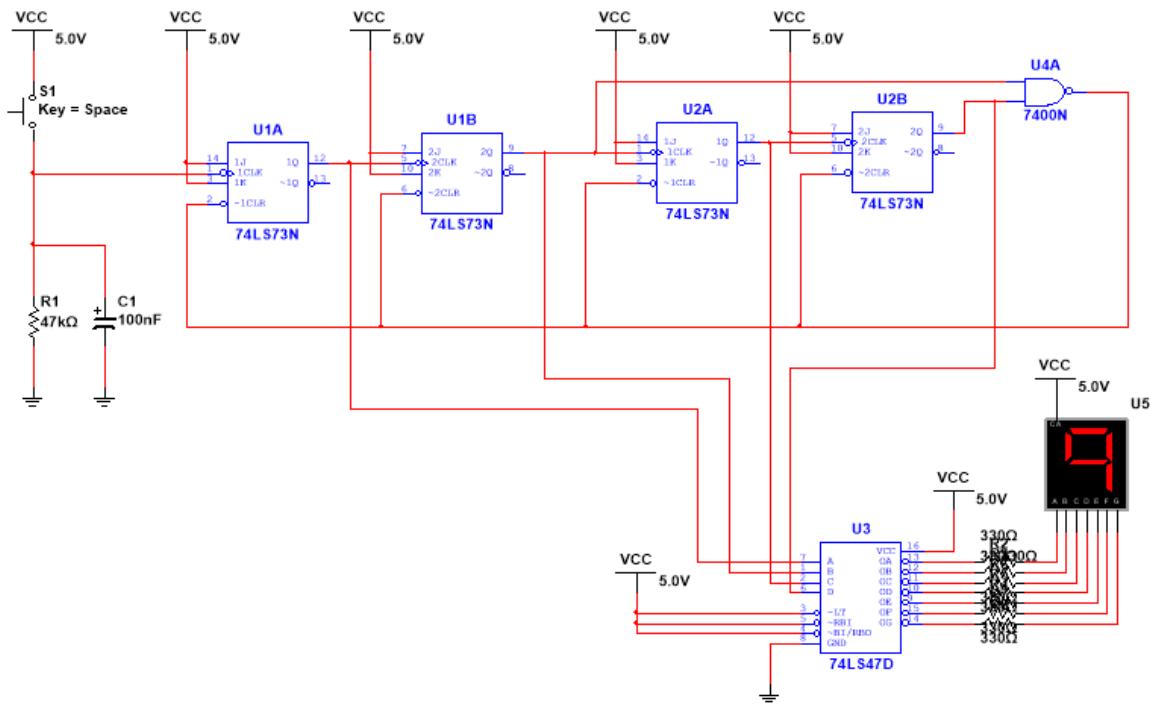
T1	0,0000	s	21	μs	(segnale a livello logico HIGH)
T2	0,0001	s	69	μs	(segnale a livello logico LOW)
T1+T2	0,0001	s	90	μs	periodo del segnale
f	11056	Hz			frequenza

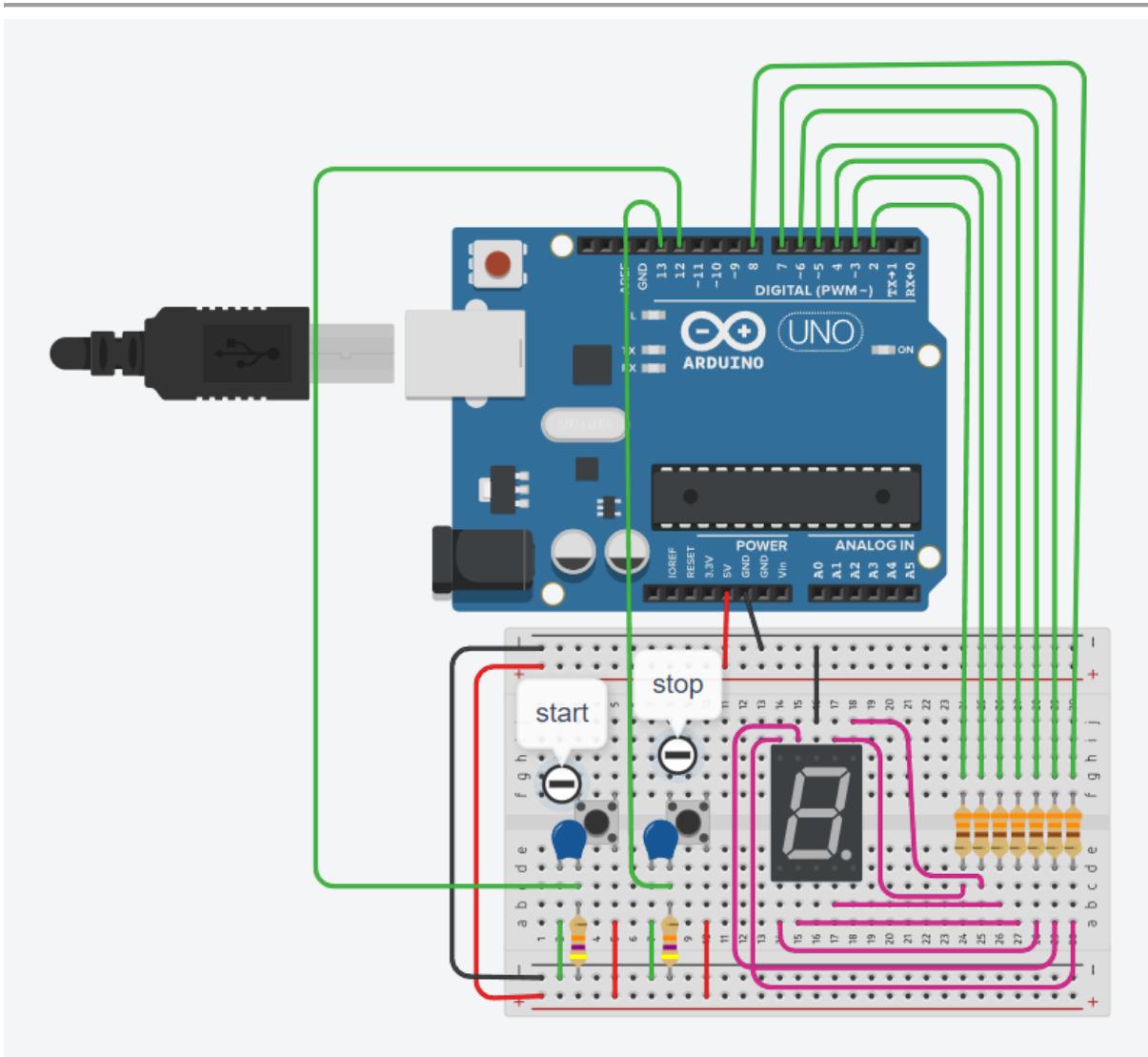


valori rilevati da Multisim

Voutput	3,3	V			
T1	32	μs	(segnale a livello logico HIGH)		
T2	93	μs	(segnale a livello logico LOW)		
T1+T2	125	μs	periodo del segnale	0,000125	
frequenza	8000	Hz	frequenza		

CONTATORE MODULO 10





// C++ code

//

```
bool statoPulsstart;  
bool statoPulsprecstart=0;  
bool statoPulsstop;  
bool statoPulsprecstop=0;  
int contatore=0;
```

```
void setup(){
    int i;
    for (i=2;i<=9;i++){
        pinMode(i,OUTPUT);
        pinMode(12,INPUT);
        pinMode(13,INPUT);
        Serial.begin(9600);
    }

void loop(){
    statoPulsstart=digitalRead(12);
    statoPulsstop=digitalRead(13);
    if (contatore <9 && statoPulsstart==1 && statoPulsprecstart==0){
        contatore++;
    } else if(contatore==9 && statoPulsstart==1 && statoPulsprecstart==0){
        contatore=0;

    if (contatore>0 && statoPulsstop==1 && statoPulsprecstop==0){
        contatore--;
    } else if(contatore==0 && statoPulsstop==1 && statoPulsprecstop==0){
        contatore=0;

switch(contatore){

case 0:
    digitalWrite(2,HIGH);
    digitalWrite(3,HIGH);
```

```
digitalWrite(4,HIGH);
digitalWrite(5,HIGH);
digitalWrite(6,HIGH);
digitalWrite(7,HIGH);
digitalWrite(8,LOW);

break;

case 1:
    digitalWrite(2,LOW);
digitalWrite(3,HIGH);
digitalWrite(4,HIGH);
digitalWrite(5,LOW);
digitalWrite(6,LOW);
digitalWrite(7,LOW);
digitalWrite(8,LOW);

break;

case 2:
    digitalWrite(2,HIGH);
digitalWrite(3,HIGH);
digitalWrite(4,LOW);
digitalWrite(5,HIGH);
digitalWrite(6,HIGH);
digitalWrite(7,LOW);
digitalWrite(8,HIGH);

break;

case 3:
    digitalWrite(2,HIGH);
digitalWrite(3,HIGH);
digitalWrite(4,HIGH);
digitalWrite(5,HIGH);
```

```
digitalWrite(6,LOW);
digitalWrite(7,LOW);
digitalWrite(8,HIGH);

break;

case 4:
    digitalWrite(2,LOW);
    digitalWrite(3,HIGH);
    digitalWrite(4,HIGH);
    digitalWrite(5,LOW);
    digitalWrite(6,LOW);
    digitalWrite(7,HIGH);
    digitalWrite(8,HIGH);

break;

case 5:
    digitalWrite(2,HIGH);
    digitalWrite(3,LOW);
    digitalWrite(4,HIGH);
    digitalWrite(5,HIGH);
    digitalWrite(6,LOW);
    digitalWrite(7,HIGH);
    digitalWrite(8,HIGH);

break;

case 6:
    digitalWrite(2,HIGH);
    digitalWrite(3,LOW);
    digitalWrite(4,HIGH);
    digitalWrite(5,HIGH);
    digitalWrite(6,HIGH);
    digitalWrite(7,HIGH);
```

```
digitalWrite(8,HIGH);
break;
case 7:
    digitalWrite(2,HIGH);
    digitalWrite(3,HIGH);
    digitalWrite(4,HIGH);
    digitalWrite(5,LOW);
    digitalWrite(6,LOW);
    digitalWrite(7,LOW);
    digitalWrite(8,LOW);
break;

case 8:
    digitalWrite(2,HIGH);
    digitalWrite(3,HIGH);
    digitalWrite(4,HIGH);
    digitalWrite(5,HIGH);
    digitalWrite(6,HIGH);
    digitalWrite(7,HIGH);
    digitalWrite(8,HIGH);
break;

case 9:
    digitalWrite(2,HIGH);
    digitalWrite(3,HIGH);
    digitalWrite(4,HIGH);
    digitalWrite(5,HIGH);
    digitalWrite(6,LOW);
    digitalWrite(7,HIGH);
    digitalWrite(8,HIGH);
```

```
break;  
  
default:  
  
break;  
  
}  
Serial.print("il valore del contatore vale= ");  
Serial.println(contatore);  
statoPulsprecstart=statoPulsstart;  
statoPulsprecstop=statoPulsstop;  
}
```

ESERCITAZIONE N°16: SISTEMA DI CHIAMATA 7 STANZE

Progettare un circuito che visualizzi, su un Display a 7 segmenti, il numero corrispondente al tasto premuto di una pulsantiera che ne contiene 7. Contemporaneamente alla visualizzazione del numero del pulsante chiamante dovrà essere emesso un segnale acustico.

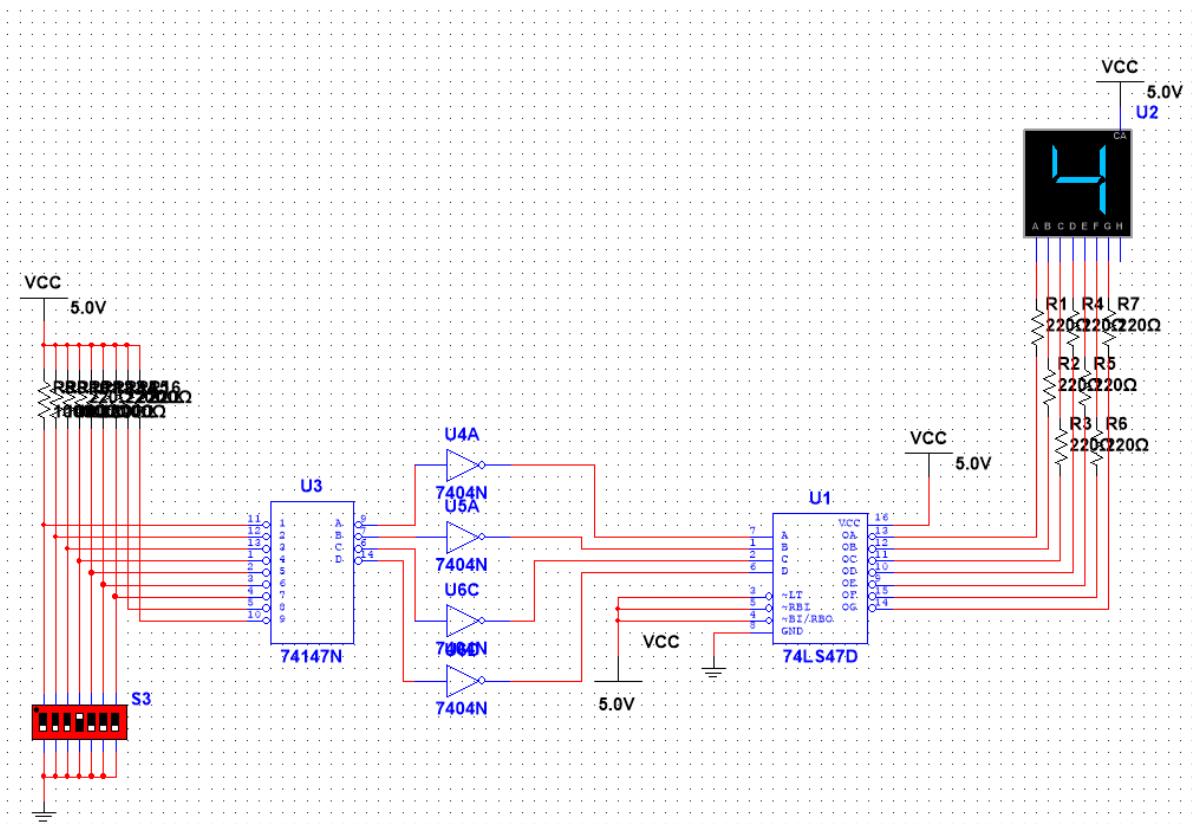
Note

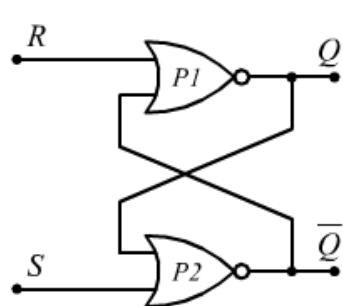
Il progetto non è altro che l'assemblaggio di un circuito combinatorio che può utilizzare tutti o in parte i dispositivi seguenti: **Encoder, Decoder, Decoder/Driver, Display a 7 segmenti, MUX e DEMUX.**

Nella dispensa sui circuiti combinatori fornita ci sono tutti gli elementi che possono essere utili alla realizzazione del progetto.

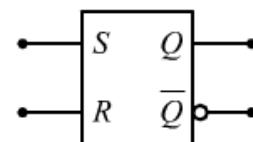
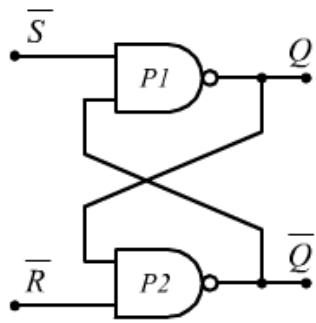
Consegne

- 1 Dopo aver creato una bozza del circuito necessario al progetto, implementarlo e simularlo al Multisim o TinkerCad.
- 2 Implementare il circuito su breadboard ed eseguire le prove con gli strumenti di laboratorio

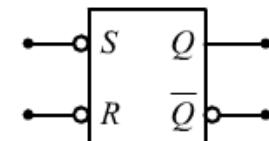
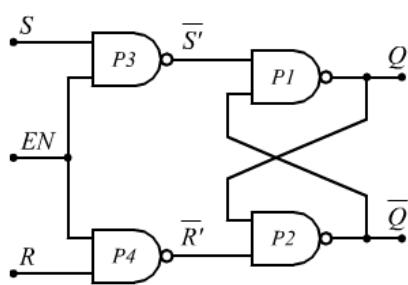


LATCH SR REALIZZATO A PORTE LOGICHE NOR

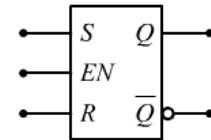
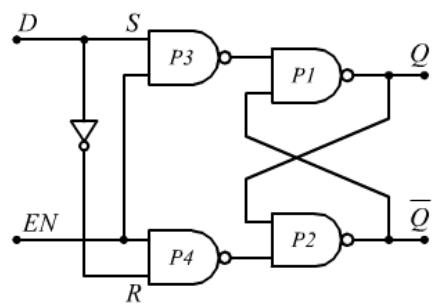
S	R	Q_{n+1}	\bar{Q}_{n+1}	
0	0	Q_n	\bar{Q}_n	memoria
0	1	0	1	reset
1	0	1	0	set
1	1	X	X	non ammesso

**LATCH SR REALIZZATO A PORTE LOGICHE NAND**

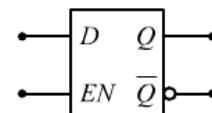
\bar{S}	\bar{R}	Q_{n+1}	\bar{Q}_{n+1}	
0	0	X	X	non ammesso
0	1	1	0	set
1	0	0	1	reset
1	1	Q_n	\bar{Q}_n	memoria

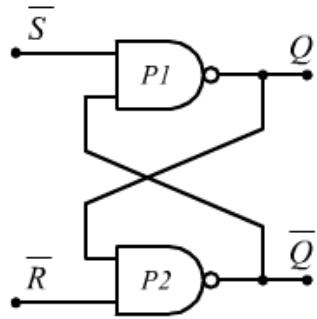
**LATCH SR CON ENABLE REALIZZATO A PORTE LOGICHE NAND**

EN	S	R	Q_{n+1}	
0	X	X	Q_n	memoria
1	0	0	Q_n	memoria
1	0	1	0	reset
1	1	0	1	set
1	1	1	X	non ammesso

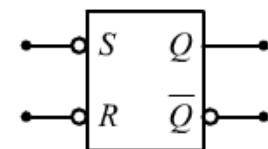
**LATCH TIPO D CON ENABLE A PORTE LOGICHE NAND**

EN	D	Q_{n+1}	
0	X	Q_n	memoria
1	0	0	reset
1	1	1	set



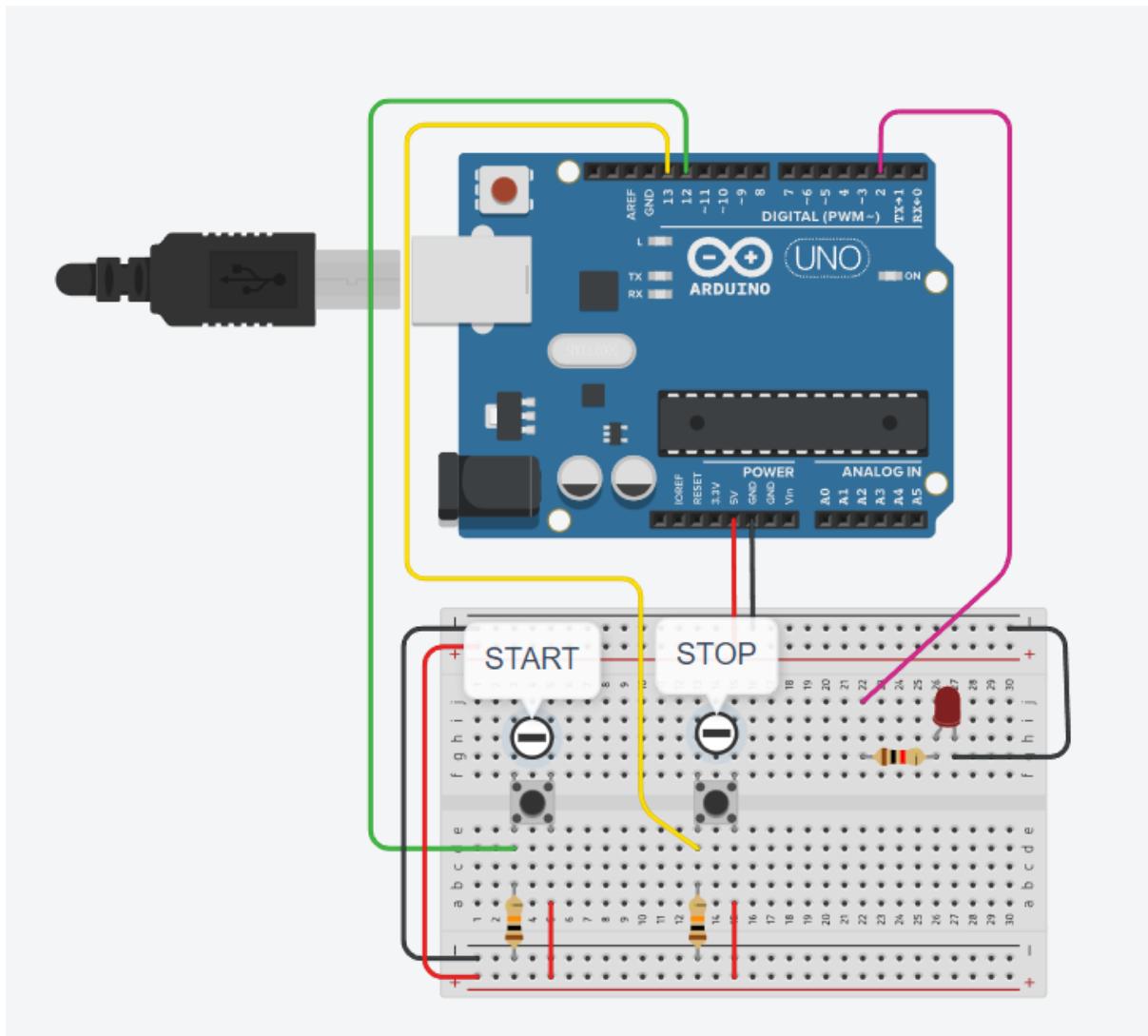
Compito di Telecomunicazioni Pratica - montaggio su breadboard**LATCH SR REALIZZATO A PORTE LOGICHE**

\overline{S}	\overline{R}	Q_{n+1}	\overline{Q}_{n+1}	
0	0	X	X	non ammesso
0	1	1	0	set
1	0	0	1	reset
1	1	Q_n	\overline{Q}_n	memoria



Portare il segnale in ingresso con due pulsanti in logica positiva, visualizzare il segnale con due led in logica negativa.

Esercizio con Arduino: Latch SET RESET



```
// SIMULAZIONE LATCH SET RESET
//
bool statostart;
bool statostartold=0;
bool statostop;
bool statostopold=0;

bool statoled;

void setup()
{
    pinMode(13,INPUT);
    pinMode(12,INPUT);
    pinMode(2,OUTPUT);}

void loop()
{
    statostart=digitalRead(12);
    statostop=digitalRead(13);
    if(statostart==1 && statostartold==0){
        statoled=1;}
    if(statostop==1 && statostopold==0){
        statoled=0;}
    digitalWrite(2,stated);
}

statostartold=statostart;
statostopold=statostop;
```

Circuito Sample & Hold

Sample and Hold

• SCOPO

Realizzare e analizzare il funzionamento di un circuito S/H che impiega un interruttore analogico integrato CMOS, per esempio il CD4016. Utilizzare un circuito S/H integrato, il classico LF398.

La simulazione consente di approfondire e verificare i concetti fondamentali relativi al campionamento e mantenimento.

Strumenti	Componenti
<ul style="list-style-type: none"> Oscilloscopio Alimentatore Generatore di segnali 	<ul style="list-style-type: none"> Integrati: TL082 o $2 \times$ TL081, CD4016, NE555, LF398 Resistori: $1 \text{ k}\Omega$; $3,3 \text{ k}\Omega$; $15 \text{ k}\Omega$ ($1/4 \text{ W}$) Condensatori: 1 nF; $3 \times 10 \text{ nF}$; $3 \times 100 \text{ nF}$ ceramici a disco Diodi: 1N4148

• PROCEDURA

- Realizzare il circuito di **Fig. 1** e applicare in ingresso un segnale va triangolare, di ampiezza compresa fra 0 V e 5 V e di frequenza $f_a = 1 \text{ kHz}$. Rilevare con l'oscilloscopio e disegnare le forme d'onda di v_a , V_c , v_o .

Calcolare e misurare la frequenza di campionamento f_c , il tempo di campionamento T_c e quello di mantenimento T_m e valutare il decadimento (droop) della tensione v_o . Ripetere le misure con un segnale di ingresso bipolare v_a con escursione $\pm 1 \text{ V}$.

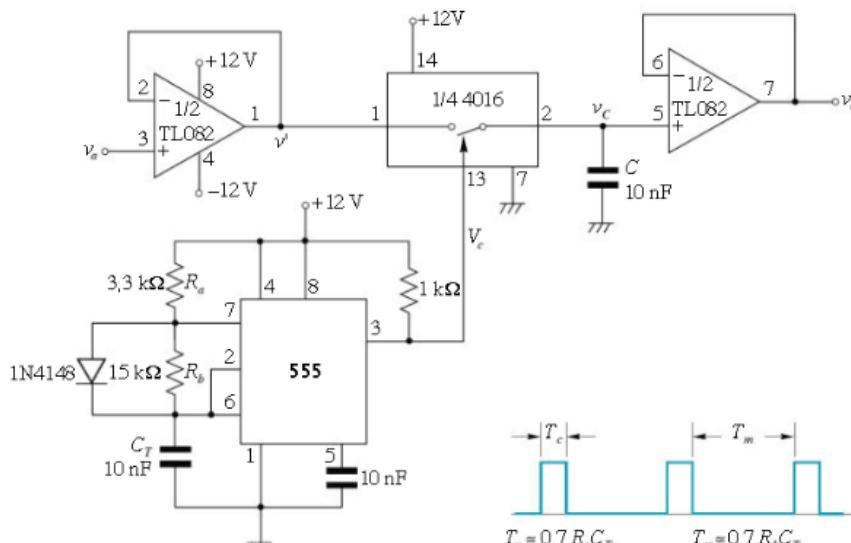


Fig. 1

- 2.** Esaminare i dati tecnici del S/H integrato LF398 (si possono usare anche LF198 oppure LF298 a caso analoghi) reperibili online e utilizzarlo per campionare segnali unipolari e bipolar. Applicare un segnale V_c con escursione da 0 V a 5 V.

• COMMENTI

- 1.** Il segnale di controllo V_c potrebbe essere fornito da un generatore di funzione anziché dall'astabile con 555. Il circuito di **Fig. 1** consente il campionamento di segnali unipolari compresi fra 0 V e circa 11 V.

La visualizzazione delle forme d'onda richiede un'attenta regolazione dei comandi Trigger e Hold dell'oscilloscopio. Conviene inserire condensatori ceramici a disco da 100 nF fra le alimentazioni e la massa per minimizzare i disturbi dovuti alle rapide commutazioni di V_c .

- 2.** Il circuito può essere realizzato sulla base dello schema di connessione tipico riportato in **Fig. 2**. Il valore della capacità può essere scelto in funzione delle prestazioni desiderate utilizzando i grafici *Acquisition time* e *Output droop rate*.

Si noti che nell'integrato proposto il segnale di controllo viene applicato all'interruttore interno attraverso uno stadio differenziale. Ciò consente di pilotare il S/H con vari tipi di segnali, unipolari (TTL, CMOS) e bipolar, purché entro i limiti delle tensioni di alimentazione.

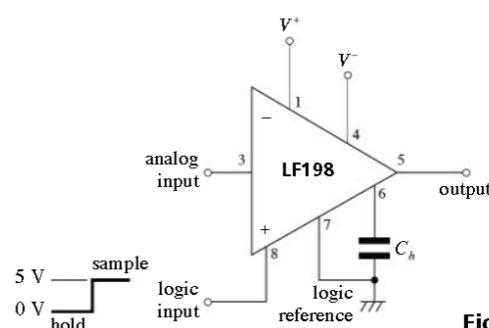


Fig. 2



Per campionare segnali bipolari, occorre che l'alimentazione del S/H sia bipolar. Il terminale 7 dell'integrato deve quindi essere portato a una tensione V^- (per es. -12 V).

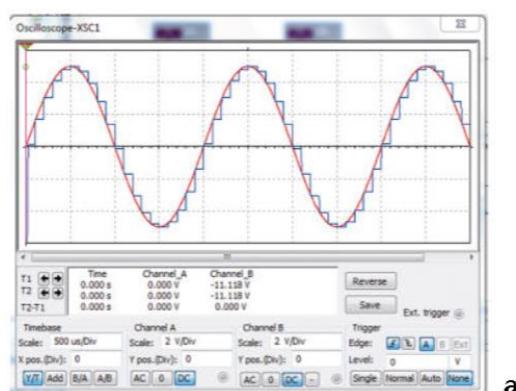


File di simulazione scaricabile dal sito di questo corso

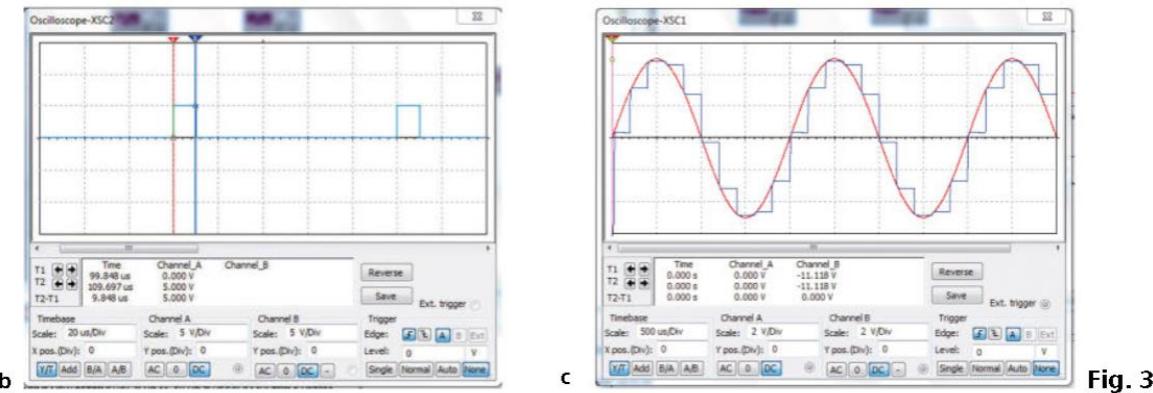


Il file V3_U12_LAB1.ms14 propone un circuito S/H realizzato con due operazionali e un interruttore a JFET. Si possono variare i parametri circuituali e verificare le prestazioni del sistema.

In **Fig. 3a** vediamo la forma d'onda del segnale da campionare a 500 Hz (in rosso) e il segnale campionato con $f_c=10$ kHz (in blu). In **Fig. 3b** vediamo il treno di impulsi di campionamento (in blu) a frequenza $f_c=10$ kHz e duty cycle=10%. In **Fig. 3c** vediamo la forma d'onda del segnale da campionare a 500 Hz (in rosso) e il segnale campionato con $f_c=5$ kHz (in blu). Si nota come il segnale campionato ottenuto sia meno fedele rispetto a quello ottenuto in **Fig. 3a**.



a



Prima parte esercitazione con operazionali

Vinput segnale triangolare unipolare tra 0 e 5V di frequenza= 1KHz.

Segnale di campionamento

$$T_c = 0.7 \cdot 3300 \cdot 10 \cdot 10^{-9} = 23.1 \text{ micros}$$

$$T_m = 0.7 \cdot 15000 \cdot 10 \cdot 10^{-9} = 105 \text{ micros}$$

$$T = T_c + T_m = 128 \text{ micros}$$

$$f = 7812 \text{ Hz circa 8KHz}$$

$$D_c = 22\%$$

materiale:

n° 1 integrato TL082CD

CD4016 (interruttore CMOS quindi comandato in tensione)

Seconda parte esercitazione con integrato LF398

Segnale Vc da 0 a 5V

Materiale Fase 1:

LF398

Vingresso = onda triangolare tra 0 e 5V a 1Khz.

sinusoidale avente 2Vpp e frequenza 1kHz

Segnale impulsivo= 2Vpp con dc=10% con frequenza 30kHz

Serie di Fourier e Analisi dello Spettro

Teoria: Una forma d'onda periodica $f(t)$ è data dalla somma di infinite forme d'onda sinusoidali aventi differenti caratteristiche di frequenza, ampiezza e fase.

Possiamo scrivere nel seguente modo la “**Serie di Fourier**”:

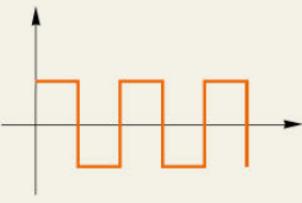
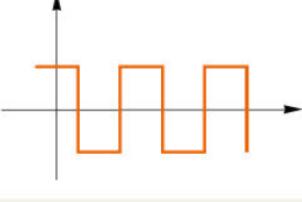
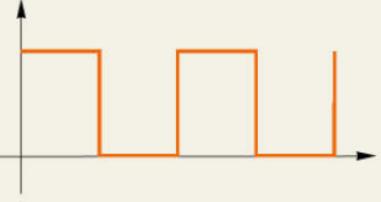
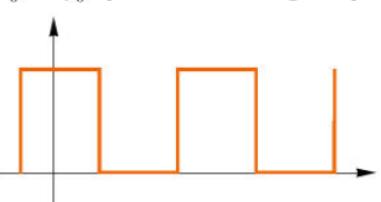
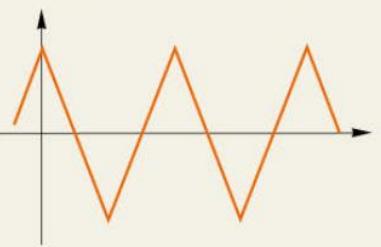
$$f(t) = C_0 + \sum_{k=1}^{\infty} C_k \sin(k\omega t + \varphi_k)$$

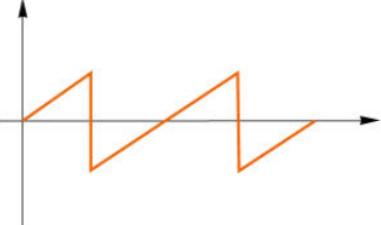
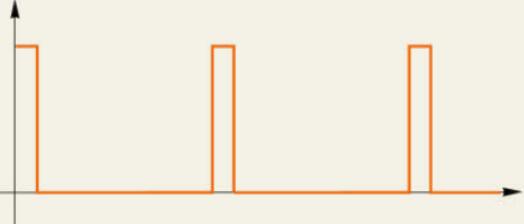
La serie di Fourier è data dalla somma di C_0 che è la componente media del segnale e una serie infinita di sinusoidi.

La prima sinusoide ha la stessa frequenza del segnale periodico ed è detta armonica fondamentale. Le altre armoniche hanno frequenza multiple della prima e generalmente ma non sempre ampiezza decrescente.

Le armoniche che formano il segnale possono essere rappresentate con delle linee verticali di dimensione pari all'ampiezza della singola componente e posizione pari alla frequenza. Tale rappresentazione prende il nome di Spettro delle ampiezze.

Dal teorema si evince che dato un circuito al quale è applicata una forma d'onda periodica $f(t)$, per determinare la risposta è applicabile il principio di sovrapposizione degli effetti di infinite forme d'onda sinusoidali.

Descrizione del segnale	Sviluppo in serie di Fourier
Onda quadra bipolare dispari V_M =valore alto $\omega_0=2\pi f_0$ =pulsazione del segnale periodico 	$f(t) = \frac{4V_M}{\pi} \sum_{m=0}^{\infty} \frac{\sin k\omega_0 t}{k} \quad k=2m+1$ $f(t) = \frac{4V_M}{\pi} \sin \omega_0 t + \frac{4V_M}{3\pi} \sin 3\omega_0 t + \frac{4V_M}{5\pi} \sin 5\omega_0 t + \dots$
Onda quadra bipolare pari V_M =valore alto $\omega_0=2\pi f_0$ =pulsazione del segnale periodico 	$f(t) = \frac{4V_M}{\pi} \sum_{m=0}^{\infty} (-1)^m \frac{\cos k\omega_0 t}{k} \quad k=2m+1$ $f(t) = \frac{4V_M}{\pi} \cos \omega_0 t - \frac{4V_M}{3\pi} \cos 3\omega_0 t + \frac{4V_M}{5\pi} \cos 5\omega_0 t - \frac{4V_M}{7\pi} \cos 7\omega_0 t + \dots$
Onda quadra unipolare dispari V_M =valore alto $\omega_0=2\pi f_0$ =pulsazione del segnale periodico 	$f(t) = \frac{V_M}{2} + \frac{2V_M}{\pi} \sum_{m=0}^{\infty} \frac{\sin k\omega_0 t}{k} \quad k=2m+1$ $f(t) = \frac{V_M}{2} + \frac{2V_M}{\pi} \sin \omega_0 t + \frac{2V_M}{3\pi} \sin 3\omega_0 t + \frac{2V_M}{5\pi} \sin 5\omega_0 t + \frac{2V_M}{7\pi} \sin 7\omega_0 t + \dots$
Onda quadra unipolare pari V_M =valore alto $\omega_0=2\pi f_0$ =pulsazione del segnale periodico 	$f(t) = \frac{V_M}{2} + \frac{2V_M}{\pi} \sum_{m=0}^{\infty} (-1)^m \frac{\cos k\omega_0 t}{k} \quad k=2m+1$ $f(t) = \frac{V_M}{2} + \frac{2V_M}{\pi} \cos \omega_0 t - \frac{2V_M}{3\pi} \cos 3\omega_0 t + \frac{2V_M}{5\pi} \cos 5\omega_0 t - \frac{2V_M}{7\pi} \cos 7\omega_0 t + \dots$
Onda triangolare pari V_M =valore di picco $\omega_0=2\pi f_0$ =pulsazione del segnale periodico 	$f(t) = \frac{8V_M}{\pi^2} \sum_{m=0}^{\infty} \frac{\cos k\omega_0 t}{k^2} \quad k=2m+1$ $f(t) = \frac{8V_M}{\pi^2} \cos \omega_0 t + \frac{8V_M}{9\pi^2} \cos 3\omega_0 t + \frac{8V_M}{25\pi^2} \cos 5\omega_0 t + \frac{8V_M}{49\pi^2} \cos 7\omega_0 t + \dots$

Descrizione del segnale	Sviluppo in serie di Fourier
Onda triangolare dispari V_M =valore di picco $\omega_0=2\pi f_0$ =pulsazione del segnale periodico 	$f(t) = \frac{8V_M}{\pi^2} \sum_{m=0}^{\infty} (-1)^m \frac{\sin k\omega_0 t}{k^2} \quad k=2m+1$ $f(t) = \frac{8V_M}{\pi^2} \sin \omega_0 t - \frac{8V_M}{9\pi^2} \sin 3\omega_0 t + \frac{8V_M}{25\pi^2} \sin 5\omega_0 t + \dots - \frac{8V_M}{49\pi^2} \sin 7\omega_0 t + \dots$
Onda a dente di sega V_M =valore di picco $\omega_0=2\pi f_0$ =pulsazione del segnale periodico 	$f(t) = \frac{2V_M}{\pi} \sum_{m=0}^{\infty} (-1)^{m+1} \frac{\sin m\omega_0 t}{m}$ $f(t) = \frac{2V_M}{\pi} \left(\sin \omega_0 t - \frac{1}{2} \sin 2\omega_0 t + \frac{1}{3} \sin 3\omega_0 t + \dots - \frac{1}{4} \sin 4\omega_0 t + \frac{1}{5} \sin 5\omega_0 t + \dots \right)$
Treno d'impulsi (durata τ) V_M =valore alto $\omega_0=2\pi f_0$ =pulsazione del segnale periodico 	$f(t) = V_M \frac{\tau}{T} + \frac{2V_M}{\pi} \sum_{m=1}^{\infty} \left[\left(\frac{\sin m k \pi}{m} \right) \cos m \omega_0 t \right] \quad k = \frac{\tau}{T}$ $f(t) = V_M \frac{\tau}{T} + \frac{2V_M}{\pi} \left[(\sin k \pi) \cos \omega_0 t + \left(\frac{\sin 2k\pi}{2} \right) \cos 2\omega_0 t + \left(\frac{\sin 3k\pi}{3} \right) \cos 3\omega_0 t + \dots + \left(\frac{\sin 4k\pi}{4} \right) \cos 4\omega_0 t + \dots \right]$

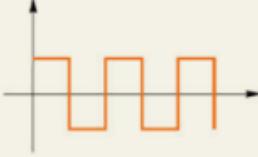
Si riporta un esempio per la forma d'onda bipolare dispari avente

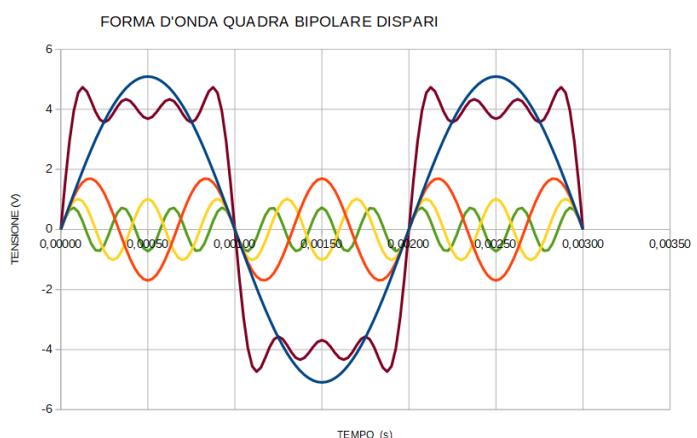
Vmax=4V e f=500Hz

si ripeta l'esercizio:

- **onda quadra bipolare pari con Vmax=2V e f= 50 kHz**
- **onda triangolare dispari: Vmax= 3V e f=10kHz**
- **treno di impulsi Vmax=2V, f=60kHz e Dc%=10%**

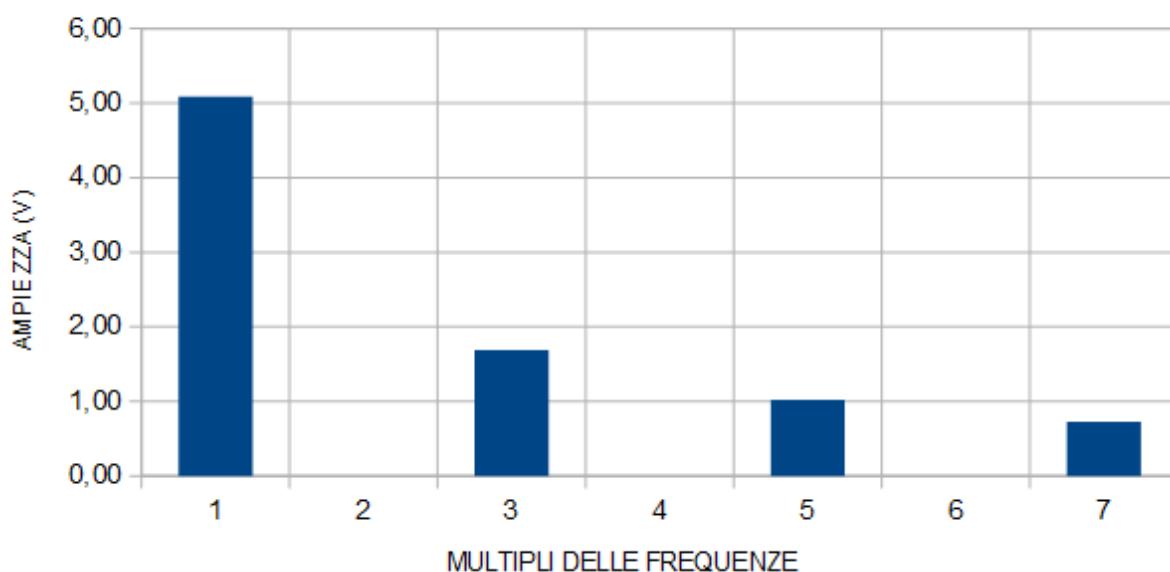
FORMA D'ONDA QUADRA BIPOLARE DISPARI

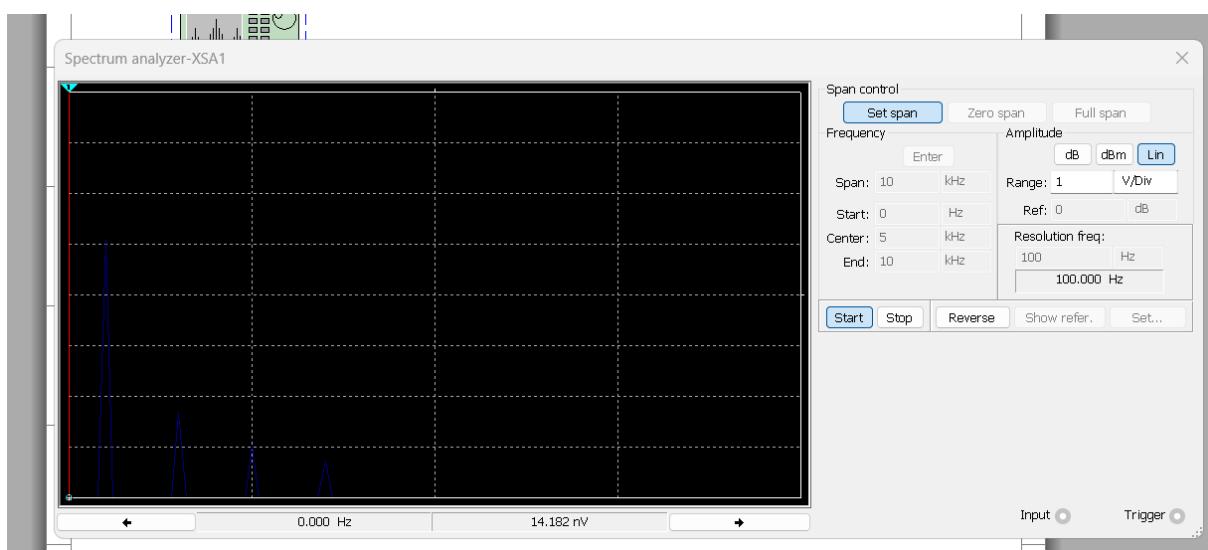
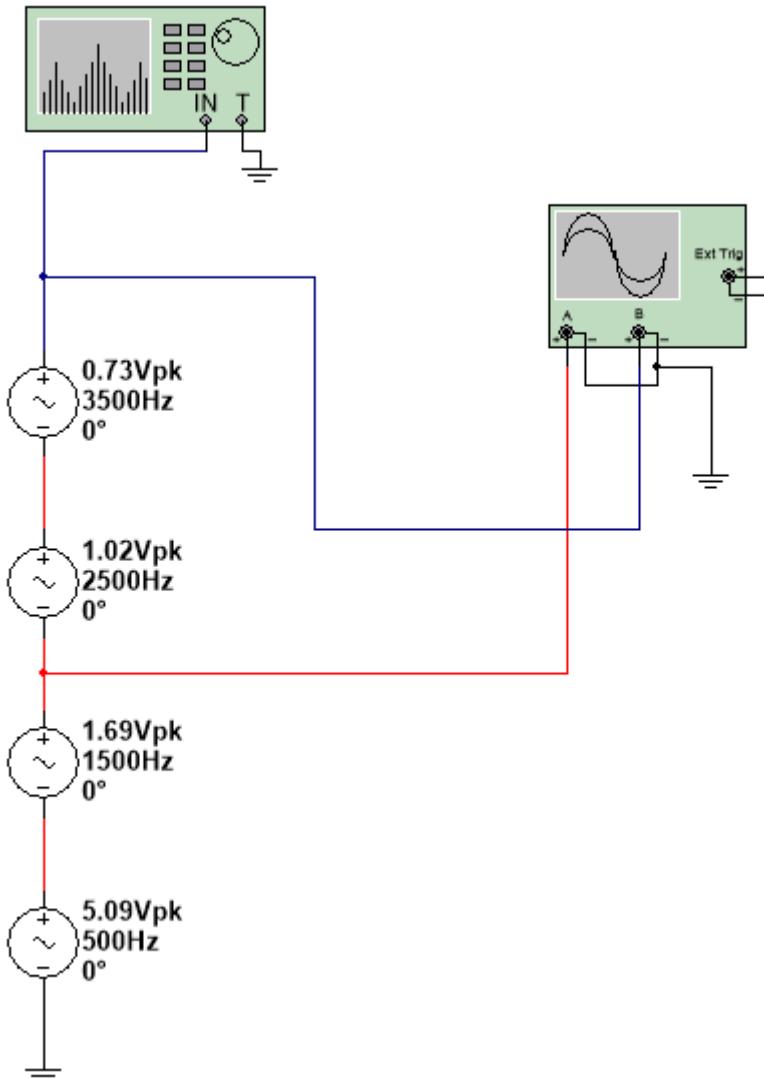
Descrizione del segnale		Sviluppo in serie di Fourier			
Onda quadra bipolare dispari		$f(t) = \frac{4V_M}{\pi} \sum_{m=0}^{\infty} \frac{\sin k\omega_0 t}{k}$	$k = 2m + 1$		
V_M = valore alto		$f(t) = \frac{4V_M}{\pi} \sin \omega_0 t + \frac{4V_M}{3\pi} \sin 3\omega_0 t + \frac{4V_M}{5\pi} \sin 5\omega_0 t + \dots$			
$\omega_0 = 2\pi f_0$ = pulsazione del segnale periodico					
					
V_M	4 V	Valore alto della forma d'onda quadra			
f	500 Hz	Frequenza della forma d'onda quadra			
T	0,002 s	Periodo della forma d'onda quadra			
ω	3142 rad/s	Pulsazione della forma d'onda quadra			
ΔT	2,5E-05 s	Intervallo colonna dei tempi per tracciare il grafico			
4V_M/π	5,09 V	termine che semplifica le formule da inserire nelle colonne			
t (s)	ARMONICA n° 1 (V)	ARMONICA n° 3 (V)	ARMONICA n° 5 (V)	ARMONICA n° 7 (V)	SEGNAL RISULTANTE (V)
0,00000	0	0	0	0	0
0,00003	0,40	0,40	0,39	0,38	1,57
0,00005	0,80	0,77	0,72	0,65	2,94
0,00008	1,19	1,10	0,94	0,73	3,96
0,00010	1,57	1,37	1,02	0,59	4,55
0,00013	1,95	1,57	0,94	0,28	4,74
0,00015	2,31	1,68	0,72	-0,11	4,60
0,00018	2,66	1,69	0,39	-0,47	4,27
0,00020	2,99	1,61	0,00	-0,69	3,92
0,00023	3,31	1,45	-0,39	-0,71	3,66
0,00025	3,60	1,20	-0,72	-0,51	3,57
0,00028	3,87	0,89	-0,94	-0,17	3,65
0,00030	4,12	0,52	-1,02	0,22	3,85
0,00033	4,34	0,13	-0,94	0,55	4,09
0,00035	4,54	-0,27	-0,72	0,72	4,27

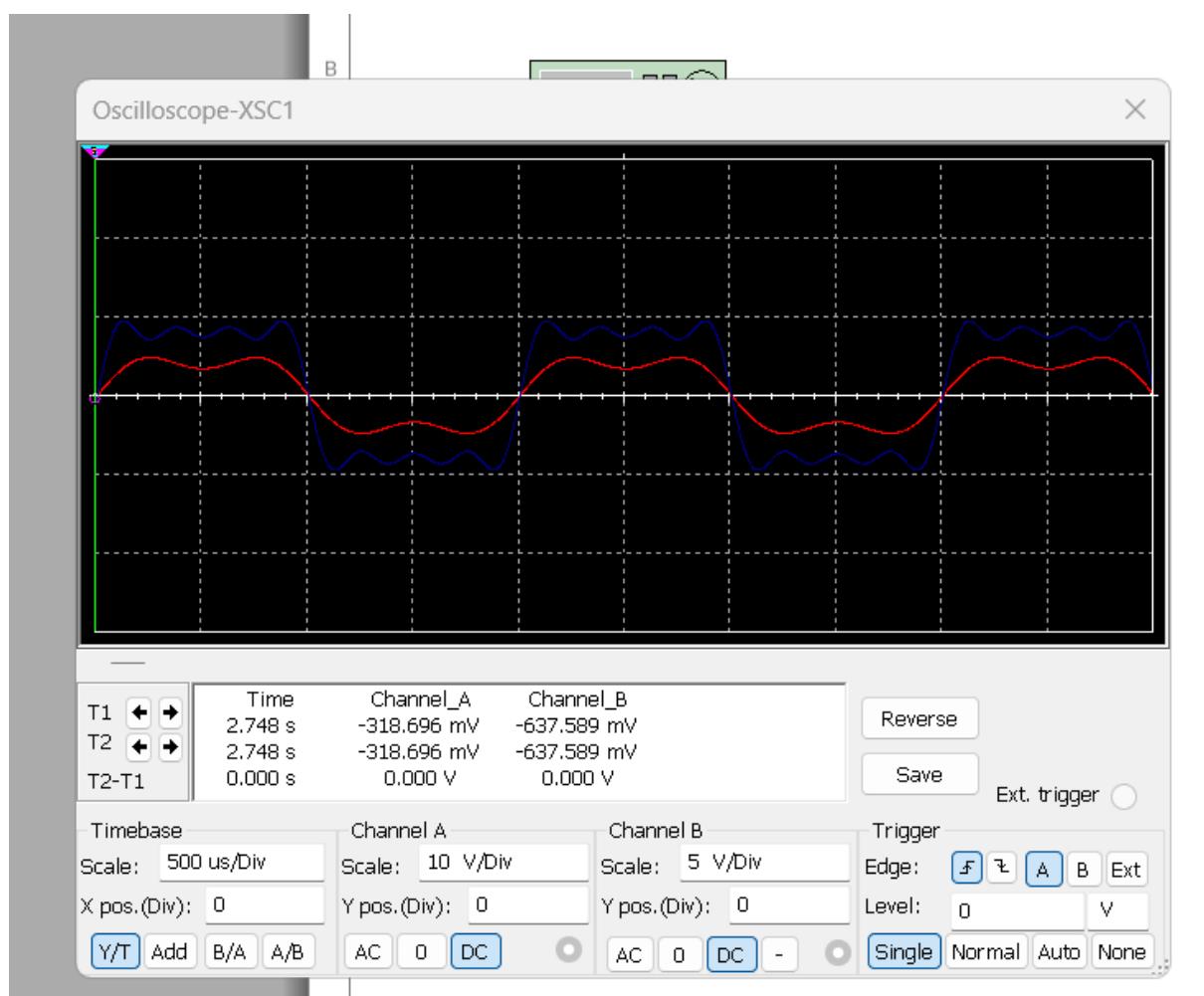


Analisi spettrale della ampiezza del segnale

	pulsazione w	frequenza	V _{max}
	rad/s	Hz	V
1 ARMONICA	3142	500,25	5,09
2 ARMONICA			0,00
3 ARMONICA	9425	1500,76	1,69
4 ARMONICA			0,00
5 ARMONICA	15708	2501,27	1,02
6 ARMONICA			0,00
7 ARMONICA	21991	3501,78	0,73

SPETTRO DELLE AMPIEZZE



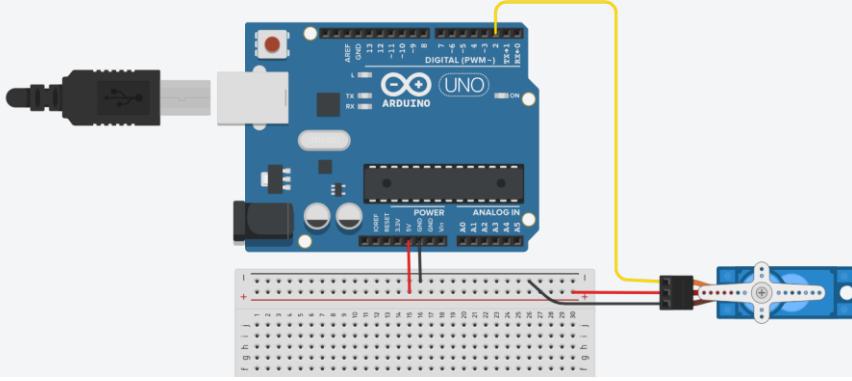


Arduino: inviare comandi da Monitor Seriale

L'esercitazione consiste nel pilotare l'accensione del led collegato ad arduino inviando un comando da Tastiera del PC tramite l'interfaccia “Monitor Seriale”

Bluetooth_monitor_Seriale \$

```
1 int led=6;
2 char comando;
3 bool statoled;
4
5 void setup() {
6     //inizializzo la comunicazione Seriale
7     Serial.begin(9600);
8     pinMode(6,OUTPUT); }
9
10 void loop() {
11     comando=Serial.read();
12     if (comando=='A') {
13         statoled=0; }
14     if (comando=='S') {
15         statoled=1; }
16
17 digitalWrite(led,statedoled);
18 }
```

TITOLO:	<i>Posizionamento albero del servomotore</i>	
MATERIALI:	<i>PC + Arduino + cavo USB, Servomotore, Breadboard, cavetti.</i>	
SCHEMA ELETTRICO:		
CENNI TEORICI:	<p>I servomotori sono componenti con i quali è possibile controllare in modo preciso la posizione dell'albero. La movimentazione possibile è di 180°. Il servomotore è dotato di tre cavi, due dei quali alimentano il motore VCC e GND, il terzo cavo invia il segnale che consente la regolazione della posizione.</p>	
IDE:	<p><u>inizio programma</u></p> <p>1) richiamare la libreria</p> <p>2) dichiarare il nome dell'oggetto</p> <p><u>nel voidsetup()</u></p> <p>dichiarare il pin che invia il segnale di comando</p> <p><u>nel voidloop()</u></p> <p>definizione della posizione dell'albero. val è un numero compreso tra 0° e 180°</p>	<pre>#include<Servo.h> Servo.servoname; servoname.attach(numeropin); nomeservo.write(val);</pre>

ARGOMENTO:	ARDUINO - SERVOMOTORI
ESERCIZIO PROPOSTO N.1	<i>implementare un programma che posiziona l'asse del motore a 45°.</i>
ESERCIZIO PROPOSTO N.2	<i>implementare un programma che partendo da 0°, incrementi la rotazione fino a 180°, per poi ritornare a 0°, incrementando e decrementando l'angolo con passi di 1°.</i>

SOLUZIONE ESERCIZIO PROPOSTO N.1

```
1  /*richiamo la libreria Servo.h
2   e creo l'oggetto Servo*/
3
4
5 #include <Servo.h>
6
7 Servo motorestudio;
8
9
10 void setup() {
11   pinMode(2, OUTPUT);
12   // dichiarazione del pin PWM
13   motorestudio.attach(2);
14 }
15
16 void loop() {
17   //dichiaro la posizione del motore
18   motorestudio.write(45);
19 }
```

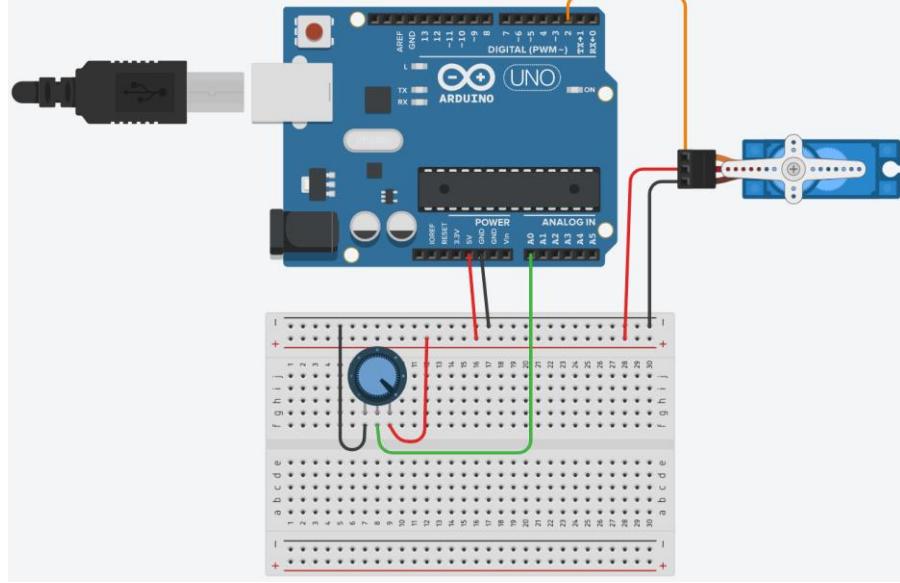
SOLUZIONE ESERCIZIO PROPOSTO N.2

```
1  /*richiamo la libreria Servo.h
2   e creo l'oggetto Servo*/
3
4
5 #include <Servo.h>
6
7 Servo motorestudio;
8 int gradi;
9
10 void setup() {
11   pinMode(2, OUTPUT);
12   // dichiarazione del pin PWM
13   motorestudio.attach(2);
14 }
15
16 void loop() {
17   //incremento da 0° a 180°
18   for(gradi=0;gradi<=180;gradi++){
19     motorestudio.write(gradi);
20     delay(10);}
21   //decremento da 180° a 0°
22   for(gradi=180;gradi>=0;gradi--){
23     motorestudio.write(gradi);
24     delay(10);}
25 }
```

ARGOMENTO:	ARDUINO - SERVOMOTORI
TITOLO:	<i>implementare un programma che partendo da 0° incrementi la rotazione fino a 180°, per poi decrementarla da 180° a 0° con passi di 1°. La velocità della rotazione dipende dal valore letto da potenziometro.</i>
MATERIALI:	PC + Arduino + cavo USB, Servomotore, Breadboard, cavetti, potenziometro da 10kOhm
SCHEMA ELETTRICO:	
IDE:	<p>Leggere il valore in ingresso ad A0 al quale è collegato il potenziometro tramite:</p> <pre>variableA0 = analogRead(A0);</pre> <p>la variableA0 acquisita sarà un valore intero compreso tra 0 e 1023.</p> <p>Quando il valore letto vale 0 si applichi un delay=10ms.</p> <p>Quando il valore letto vale 1023 si applichi un delay=1000ms.</p> <pre>delay(rit);</pre> <p>$rit = 10 + (((1000-10)/1023)*valoreA0)$</p> <p>In alternativa usare la funzione map che compie la proporzione:</p> <pre>rit = map(valoreA0,0,1023,10,1000)</pre>

Soluzione:

```
1  /*richiamo la libreria Servo.h
2  e creo l'oggetto Servo
3  dichiaro la variabile valoreA0 come intera 0-1023,
4  e la varibile gradi e ritardo. */
5
6 #include <Servo.h>
7 Servo motorestudio;
8
9 int valoreA0;
10 int gradi;
11 int rit;
12
13 void setup() {
14     Serial.begin(9600);
15     pinMode(2, OUTPUT);
16     // dichiarazione del pin PWM
17     motorestudio.attach(2);}
18
19 void loop() {
20     for(gradi=0;gradi<=180;gradi++){
21         //lettura della variabile valoreA0 valore compre tra 0 e 1023
22         valoreA0=analogRead(A0);
23         //proporziona la variabile valoreA0 al valore del ritardo
24         // da applicare
25         // quando valoreA0=0 rit=10ms;
26         // quando valoreA0=1023 rit=1000ms;
27         rit = 10+(990.0/1023)*valoreA0;
28         Serial.print ("il valoreA0=");
29         Serial.println(valoreA0);
30         Serial.print ("rit=");
31
32         Serial.println(rit);
33         //dichiaro la posizione del motore
34         motorestudio.write(gradi);
35         delay(rit);}
36         for(gradi=180;gradi>=0;gradi--){
37             //lettura della variabile valoreA0 valore compre tra 0 e 1023
38             valoreA0=analogRead(A0);
39             //proporziona la variabile valoreA0 al valore del ritardo
40             // da applicare
41             // quando valoreA0=0 rit=10ms;
42             // quando valoreA0=1023 rit=1000ms;
43             rit = 10+(990.0/1023)*valoreA0;
44             Serial.print ("il valoreA0=");
45             Serial.println(valoreA0);
46             Serial.print ("rit=");
47             Serial.println(rit);
48             //dichiaro la posizione del motore
49             motorestudio.write(gradi);
50             delay(rit);}
51 }
```

ARGOMENTO:	ARDUINO - SERVOMOTORI
TITOLO:	<i>Posizionamento albero del servomotore proporzionalmente al segnale applicato tramite potenziometro.</i>
MATERIALI:	<i>PC + Arduino + cavo USB, Servomotore, Breadboard, cavetti, potenziometro da 10kOhm</i>
SCHEMA ELETTRICO:	
IDE:	<p>Leggere il valore in ingresso ad A0 al quale è collegato il potenziometro tramite:</p> <pre>variabileA0 = analogRead(A0);</pre> <p>la variabileA0 acquisita sarà un valore intero compreso tra 0 e 1023.</p> <p>Quando il valore letto vale 0 si applichi un angolo di 0°, quando il valore vale 1023 si applichi un valore di 180°.</p> <pre>nomeservo.write(val);</pre> $\text{val} / \text{variabileA0} = 180.0 / 1023$ $\text{val} = 180.0 / 1023 \times \text{variabile A0}$ <p>In alternativa usare la funzione map che compie dato un valore la proporzione:</p> <pre>val = map(valoreA0, 0, 1023, 0, 180)</pre>

Soluzione:

```
1  /*richiamo la libreria Servo.h
2   e creo l'oggetto Servo
3   dichiaro la variabile valoreA0 come intera 0-1023,
4   e la variabile gradi; */
5
6 #include <Servo.h>
7
8 Servo motorestudio;
9 int valoreA0;
10 int gradi;
11
12 void setup() {
13   Serial.begin(9600);
14   pinMode(2, OUTPUT);
15   // dichiarazione del pin PWM
16   motorestudio.attach(2);}
17
18 void loop() {
19   //lettura della variabile valoreA0 valore compre tra 0 e 1023
20   valoreA0=analogRead(A0);
21
22   //proporziona la variabile valoreA0 al valore dell'angolo
23   //da applicare
24   // quando valoreA0=0 angolo=0°,
25   // quando valoreA0=1023 angolo=180°
26
27   gradi=valoreA0*180.0/1023;
28   Serial.print ("il valoreA0=");
29   Serial.println(valoreA0);
30   Serial.print ("gradi=");
31   Serial.println(gradi);
32
33   //dichiaro la posizione del motore
34   motorestudio.write(gradi);
35   delay(10);}
```

ARGOMENTO:	ARDUINO - SENSORE HC-SR04
TITOLO:	<i>Implementare un programma in modo che quando viene rilevata la presenza di un ostacolo da parte del sensore entro un certa distanza il buzzer suoni.</i>
MATERIALI:	PC + Arduino + cavo USB, Sensore HCSR04, buzzer Breadboard, cavetti.
SCHEMA ELETTRICO:	
CENNI TEORICI:	<p>Il sensore HCSR04 è un sensore ad ultrasuoni che consente di misurare la distanza da un ostacolo.</p> <p>Il componente presenta quattro terminali, VCC e GND (per l'alimentazione), TRIGGER e ECHO.</p> <p>Al terminale TRIGGER invia un impulso di durata pari a 10 microsecondi, la linea ECHO riceve il segnale di ritorno. Il suono che infrangendosi sull'ostacolo ritorna al sensore. La durata dell'impulso di ritorno coincide con il tempo impiegato dal segnale per percorrere TRIGGER-OSTACOLO-ECHO.</p> <p>La funzione che ci consente di determinare il tempo è: <code>Techo=pulseIn(pin,timeout);</code> in microsecondi</p> <p>Il tempo OSTACOLO-ECHO vale metà del tempo rilevato.</p> <p>Lo spazio espresso in centimetri è dato da: $s=v*t = 343,8\text{m/s} * \text{Techo}(\text{microsec}) / 2 = 0,0172 * \text{Techo}$</p>
IDE:	Per determinare la durata dell'impulso del segnale di ritorno in microsecondi:

`Techo=PulseIn(pin,timeout);`
`pin è il pin di Arduino al quale è collegato il terminale Echo del sensore, timeout è il valore massimo di tempo dopo il quale la trasmissione si interrompe in quanto l'ostacolo è troppo lontano.`
`Per attivare il buzzer utilizzo la funzione:`
`Tone(pin,frequenza in Hz, durata in millisecondi);`

Soluzione:

```

1 //Techo misura della durata dell'impulso di ritorno
2 //valDist è la distanza dall'ostacolo
3 //temp è il tempo di riacquisizione di nuovo ciclo.
4 unsigned long Techo=0;
5 int valDist;
6 unsigned long temp;
7 void setup()
8 {
9     //pin 3 applico il segnale impulsivo di durata 10microsec,
10    pinMode(3,OUTPUT);
11    //pin 2 leggo il segnale Techo di ritorno
12    pinMode(2,INPUT);
13    //buzzer
14    pinMode(9,OUTPUT);
15    //inizializzo la comunicazione Seriale
16    Serial.begin(9600);
17 }
18 void loop()
19 {
20     // applico il segnale impulsivo di durata 10 microsecondi
21     digitalWrite(3,HIGH);
22     delayMicroseconds(10);
23     digitalWrite(3,LOW);
24     //leggo la durata del segnale di ritorno
25     Techo=pulseIn(2,1000000);
26     //calcolo la distanza dall'ostacolo
27     valDist=0.01723*Techo;
28     Serial.print("Techo=");
29     Serial.println(Techo);
30     Serial.print("distanza=");
31     Serial.println(valDist);
32     // Se la distanza è minore di 120 cm suona il buzzer
33     if(valDist<=120) {
34         tone(9,1000,50);
35         //attendi sempre meno tempo per ripartire con l'aquisizione
36         //dei valori Techo del ciclo successivo.
37         temp=map(valDist,0,120,200,1000);
38         delay(temp);
39     }
40 }
```

ARGOMENTO:	SCHEMI ELETTROMECCANICI
<i>TITOLO:</i>	- SIMBOLI
	SA: selettore SB: pulsante SQ: sensore finecorsa KA: rele' KM: contattore HL: segnalatore luminoso YV: eletrovalvola KT: timer

ARGOMENTO:	PLC - HARDWARE
<i>TITOLO:</i>	<i>Componenti hardware di un PLC.</i>
CENNI TEORICI:	
Il plc è un controllore a logica Programmabile, cioè il ciclo di funzionamento è affidato ad una serie di istruzioni presenti nel programma che l'utilizzatore sviluppa. Per le logiche cablate il ciclo di funzionamento era affidato al tipo di componenti scelti	

e ai relativi collegamenti.

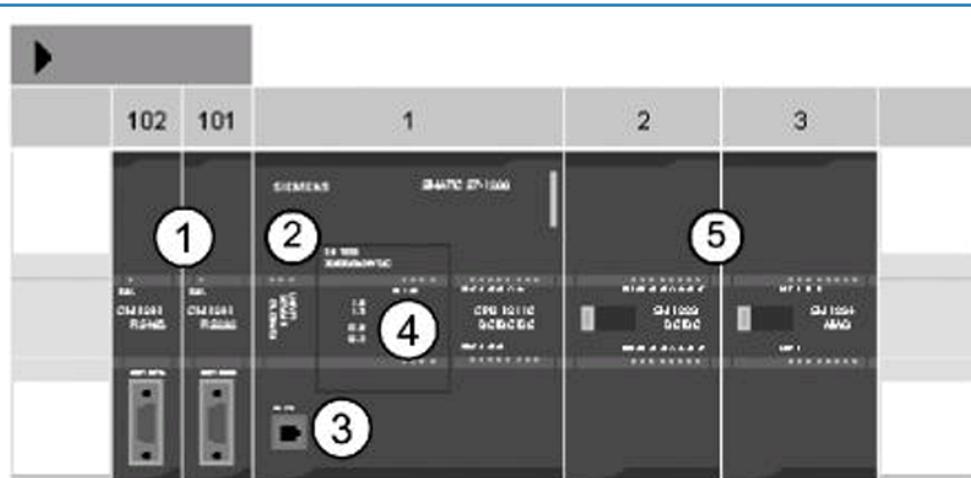
AI PLC si connettono direttamente dispositivi che fungono da ingressi e dispositivi che fungono da uscite.

Ad esempio letto lo stato di un pulsante in ingresso, modifica lo stato di una lampada collegata in uscita.

L'assemblaggio avviene, inserendo su di una guida RACK tipo DIN, tutti i moduli.

I moduli vengono inseriti nella guida rack secondo l'ordine sotto riportato da sinistra verso destra:

- moduli di comunicazione,
- alimentatore
- CPU con optional SIGNAL BOARD
- moduli di espansione ingressi e uscite chiamate SIGNAL MODULE



- ① Modulo di comunicazione (CM): fino a 3, inseriti nei posti connettore 101, 102 e 103
- ② CPU: posto connettore 1
- ③ Porta Ethernet della CPU
- ④ Signal board (SB): 1 al massimo, inserita nella CPU
- ⑤ Modulo di I/O (SM) per I/O digitali e analogici: fino a 8, inseriti nei posti connettore da 2 a 9 (la CPU 1214C ne consente 8, la CPU 1212C 2, la CPU 1211C nessuno)

Vediamo singolarmente il significato:

ALIMENTATORE:

L'alimentatore è il dispositivo modulare che fornisce energia al sistema, a 24V per potenze che variano in base al tipo di CPU e al numero e assorbimento delle uscite (lampade, elettrovalvole ecc....). Siemens fornisce alimentatori modulari a 24V con correnti di valore crescente 2,5 A, 5 A, 10 A, idoneo a soddisfare la potenza richiesta.

CPU:

In linea di massima la CPU riunisce in un'unica apparecchiatura:

- ▶ un microprocessore,
- ▶ memorie RAM e ROM.
- ▶ I/O digitali e analogiche on board
- ▶ PROFINET integrato,

La CPU svolge ciclicamente una serie di operazioni sequenziali seguendo le istruzioni memorizzate nel programma, questa attività prende il nome di ciclo di scansione e richiede un tempo (tempo di scansione) che dipende sia dal tipo di CPU che dalla complessità del programma, in genere questi tempi hanno il valore di poche decimi di secondi. La CPU controlla gli ingressi e modifica le uscite in base alla logica del programma utente.

La CPU S7-1200 è già dotata di un discreto numero di I/O digitali e ingressi e uscite analogiche (detti per questo on board) che ne permettono il collegamento con i sensori e gli attuatori del sistema

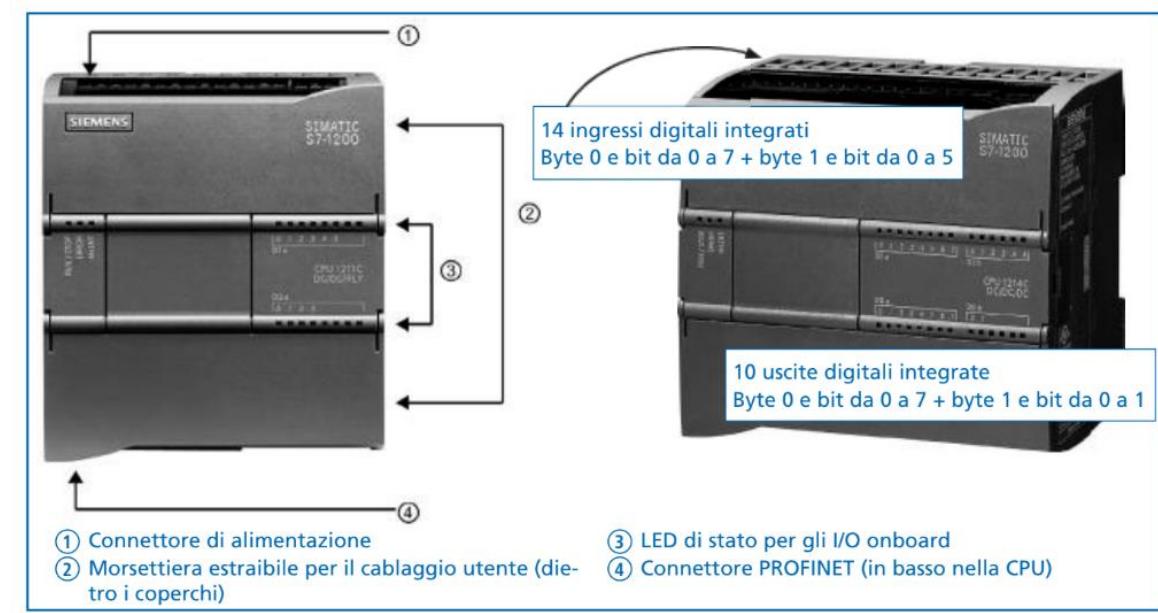
Analizzando la tabella sotto riportata la CPU 1215C presenta:

14 ingressi digitali on board

10 uscite digitali on board

2 ingressi analogici on board

2 uscite analogiche on board



Caratteristiche CPU S7-1200.

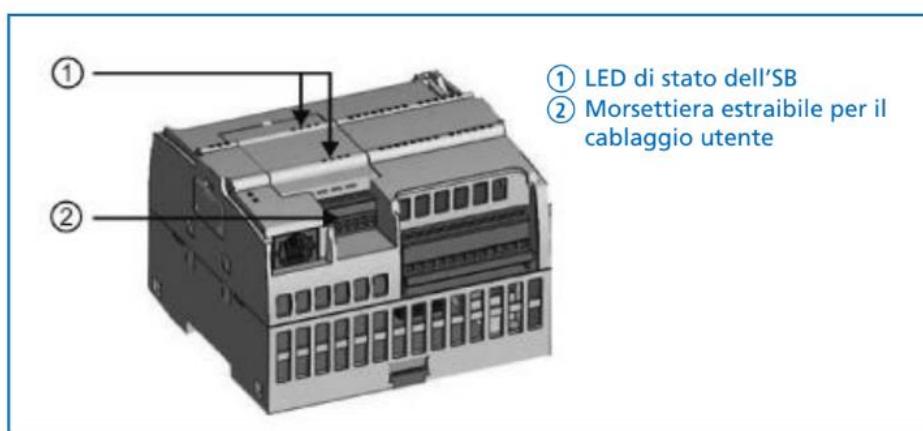
Caratteristiche	CPU 1211C	CPU 1212C	CPU 1214C	CPU 1215C	CPU 1217C
Versione	DC/DC/DC, AC/DC/relay, DC/DC/relay				
Memoria di lavoro integrata	50 KB	75 KB	100 KB	125 KB	150 KB
Memoria di caricamento integrata	1 MB	1 MB	4 MB	4 MB	4 MB
Memory card	SIMATIC Memory Card (opzionale)				
Ingressi/Uscite digitali integrate	6/4	8/6	14/10	14/10	14/10
Ingressi Analogici integrati	2	2	2	2	2
Uscite Analogiche integrate	0	0	0	2	2
Registri immagine di processo	1024 bytes per gli ingressi, 1024 bytes per le uscite				
Espansione con Signal Board	Max. 1	Max. 1	Max. 1	Max. 1	Max. 1
Espansione con moduli aggiuntivi	No	Max. 2	Max. 8	Max. 8	Max. 8

Caratteristiche	CPU 1214 FC	CPU 1215 FC
Versione	DC/DC/DC, DC/DC/relay	DC/DC/DC, DC/DC/relay
Memoria di lavoro integrata	125 KB	150 KB
Memoria di caricamento integrata	4 MB	4 MB
Memory card	SIMATIC Memory Card (opzionale)	SIMATIC Memory Card (opzionale)
Ingressi/Uscite digitali integrate	14/10	14/10
Ingressi Analogici integrati	2	2
Uscite Analogiche integrate	-	2
Registri immagine di processo	1024 bytes di ingresso, 1024 bytes di uscita	1024 bytes di ingresso, 1024 bytes di uscita
Espansione con Signal Board	Max. 1	Max. 1
Espansione con moduli aggiuntivi	Max. 8	Max. 8

Per comunicare con il computer la CPU S7-1200 mette a disposizione una porta PROFINET integrata.

Attraverso la rete PROFINET può comunicare con i pannelli HMI

Nella CPU è presente un alloggiamento per la “SIGNAL BOARD”.



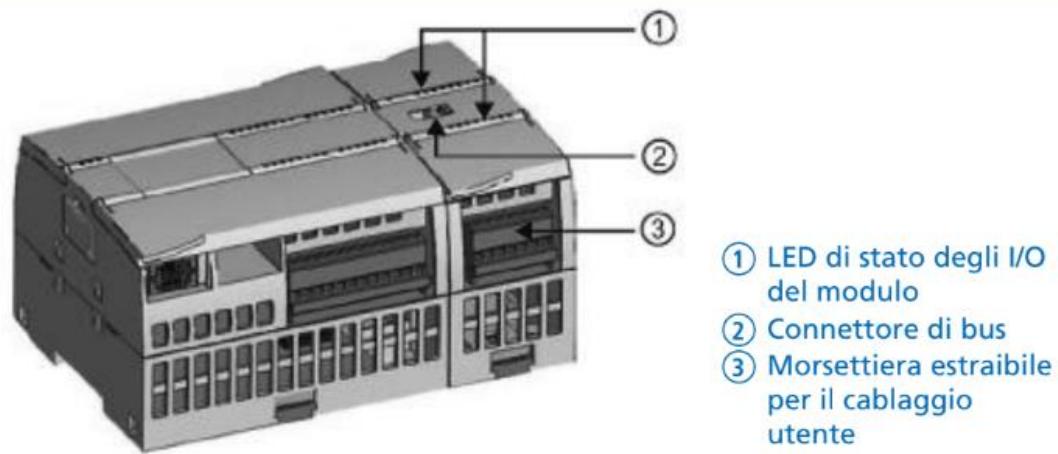
Le signal board (SB) consentono di aggiungere ingressi e uscite alla CPU. Le SB possono disporre di I/O digitali o analogici e vengono installate sul lato anteriore della CPU.

Permettono di ampliare la dotazione di I/O della CPU senza richiedere l'inserimento

di un modulo aggiuntivo, risultano utilissime quando viene richiesto un limitato incremento di I/O.

SIGNAL MODULE I/O:

La serie S7- 1200 comprende numerosi moduli di espansione Input/Output che consentono di ampliare le funzionalità della CPU.



A seguire una tabella che riporta I/O per Signal Board e Moduli di Espansione SM

Tipo	Solo ingresso	Solo uscita	Combinazione ingressi/uscite
3) SB digitale	4 ingressi 24V DC 200KHz 4 ingressi 5V DC 200KHz	4 uscite 24V DC 200KHz 4 uscite 5V DC 200KHz	2 ingressi 24V DC/ 2 uscite 24V DC 2 ingressi 5V DC/ 2 uscite 5V DC
4) SM digitale	8 ingressi 24V DC	8 uscite 24V DC 8 uscite relè 8 uscite relè in scambio	8 ingressi 24V DC/ 8 uscite 24V DC 8 ingressi 24V DC/ 8 uscite relè 8 ingressi 120/230V/8 uscite relè
	16 ingressi 24V DC	16 uscite 24V DC 16 uscite relè	16 ingressi 24VDC/16 uscite 24VDC 16 ingressi 24VDC/16 uscite relè
3) SB analogica	1 ingresso 1 RTD 1 termocoppia	1 uscita analogica	
4) SM analogico	8 ingressi 8 termocoppia 8 RTD	2 uscite analogiche 4 uscite analogiche	4 ingressi analogici/ 2 uscite analogiche

A seguire esempi tratti da catalogo SIEMENS di SB e SM.

<input type="checkbox"/> Codice Materiale / Descrizione	Prezzo di listino / Prezzo al cliente
<input type="checkbox"/>  6ES7221-1BF32-0XB0 SIMATIC S7-1200, unità di ingressi digitali SM 1221, 8 DI, DC 24V, Sink/Source	Mostra Prezzi
<input type="checkbox"/>  6ES7221-1BH32-0XB0 SIMATIC S7-1200, unità di ingressi digitali SM 1221, 16DI, DC 24V, Sink/Source	Mostra Prezzi
Pagina 1 di 1	
 < 1 > > 	
<input type="checkbox"/> Codice Materiale / Descrizione	Prezzo di listino / Prezzo al cliente
<input type="checkbox"/>  6ES7221-3AD30-0XB0 SIMATIC S7-1200, unità di ingressi digitali SB 1221, 4DI, DC 5V 200kHz, lettura M	Mostra Prezzi
<input type="checkbox"/>  6ES7221-3BD30-0XB0 SIMATIC S7-1200, unità di ingressi digitali SB 1221, 4DI, DC 24V 200kHz, lettura M	Mostra Prezzi
Pagina 1 di 1	
 < 1 > > 	
MODULO DI COMUNICAZIONE:	
<p>Sono inoltre disponibili moduli di comunicazione (CM), in grado di dialogare con altri sistemi su protocolli diversi.</p> <p>In particolare, Siemens utilizza il Profibus (Process Field Bus), un bus di campo costituito da un unico cavo che collega con un PROTOCOLLO SERIALE il PCL ad altri dispositivi.</p> <p>La CPU supporta fino ad un massimo di 3 moduli di comunicazione</p> <p>I moduli di comunicazione devono essere montati nel rack a sinistra della CPU, o di un altro modulo CM.</p>	
SIMATIC S7-1200	
<p>Morsettiera superiore:</p> <p>Primi due terminali (L+,M) sono l'alimentazione di linea (possiamo avere PLC a 24V in continua o 220V in alternata)</p>	

Terzo terminale: messa a terra delle masse
Quarto e quinto: possono alimentare i sensori di ingresso.

SOFTWARE

OB: Blocco organizzativo

TIA PORTAL: Totaly Integrted Automation Portal,

da parte di Siemens è stato creato uno software che unisce i prodotti Simatic che operano insieme.

Si riportano le principali funzionalità:

1. Configurazione Hardware
2. Impostazione della comunicazione tra PC e PLC
3. Programmazione
4. Test e messa in servizio
5. Creazione di documentazione
6. Creare interfacce grafiche “HMI” Human Machine Interface utilizzando il software integrato WinCC.

Si imparerà a programmare su “Tia Portal”. Una volta realizzato il programma si effettuerà il download sul PLC collegato tramite Cavo Ethernet.

Configurazione Hardware:

- “Crea nuovo progetto” → assegno il nome al progetto “Livello Serbatoio” e premere il pulsante Crea.
- Apri vista progetto

- Nella finestra navigazione di progetto doppio click su “Aggiungi nuovo dispositivo”
- Selezionare PLC Simatic S7-1200, e il modello nel nostro caso:
CPU1214C DC/DC/DC
- numero di ordinazione **6ES7 214-1AG31-0XB0**
confermare premendo il pulsante ok.
- click due volte nella riga PLC si aprono le cartelle ad esempio main [OB1] nel quale è possibili memorizzare il programma.

Scrittura del programma:

Cliccare su main [OB1] si aprirà la finestra di programmazione:
la prima è la barra dei comandi
la seconda una barra con i simboli del linguaggio ladder
e in mezzo una riga denominata RANG, sarà sufficiente trascinare i simboli del linguaggio ladder sopra la linea rang.

Una volta disegnati i vari segmenti di programma KOP, i contatti, le bobine appariranno con tanti punti di domanda, facciamo click e assegnamo un nome identificativo come da Schema Elettromeccanico.
Ora lo schema continuerà a evidenziare la variabile con un tratteggio sulla parte inferiore.
Facendo doppio cclick sulla singola

Associazione simboli sullo schema ladder (contatti, bobine) agli indirizzi di memoria Ingressi I, uscite Q, merker M.

Con il mouse ci si porta sopra il simbolo, click con il tasto destro del mouse e selezionare “Definire una variabile”. Nella finestra indicare l’indirizzo di memoria e cliccare su Definisci.

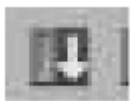
Salvataggio del programma

Compilazione del programma:



Download nel PC:

Nella finestra di Tia Portal selezionare:



Scelgo il tipo dell'interfaccia PN/IE e il tipo di scheda di rete alla quale è collegato il PLC,
avvia ricerca. E poi diamo LOAD.

Stop All e riscarico il software e Finish.

PER FARE LA SIMULAZIONE:

- Salva progetto, compila e poi “Avvia Simulazione”
si aprirà una interfaccia e poi carica il programme nel plc virtuale.



occhialini

Primo Esempio: Accensione di una lampada tramite un pulsante.

Secondo : sistema con autoritenuta

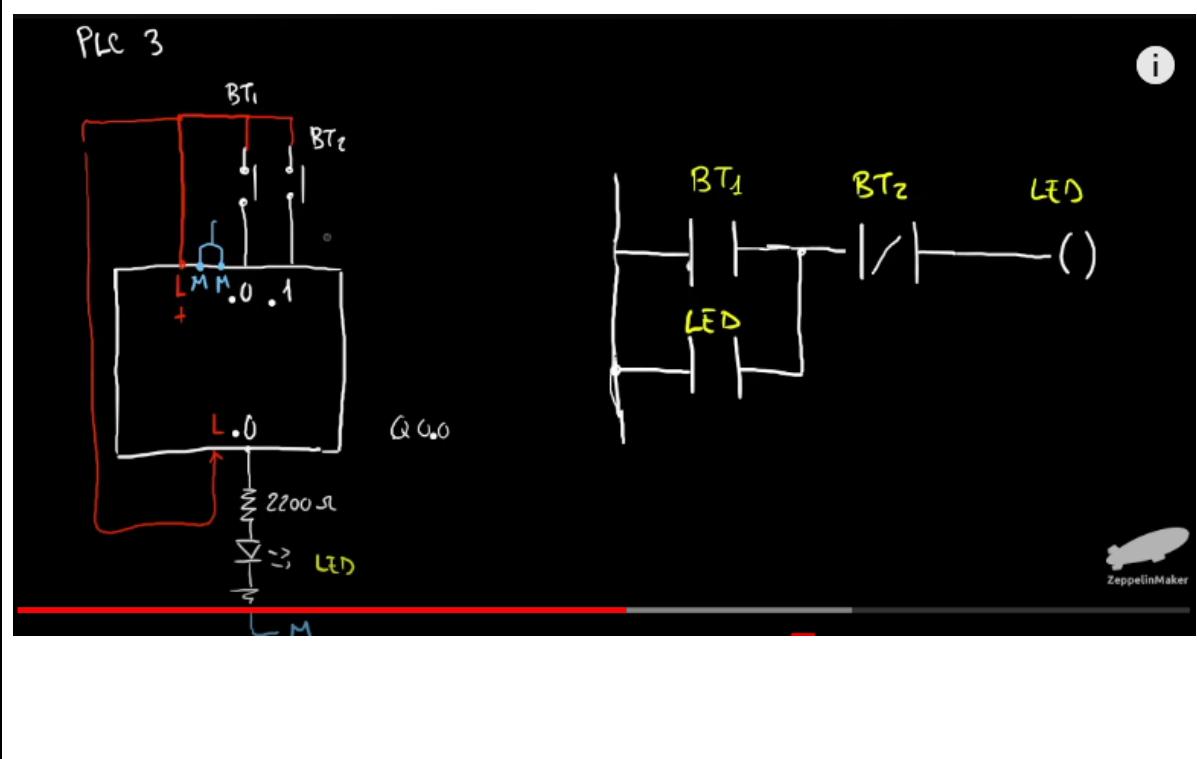
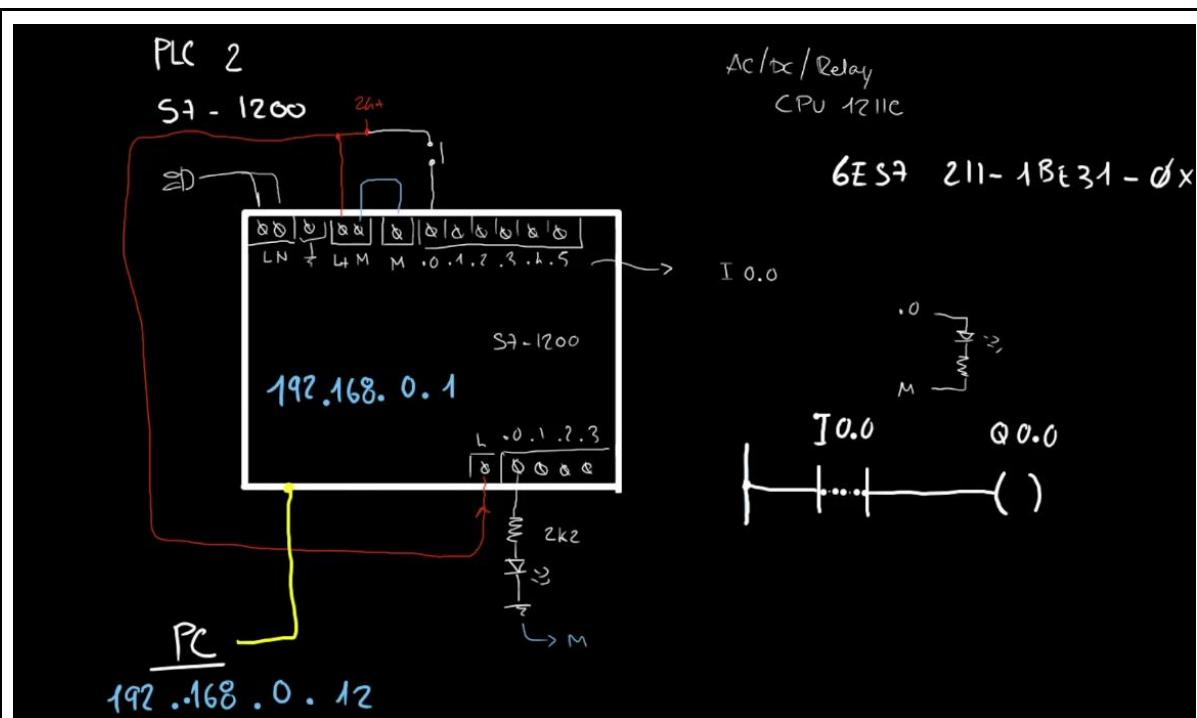
Terzo: uso del blocco SR Set e Reset

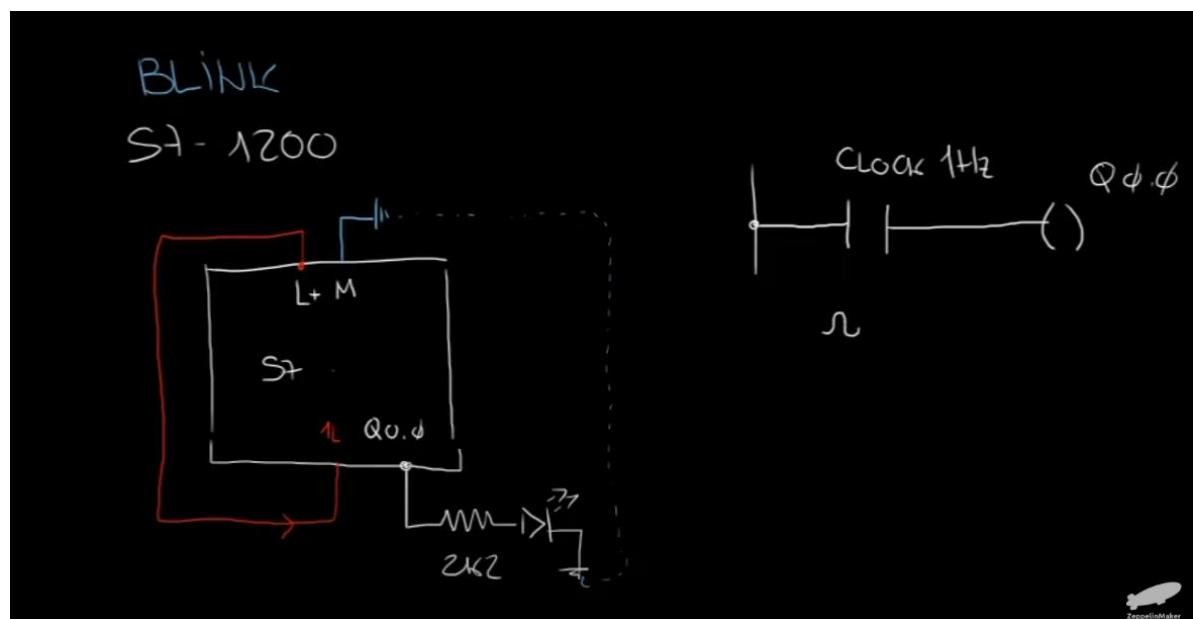
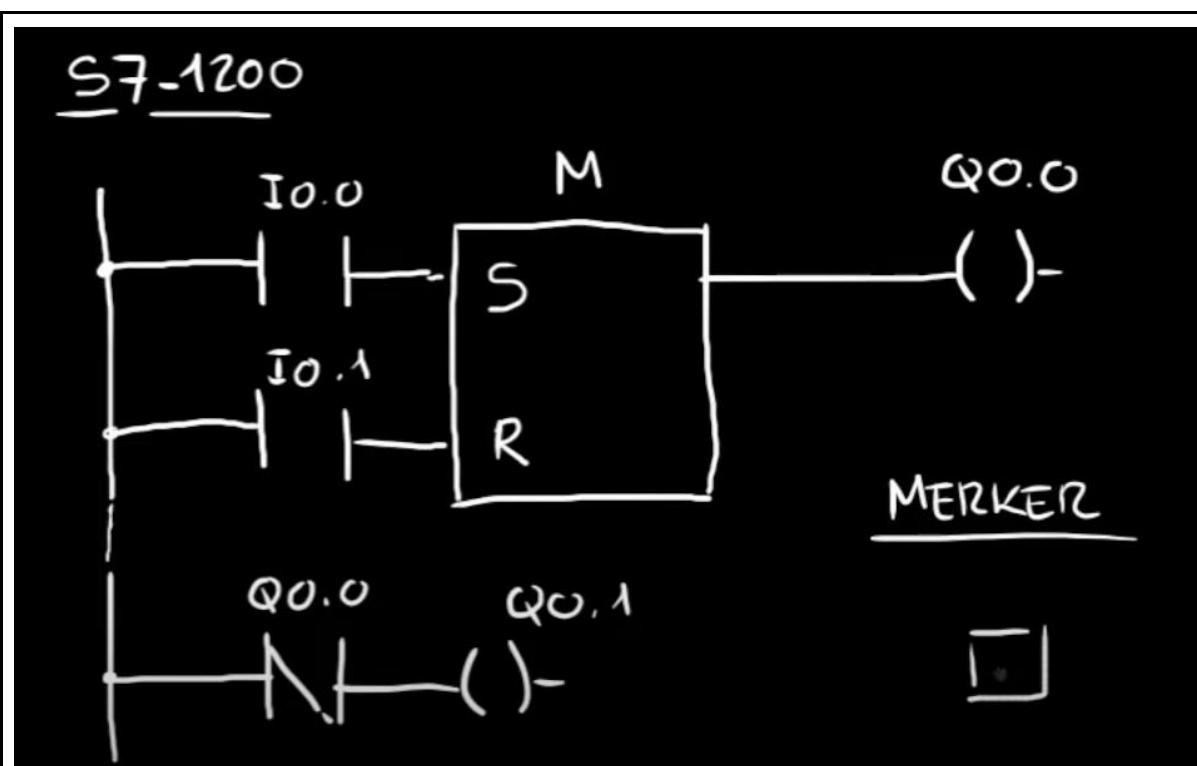
il blocco SR richiede la definizione di un marker dove memorizzare il bit

Quarto: Blink

Quinto: Timer

Sesto: Fronti di salita e di discesa dei segnali

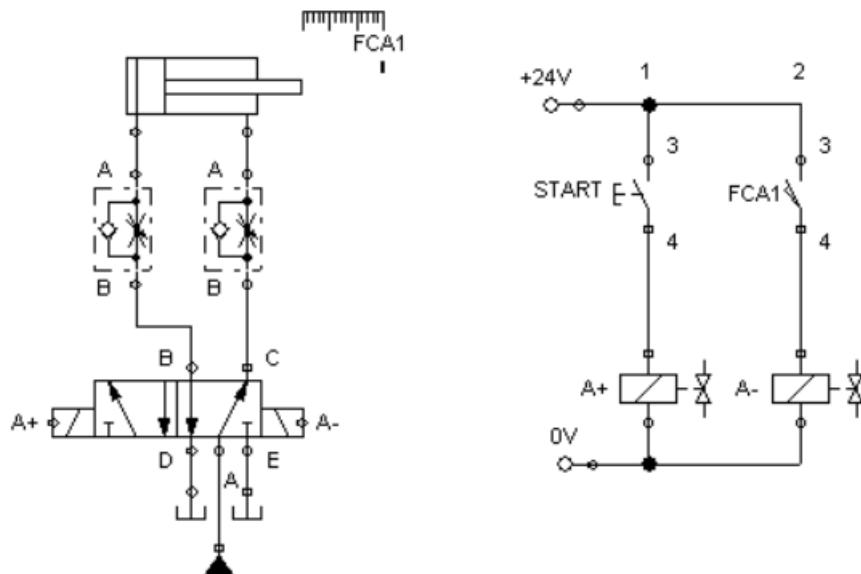




SCHEMA ELETTRICO:	
CENNI TEORICI:	<p><i>Il sensore HCSR04 è un sensore ad ultrasuoni che consente di misurare la distanza da un ostacolo.</i></p> <p><i>Il componente presenta quattro terminali, VCC e GND (per l'alimentazione), TRIGGER e ECHO.</i></p> <p><i>Al terminale TRIGGER invia un impulso di durata pari a 10 microsecondi, la linea ECHO riceve il segnale di ritorno. Il suono che infrangendosi sull'ostacolo ritorna al sensore. La durata dell'impulso di ritorno coincide con il tempo impiegato dal segnale per percorrere TRIGGER-OSTACOLO-ECHO.</i></p> <p><i>La funzione che ci consente di determinare il tempo è: Techo=pulseIn(pin,timeout); in microsecondi</i></p> <p><i>Il tempo OSTACOLO-ECHO vale metà del tempo rilevato.</i></p> <p><i>Lo spazio espresso in centimetri è dato da:</i></p> <p>$s=v*t = 343,8 \text{ m/s} * \text{Techo}(\text{microsec}) / 2 = 0,0172 * \text{Techo}$</p>

IDE:	<p><i>Per determinare la durata dell'impulso del segnale di ritorno in microsecondi:</i></p> <p><i>Techo=PulseIn(pin,timeout);</i></p> <p><i>pin è il pin di Arduino al quale è collegato il terminale Echo del sensore, timeout è il valore massimo di tempo dopo il quale la trasmissione si interrompe in quanto l'ostacolo è troppo lontano.</i></p> <p><i>Per attivare il buzzer utilizzo la funzione:</i></p> <p><i>Tone(pin,frequenza in Hz, durata in millisecondi);</i></p>
-------------	--

ARGOMENTO:	ELETTROPNEUMATICA - PLC
TITOLO:	CICLO SEMIAUTOMATICO DI UN CILINDRO IDRAULICO A DOPPIO EFFETTO CON VALVOLA 5/2 BISTABILE
DESCRIZIONE CICLO	Il ciclo semiautomatico consiste nell'estensione e rientro dello stelo quindi A+,A- a seguito del comando esercitato premendo il tasto di START.
MATERIALE:	<ul style="list-style-type: none"> - n°1 Cilindro a doppio effetto - n°1 Elettrovalvola bistabile 5/2 - n°2 valvole regolatrici di portata unidirezionali collegati allo scarico del cilindro - n°1 in finecorsa FCA1 (stelo esteso) - n°1 pulsante monostabile di START - conduttori elettrici per i collegamenti - tubi pneumatici per i collegamenti

SCHEMA ELETTROPNEUMATICO:**PRINCIPIO DI FUNZIONAMENTO:**

La pressione del tasto START attiva la bobina A+ (fuoriuscita dello stelo), essendo l'elettrovalvola del tipo bistabile questa si mantiene in tale configurazione fino a quando non viene sollecitata la bobina A-. Lo stelo completamente esteso attiva il finecorsa FCA1, che a sua volta attiva A- facendo rientrare lo stelo.

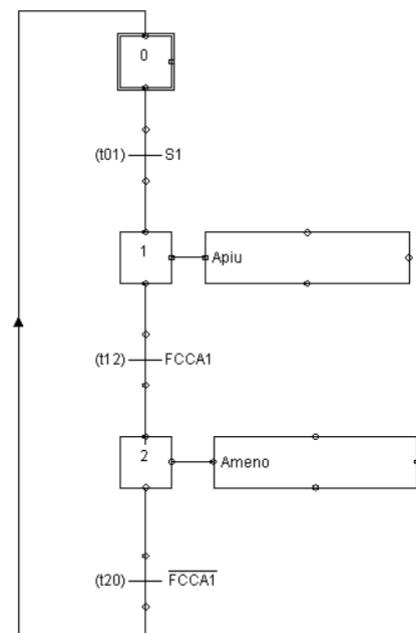
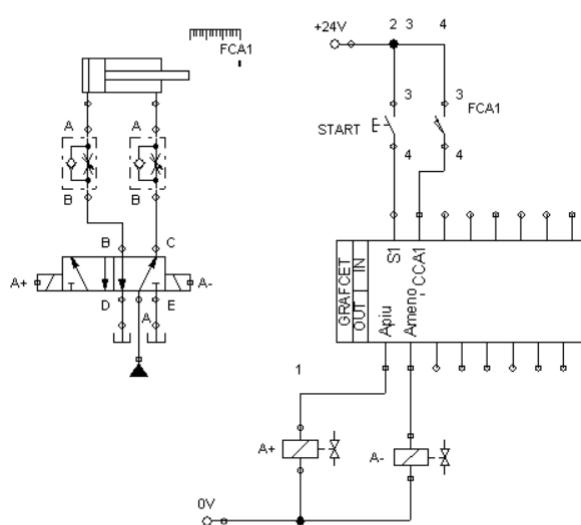
TEORIA:

Cilindro pneumatico a doppio effetto:

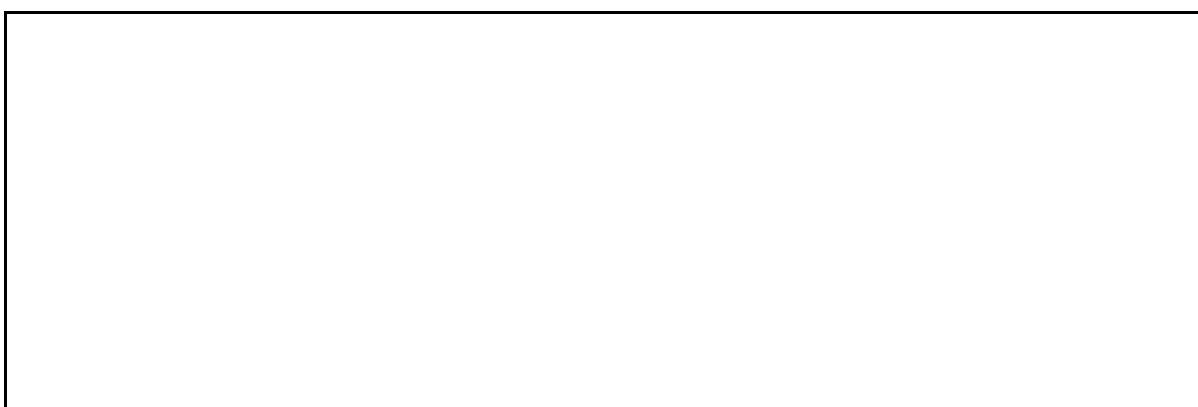
Elettrovalvola:

Valvola regolatrice di portata unidirezionale:

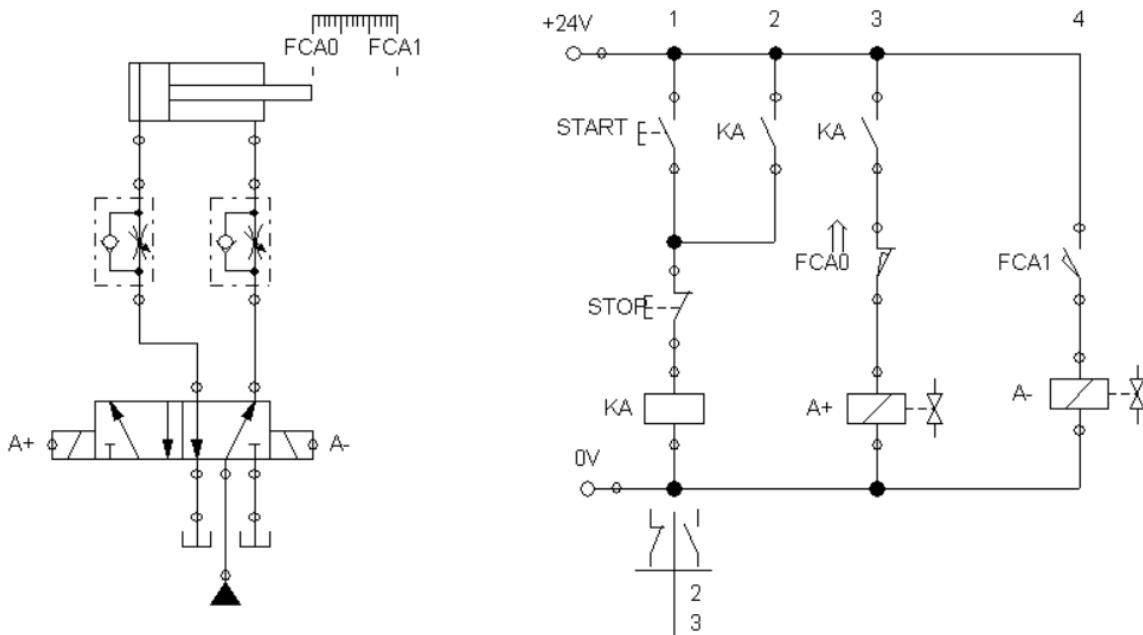
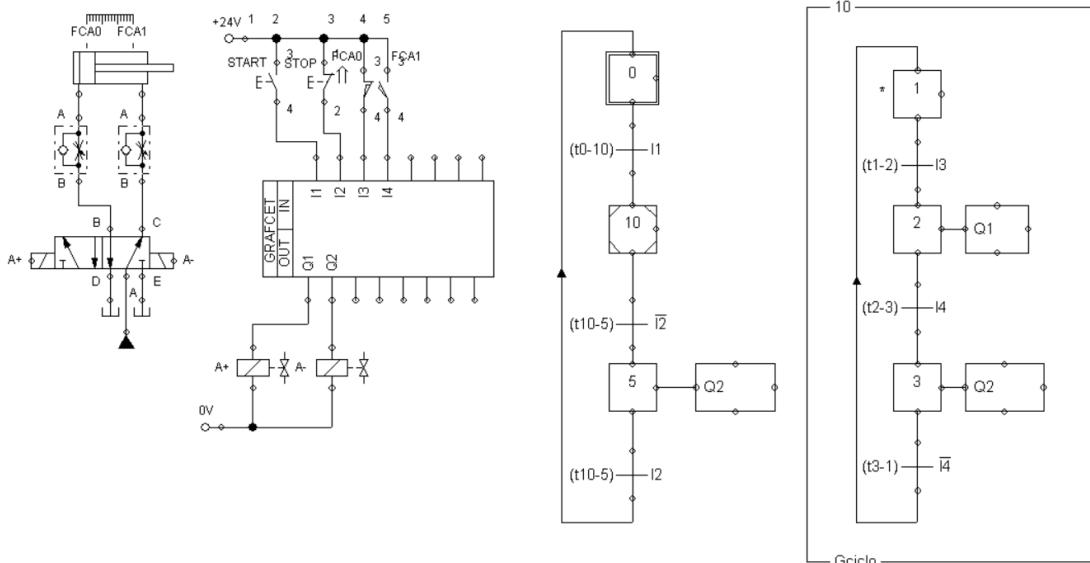
Principio di funzionamento:

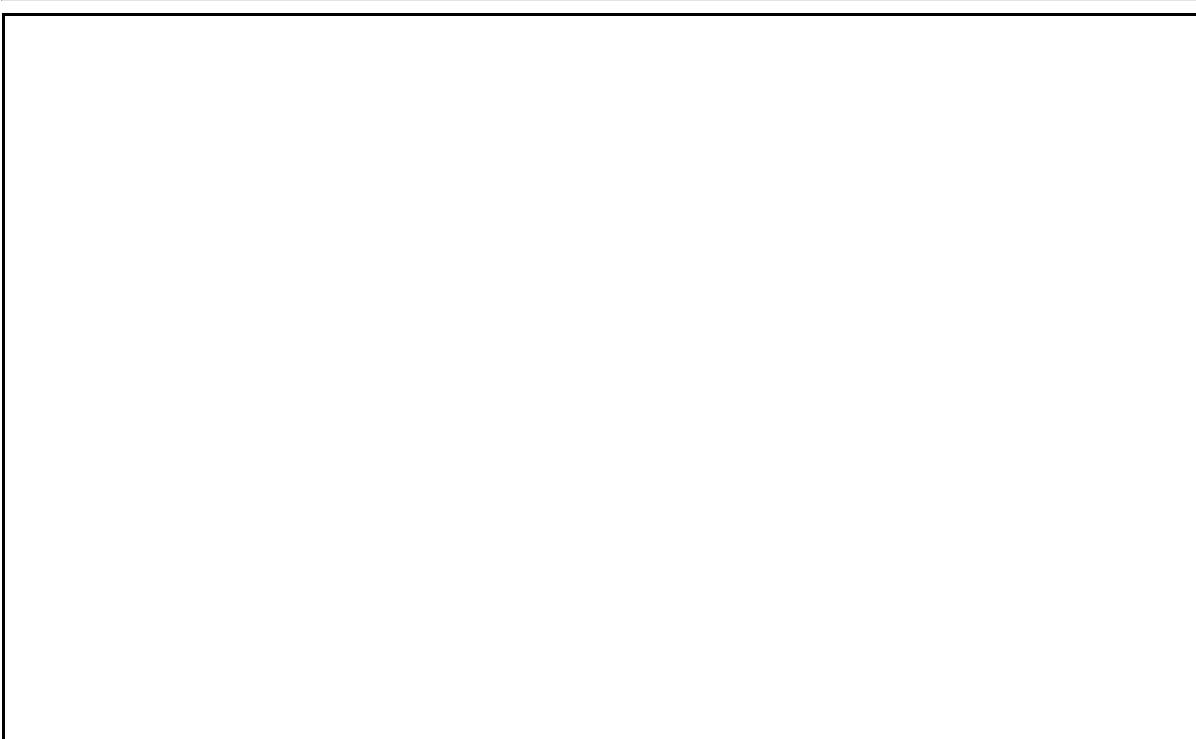
SCHEMA IN GRAFCET:

PLC



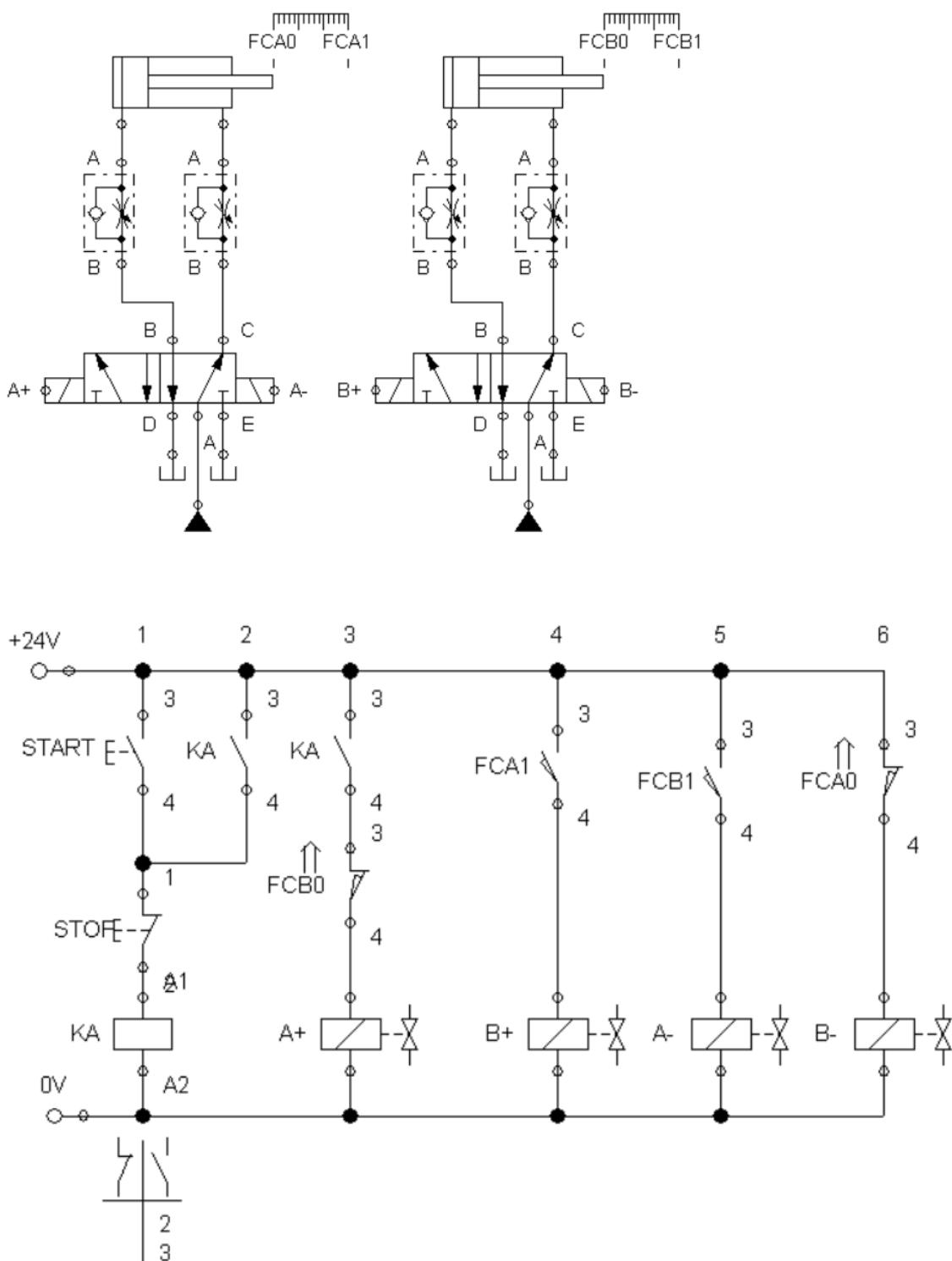
ARGOMENTO:	ELETTROPNEUMATICA - PLC
TITOLO:	CICLO AUTOMATICO DI UN CILINDRO IDRAULICO A DOPPIO EFFETTO CON VALVOLA 5/2 BISTABILE
MATERIALE:	<ul style="list-style-type: none">- n°1 Cilindro a doppio effetto- n°1 Elettrovalvola bistabile 5/2- n°2 valvole regolatrici di portata unidirezionali collegati allo scarico del cilindro- n°1 in finecorsa FCA1 (stelo esteso) e n°1 finecorsa FCA0 (stelo rientrato)- n°1 pulsante monostabile di START di inizio ciclo- n°1 pulsanti di STOP di fine ciclo- n°1 relè di autoritenuta- conduttori elettrici per i collegamenti- tubi pneumatici per i collegamenti
TEORIA:	
SCHEMA ELETTROPNEUMATICO:	

**SCHEMA IN GRAFCET**



ARGOMENTO:	ELETTROPNEUMATICA - PLC
TITOLO:	CICLO AUTOMATICO DI DUE CILINDRI IDRAULICI A DOPPIO EFFETTO CON VALVOLA 5/2 BISTABILE
MATERIALE:	<ul style="list-style-type: none">- n°2 Cilindro a doppio effetto- n°2 Elettrovalvola bistabile 5/2- n°4 valvole regolatrici di portata unidirezionali collegati allo scarico del cilindro

	<ul style="list-style-type: none">- n°1 in finecorsa FCA1 (stelo esteso) e n°1 finecorsa FCA0 (stelo rientrato)- n°1 in finecorsa FCB1 (stelo esteso) e n°1 finecorsa FCB0 (stelo rientrato)- n°1 pulsante monostabile di START di inizio ciclo- n°1 pulsanti di STOP di fine ciclo- n°1 relè di autoritenuta- conduttori elettrici per i collegamenti- tubi pneumatici per i collegamenti
SCHEMA ELETTROPNEUMATICO:	

**SCHEMA GRAFCET A+ B+ A- B-**



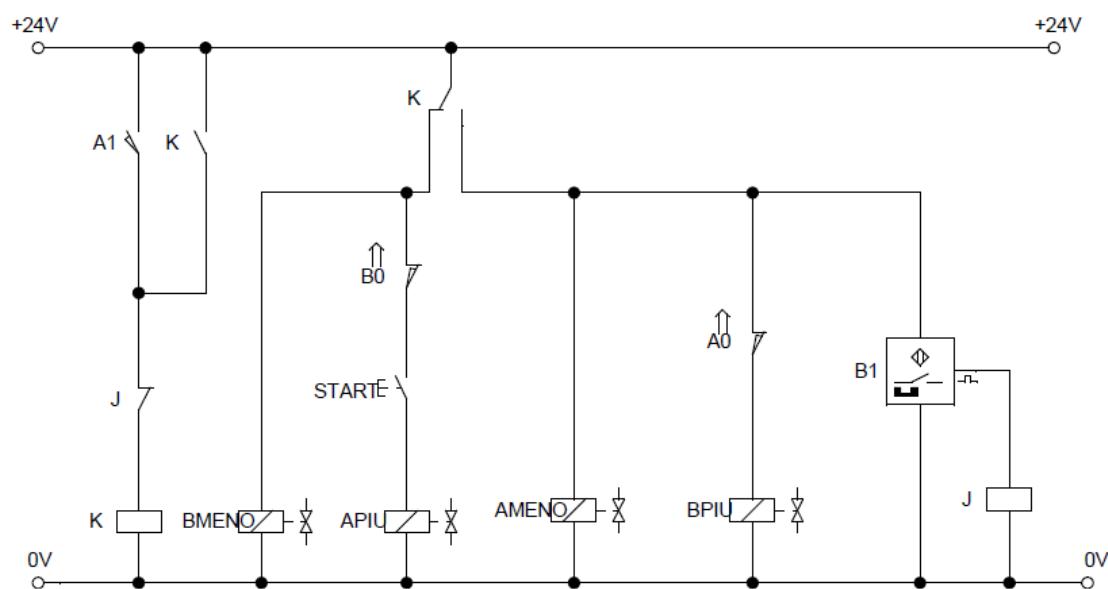
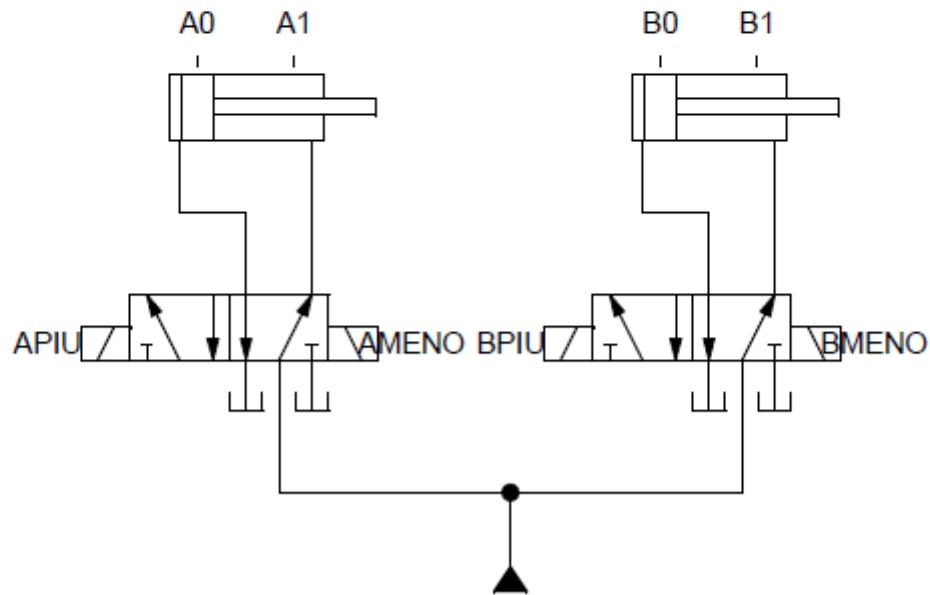
Movimento A+/A-/B+/B-

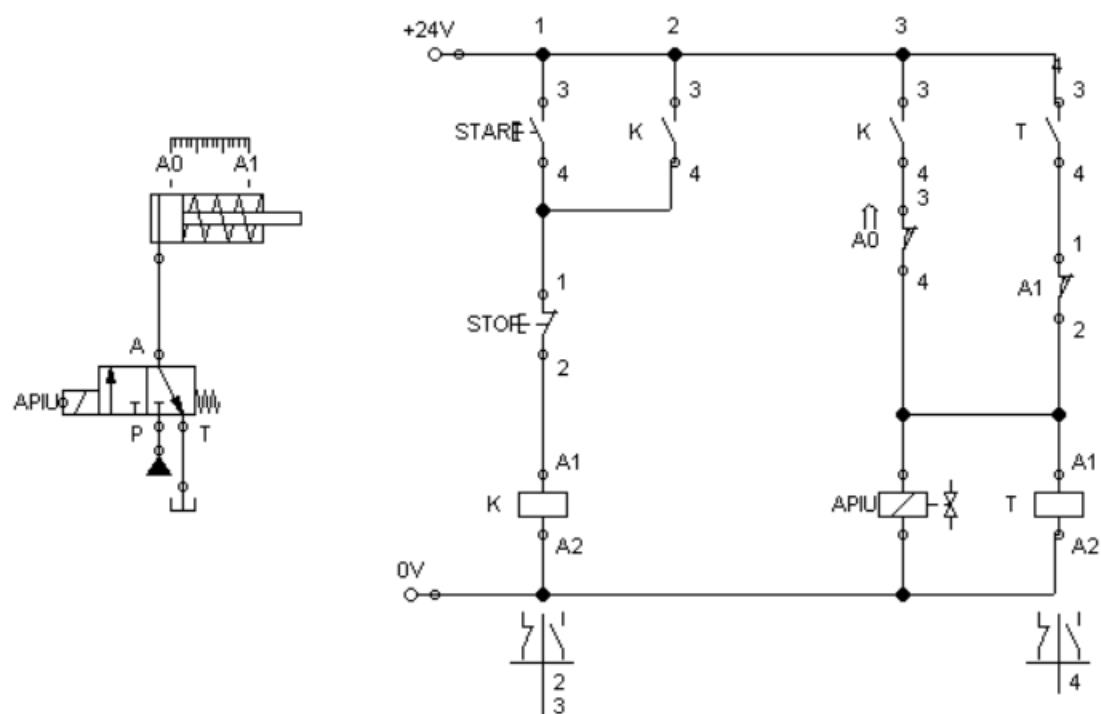
1° Gruppo B-A+

2° Gruppo A-/B+

Il secondo gruppo è attivato da a1 (inserire contatto n.o)

Il secondo gruppo è disattivato da b1 (inserire contatto n.c)



Ciclo A+/A- automatico**Cilindro a singolo effetto con ritorno a molla****Elettrovalvola monostabile 3/2**

PROGRAMMAZIONE IN C

- EDITING E COMPILAZIONE (debug+link)
- STRUTTURA DEL PROGRAMMA IN C
- FUNZIONE DI OUTPUT VISUALIZZAZIONE A MONITOR
- TIPI DI DATO INIZIAMO DAGLI INTERI
- FUNZIONE DI INPUT "ACQUISIZIONE DATO DIGITATO DA TASTIERA"

ALGORITMO

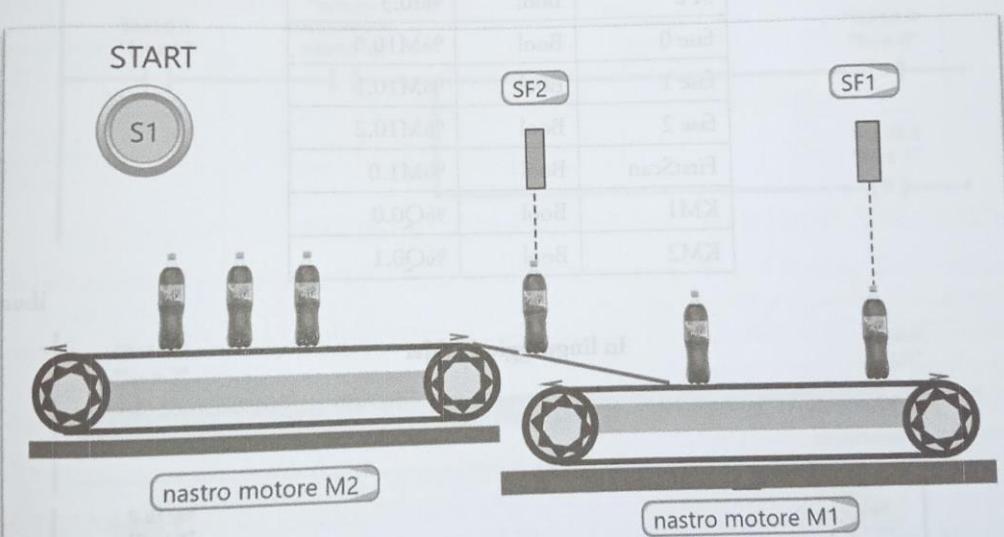
L'algoritmo è un insieme di passi semplici finalizzati alla risoluzione di un problema, nello specifico la soluzione di un processo di automazione industriale.

ESEMPIO DI UN PROBLEMA DI AUTOMAZIONE

ESERCIZIO N.4 LINEA PER IL CONFEZIONAMENTO AUTOMATICO DI PRODOTTI

Un nastro trasportatore, comandato dal motore M2, trasferisce delle bottiglie verso la stazione di confezionamento. Un contatore, attraverso un sensore (fotocellula SF2), rivela il passaggio di almeno 12 prodotti e attiva un secondo nastro trasportatore comandato dal motore M1. Durante il trasferimento della confezione di bottiglie alla stazione di stoccaggio, il nastro azionato dal motore M2 si ferma e riprende a funzionare solo dopo che sono trascorsi 5 s da quando sono state trasportate le 12 bottiglie nella stazione di ricezione, rilevate dalla fotocellula SF1, arrestando M1 e facendo riprendere nuovamente il ciclo.

Il processo si arresta entro 100 s, che è il tempo massimo per il passaggio di 12 bottiglie.



- studio del linguaggio di programmazione in C applicato ai microcontrollori (PIC, ATMEGA ecc...)
- studio del linguaggio ladder applicato al PLC (Controllore Logico Programmabile)

EDITING E COMPILAZIONE

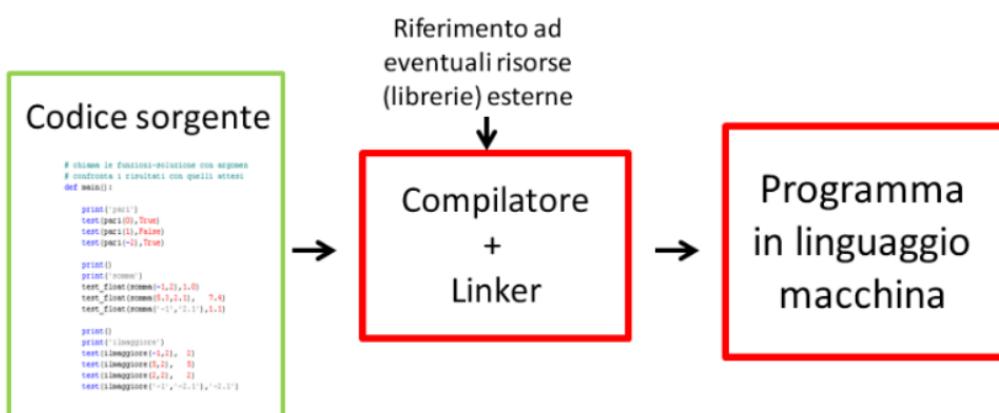
FASE DI EDITING DEL FILE SORGENTE:

- scrittura del programma. Il risultato di questa operazione sarà il salvataggio di un file avente estensione .c

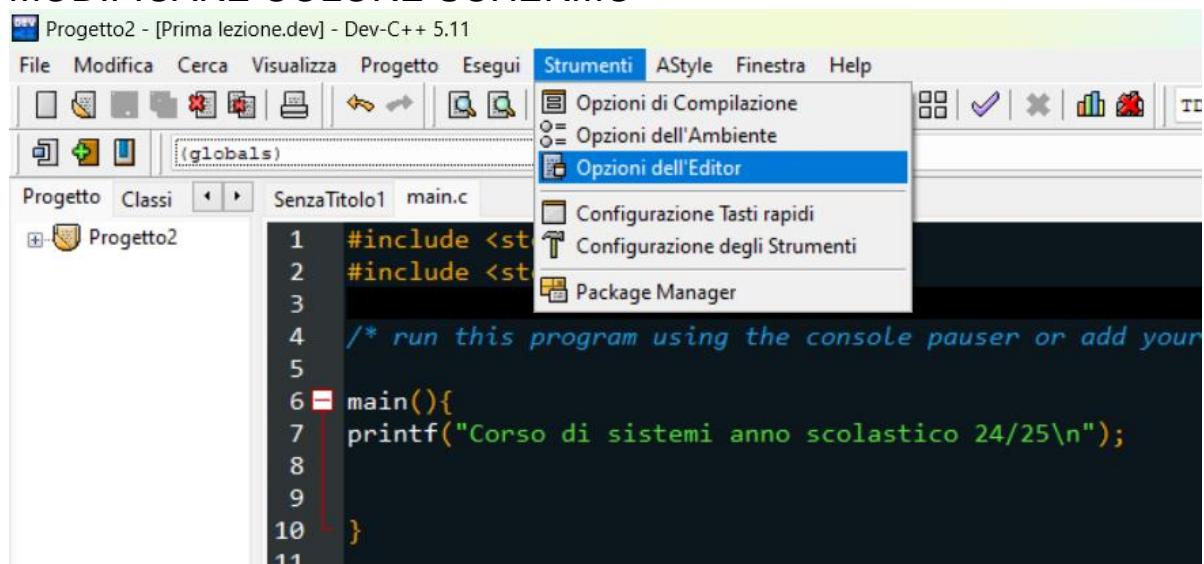
FASE DI COMPILAZIONE:

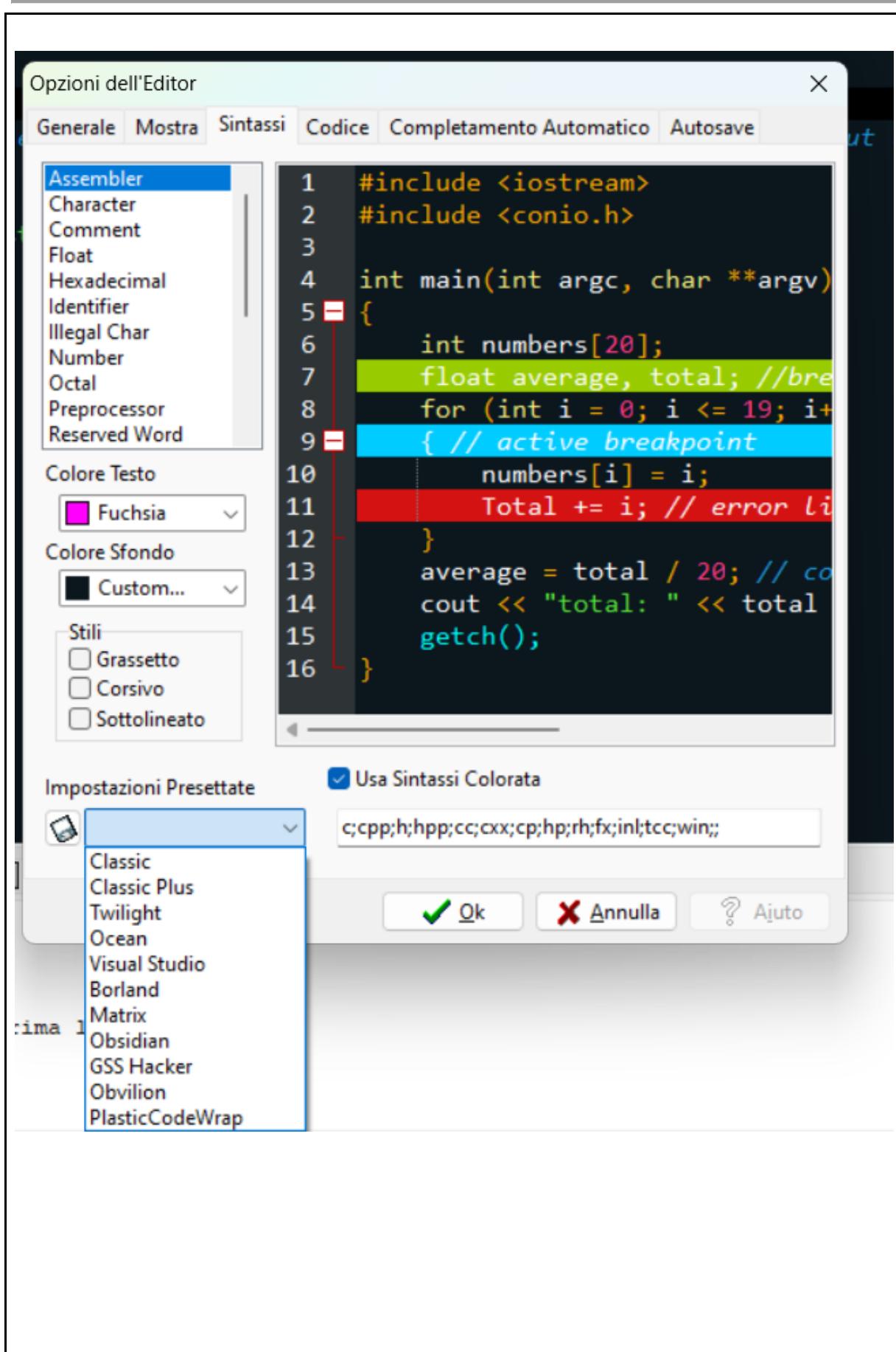
- debug: verifica eventuali errori di sintassi (regole di scrittura)
- link: collega il codice scritto alle librerie (file header) dichiarate nell'editor.

Le librerie possono essere presenti nel compilatore e librerie scritte dall'utente. Il risultato è un file in linguaggio macchina eseguibile di estensione .exe



MODIFICARE COLORE SCHERMO





STRUTTURA DEL PROGRAMMA IN C

Il programma in C deve contenere nell'ordine:

1) richiami alle librerie (file esterni) tramite la direttiva

```
#include < nome della libreria.h>
ad esempio
#include<stdio.h> //se il file è presente nella directory del compilatore
#include"miafunzione.h" //se il file è scritto dall'utente e salvato nella cartella
del progetto.
```

2) Funzione principale: “main()” { }

entro le parentesi

- dichiarazione del tipo di dato seguite dal punto e virgola,
- le istruzioni separate dal punto e virgola

```
main(){
dichiarazione del tipo di dato;
prima istruzione;
seconda istruzione;
terza istruzione;
ennesima istruzione;
system("pause");}
```

3) commenti: non vengono eseguiti servono a rendere esplicito ciò che viene scritto nell'editor.

COMMENTI:

Due modi per inserire i commenti:

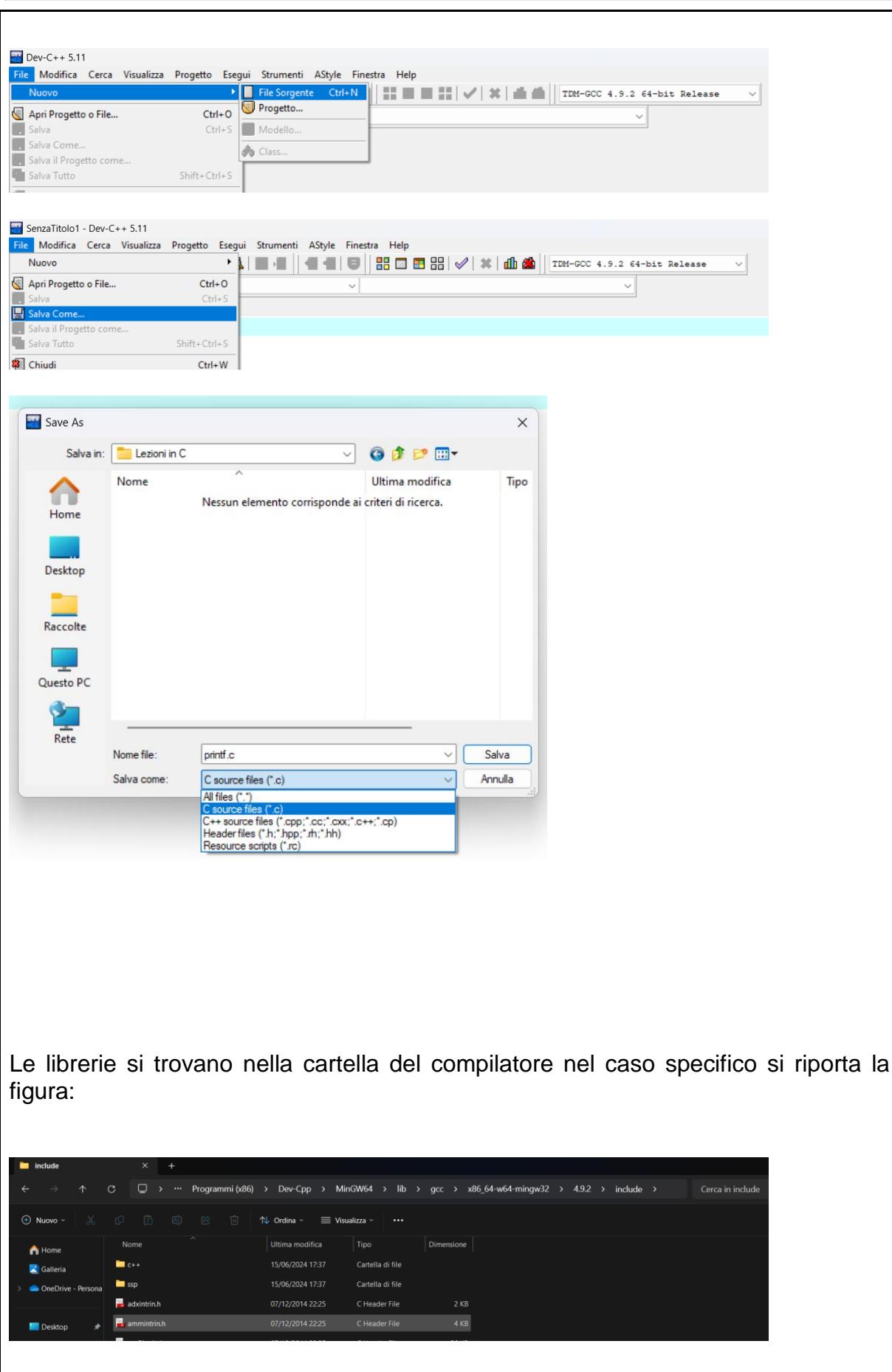
```
// commento singola riga
in alternativa
```

```
/*commento scritto
su
più righe
*/
```

FUNZIONE DI OUTPUT printf():

La funzione printf() serve per visualizzare a monitor una stringa di caratteri e valori di variabili.

Per poter essere impiegata dobbiamo da subito indicare al compilatore l'utilizzo della libreria stdio tramite la direttiva #include<stdio.h>.



Le librerie si trovano nella cartella del compilatore nel caso specifico si riporta la figura:

```
1  /* programma che stampa a monitor la scritta
2  "Corso di Sistemi automatici anno 2024/25"
3  */
4
5  //direttiva che indica l'utilizzo della libreria <stdio.h>
6  #include<stdio.h>
7  //scrittura della funzione principale
8  main(){
9      //non sono presenti dati da dichiarare
10     //Scrittura a monitor della stringa di caratteri tra doppi apici ""
11     printf("Corso di Sistemi automatici anno 2024/25");
12 }
```

Compilazione → F9

Esecuzione → F10

Compilazione + Esecuzione → F11

A monitor vedremo la scritta:

The screenshot shows a terminal window titled 'E:\Lezioni in C\printf.exe'. The window displays the text 'Corso di Sistemi automatici anno 2024/25' followed by a horizontal line and the message 'Process exited after 0.07327 seconds with return value 40'. Below this, there is a prompt 'Premere un tasto per continuare . . . |'.

Entriamo nella cartella dove è presente il file printf.c e vediamo che è stato creato un file eseguibile print.exe. Lanciamo l'esecuzione del file, constatiamo che la velocità di esecuzione è talmente grande che non ci permette di visualizzare a monitor il messaggio, quindi inseriamo una funzione che indica al software di sospendere l'esecuzione al fine di visualizzare la stringa di caratteri.

```
system("pause");
```

```
1  /* programma che stampa a monitor la scritta
2  "Corso di Sistemi automatici anno 2024/25"
3  */
4
5  //direttiva che indica l'utilizzo della libreria <stdio.h>
6  #include<stdio.h>
7  //scrittura della funzione principale
8  main(){
9      //non sono presenti dati da dichiarare
10     //Scrittura a monitor della stringa di caratteri tra doppi apici ""
11     printf("Corso di Sistemi automatici anno 2024/25");
12     system("pause");
13 }
14
```

Per andare a capo utilizzare \n (newline).

```
1 /* programma che stampa a monitor la scritta
2 "Corso di Sistemi automatici anno 2024/25"
3 */
4
5 //direttiva che indica l'utilizzo della libreria <stdio.h>
6 #include<stdio.h>
7 //scrittura della funzione principale
8 main(){
9     //non sono presenti dati da dichiarare
10    //Scrittura a monitor della stringa di caratteri tra doppi apici ""
11    printf("Corso di Sistemi automatici anno 2024/25");
12    //andare a capo con la stringa successiva
13    printf("\n");
14    system("pause");
15 }
```

in alternativa:

```
1 /* programma che stampa a monitor la scritta
2 "Corso di Sistemi automatici anno 2024/25"
3 */
4
5 //direttiva che indica l'utilizzo della libreria <stdio.h>
6 #include<stdio.h>
7 //scrittura della funzione principale
8 main(){
9     //non sono presenti dati da dichiarare
10    //Scrittura a monitor della stringa di caratteri tra doppi apici ""
11    printf("Corso di Sistemi automatici anno 2024/25\n");
12    system("pause");
13 }
14 |
```

Esercizio provare a ottenere il seguente risultato:

Corso di sistemi automatici anno scolastico 2024/25

```
##### # #####  
# # # # #  
# # # # #  
# # # # #  
##### # #####  
# # # # #  
# # # # #  
# # # # #  
##### ##### # #
```

Premere un tasto per continuare . . . |

CODICE ASCII (AMERICAN STANDARD CODE FOR INFORMATION INTERCHANGE)

Binary	Oct	Dec	Hex	Glyph
010 0000	040	32	20	
010 0001	041	33	21	!
010 0010	042	34	22	"
010 0011	043	35	23	#
010 0100	044	36	24	\$
010 0101	045	37	25	%
010 0110	046	38	26	&
010 0111	047	39	27	'
010 1000	050	40	28	(
010 1001	051	41	29)
010 1010	052	42	2A	*
010 1011	053	43	2B	+
010 1100	054	44	2C	,
010 1101	055	45	2D	-
010 1110	056	46	2E	.
010 1111	057	47	2F	/
011 0000	060	48	30	0
011 0001	061	49	31	1
011 0010	062	50	32	2
011 0011	063	51	33	3
011 0100	064	52	34	4
011 0101	065	53	35	5
011 0110	066	54	36	6
011 0111	067	55	37	7
011 1000	070	56	38	8
011 1001	071	57	39	9
011 1010	072	58	3A	:
011 1011	073	59	3B	;
011 1100	074	60	3C	<
011 1101	075	61	3D	=
011 1110	076	62	3E	>
011 1111	077	63	3F	?

Binary	Oct	Dec	Hex	Glyph
100 0000	100	64	40	@
100 0001	101	65	41	A
100 0010	102	66	42	B
100 0011	103	67	43	C
100 0100	104	68	44	D
100 0101	105	69	45	E
100 0110	106	70	46	F
100 0111	107	71	47	G
100 1000	110	72	48	H
100 1001	111	73	49	I
100 1010	112	74	4A	J
100 1011	113	75	4B	K
100 1100	114	76	4C	L
100 1101	115	77	4D	M
100 1110	116	78	4E	N
100 1111	117	79	4F	O
101 0000	120	80	50	P
101 0001	121	81	51	Q
101 0010	122	82	52	R
101 0011	123	83	53	S
101 0100	124	84	54	T
101 0101	125	85	55	U
101 0110	126	86	56	V
101 0111	127	87	57	W
101 1000	130	88	58	X
101 1001	131	89	59	Y
101 1010	132	90	5A	Z
101 1011	133	91	5B	[
101 1100	134	92	5C	\
101 1101	135	93	5D]
101 1110	136	94	5E	^
101 1111	137	95	5F	_

Binary	Oct	Dec	Hex	Glyph
110 0000	140	96	60	`
110 0001	141	97	61	a
110 0010	142	98	62	b
110 0011	143	99	63	c
110 0100	144	100	64	d
110 0101	145	101	65	e
110 0110	146	102	66	f
110 0111	147	103	67	g
110 1000	150	104	68	h
110 1001	151	105	69	i
110 1010	152	106	6A	j
110 1011	153	107	6B	k
110 1100	154	108	6C	l
110 1101	155	109	6D	m
110 1110	156	110	6E	n
110 1111	157	111	6F	o
111 0000	160	112	70	p
111 0001	161	113	71	q
111 0010	162	114	72	r
111 0011	163	115	73	s
111 0100	164	116	74	t
111 0101	165	117	75	u
111 0110	166	118	76	v
111 0111	167	119	77	w
111 1000	170	120	78	x
111 1001	171	121	79	y
111 1010	172	122	7A	z
111 1011	173	123	7B	{
111 1100	174	124	7C	
111 1101	175	125	7D	}
111 1110	176	126	7E	~

```
a='@';
a=0b01000000;
a=0x40;
a=64;
printf("il carattere vale c%",a);
```

TIPI DI DATO INIZIAMO DAGLI INTERI:

La memoria è organizzata in bit che possono assumere valori binari 0 o 1. Quando si deve memorizzare il valore di un numero intero cioè privo di virgola si utilizza il tipo di dato int.

L'istruzione per dichiarare che il dato è intero è la seguente:

int xxx;

dove xxx è l'*identificatore*

Per identificatore si intende il nome del dato.

L'identificatore deve iniziare con lettere mai numeri, non deve coincidere con parole utilizzate nel linguaggio di programmazione, non sono permessi spazi e punti. Maiuscolo e minuscolo sono differenti.

Il valore massimo memorizzabile dipende dal numero dei bit attribuiti in memoria.

Per ottenere il valore intero si sommano le potenze di base due elevate per il corrispondente peso.

Ad esempio se si hanno a disposizione 16bit per memorizzare un dato intero si ottiene il valore massimo ponendo a 1 il valore del singolo bit e moltiplicando per le potenze in base 2 elevate per il corrispondente peso del bit.

$$\begin{aligned} \text{n}^{\circ} \text{massimo} = & 1 * 2^{15} + 1 * 2^{14} + 1 * 2^{13} + 1 * 2^{12} + 1 * 2^{11} + 1 * 2^{10} + 1 * 2^9 \\ & + 1 * 2^8 + 1 * 2^7 + 1 * 2^6 + 1 * 2^5 + 1 * 2^4 + 1 * 2^3 + 1 * 2^2 + 1 * 2^1 + 1 * 2^0 = 65535 \end{aligned}$$

	MSB																MSB
indice bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
massimo: bit tutti a 1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	
	32768	16384	8192	4096	2048	1024	512	256	128	64	32	16	8	4	2	1	

Per un numero generico di bit abbiamo:

numero combinazioni massime 2^n bit

valore minimo = 0

valore massimo = (2^n) - 1

Esercizio: calcolare il numero di combinazioni, valore minimo e massimo avendo a disposizione 32bit per memorizzare un numero intero positivo.

Valori interi con segno:

Possiamo utilizzare i bit definiti dal compilatore per memorizzare un valore intero con segno.

numero combinazioni= 2^n bit

valore massimo positivo= $((2^n) / 2) - 1$

valore minimo negativo= $-((2^n) / 2)$

	MSB															LSB	
indice bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
numero positivo (bit 15 =0)	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	
	0	16384	8192	4096	2048	1024	512	256	128	64	32	16	8	4	2	1	
indice bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
numero negativo(bit 15=1)	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
	-32768	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-32768	

Esercizio: calcolare il numero di combinazioni, valore minimo e massimo avendo a disposizione 32bit per memorizzare un numero intero con segno.

!!! attenzione al compilatore ad esempio:

TDM-GCC 4.9.2 destina 32 bit ai valori interi con segno cioè dichiarando:

int valore;

il dato valore potrà essere compreso tra $((2^{32})/2)-1$ e $-((2^{32})/2)$

```

1 //Programma che stampa a monitor il valore di una variabile
2
3 #include<stdio.h>      //direttiva libreria stdio
4
5 main(){
6     int NUM;           //dichiarazione della variabile intera NUM
7     NUM=25;           //assegnazione del valore intero alla variabile
8     // potevo con un'unica riga dichiarare e assegnare con l'istruzione
9     // int NUM=25 si tratta di inizializzazione
10    printf("%d",NUM);
11    system("pause");
12 }
```

E:\Lezioni in C\printfvariabili × + ▾

25Premere un tasto per continuare . . .

Osservazioni:

- mandare a capo “Premere un tasto per continuare” usando il carattere \n
- Proviamo a inserire una scritta che specifica “Il valore assegnato a NUM vale “

```

1 //Programma che stampa a monitor il valore di una variabile
2
3 #include<stdio.h>      //direttiva libreria stdio
4
5 main(){
6     int NUM;           //dichiarazione della variabile intera NUM
7     NUM=25;           //assegnazione del valore intero alla variabile
8     // potevo con un'unica riga dichiarare e assegnare con l'istruzione
9     // int NUM=25 si tratta di inizializzazione
10    printf("Il valore assegnato a NUM vale %d\n",NUM);
11    // %d è uno specificatore di formato, indica il formato della variabile
12    // e la posizione di NUM da occupare internamente alla stringa.
13    system("pause");
14 }

```

E:\Lezioni in C\printfvariabili < X + ▾

Il valore assegnato a NUM vale 25
Premere un tasto per continuare . . .

Esercizio: realizzare un programma che visualizzi a schermo la scritta:
La classe terza è formata da 20 studenti di età media 16 anni.
Consiglio: dichiarare e assegnare due variabili STUDENTI e ETA'.

```

1 //Programma che stampa a monitor il valore di una variabile
2
3 #include<stdio.h>      //direttiva libreria stdio
4
5 main(){
6     int STUDENTI,ETA;    //dichiarazione delle variabili STUDENTI ed ETA
7     STUDENTI=20;         //assegnazione del valore intero alla variabile STUDENTE
8     ETA=16;             //assegnazione del valore intero alla variabile ETA
9     printf("\nLa classe terza e' formata da %d studenti di eta' media %d anni\n\n",STUDENTI,ETA);
10    system("pause");
11 }
12

```

E:\Lezioni in C\printfvariabili < X + ▾

La classe terza e' formata da 20 studenti di eta' media 16 anni
Premere un tasto per continuare . . .

Modificare l'istruzione nel seguente modo:

```
printf("\nLa classe terza e' formata da %8d studenti di eta' media %5d anni\n\n",STUDENTI,ETA);
```

cosa succede?

```

1 //Programma che stampa a monitor il valore di una variabile
2
3 #include<stdio.h>      //direttiva libreria stdio
4
5 main(){
6     int STUDENTI,ETA;      //dichiarazione delle variabili STUDENTI ed ETA
7     STUDENTI=20;           //assegnazione del valore intero alla variabile STUDENTE
8     ETA=16;                //assegnazione del valore intero alla variabile ETA
9     printf("\nLa classe terza e' formata da %8d studenti di eta' media %5d anni\n\n",STUDENTI,ETA);
10    system("pause");
11 }
12

```

```

E:\Lezioni in C\printfvariabili  × + ▾

La classe terza e' formata da      20 studenti di eta' media      16 anni
Premere un tasto per continuare . . .

```

Il numero 20 inizialmente è giustificato a destra rispetto allo spazio che lo precede, inserendo %8d, si ha lo spostamento di 8 campi verso destra.

Tabella degli speciatori di Formato:

%d	variabile intera (int)
%f	variabile reale (float)
%c	variabile carattere
%s	variabile stringa di caratteri

Esercizio:

Stampare a monitor il messaggio “6 diviso 2 da’ risultato 3”

```

1 //Stampare a monitor il messaggio "6 diviso 2 da' risultato 3"
2
3 #include<stdio.h>      //direttiva libreria stdio
4
5 main(){
6     int NUM1,NUM2,RISULTATO;          //dichiarazione delle variabili intere NUM1,NUM2,RISULTATO
7     NUM1=6;NUM2=3;                  //assegnazione delle variabili NUM1 e NUM2
8     RISULTATO=NUM1/NUM2;            //calcolo utilizzando l'operatore aritmetico divisione /
9     printf("%d diviso %d da' risultato %d\n",NUM1,NUM2,RISULTATO);
10    system("pause");
11 }
12

```

```

E:\Lezioni in C\printfvariabili  × + ▾

6 diviso 3 da' risultato 2
Premere un tasto per continuare . . .

```

Operatori aritmetici:

- + somma
- sottrazione
- * moltiplicazione
- / divisione
- % modulo cioè il resto di una divisione

Realizzare un programma che compia la somma, la moltiplicazione, la sottrazione e la divisione, il modulo di due numeri interi, stampando a monitor:

La somma tra 20 e 5 vale 25 (+)

La sottrazione tra 20 e 5 vale 15 (-)

La moltiplicazione tra 20 e 5 vale 100 (*)

La divisione tra 20 e 5 vale 4 (/)

Il resto tra 20 e 5 vale 0 (%)

```

1 //Utilizzare gli operatori aritmetici
2 /*Realizzare un programma che compia la somma,
3 //la moltiplicazione, la sottrazione e la divisione,
4 //il resto di due numeri interi, stampando a monitor:
5 La somma tra 20 e 5 vale 25
6 La sottrazione tra 20 e 5 vale 15
7 La moltiplicazione tra 20 e 5 vale 100
8 La divisione tra 20 e 5 vale 4
9 Il resto tra 20 e 5 vale 0*/
10
11 #include<stdio.h>           //direttiva libreria stdio
12
13 main(){
14     //dichiarazione delle variabili intere
15     int NUM1,NUM2,SOMMA,SOTTRAZIONE,MOLTIPLICAZIONE,DIVISIONE,RESTO;
16     //assegnazione dei valori a NUM1 e NUM2
17     NUM1=20;NUM2=5;
18     //operatori aritmetici (+,-,*,/,%)
19     SOMMA=NUM1+NUM2;
20     SOTTRAZIONE=NUM1-NUM2;
21     MOLTIPLICAZIONE=NUM1*NUM2;
22     DIVISIONE=NUM1/NUM2;
23     RESTO=NUM1%NUM2;
24     printf("%d sommato a %d da' risultato %d\n",NUM1,NUM2,SOMMA);
25     printf("%d sottratto %d da' risultato %d\n",NUM1,NUM2,SOTTRAZIONE);
26     printf("%d moltiplicato %d da' risultato %d\n",NUM1,NUM2,MOLTIPLICAZIONE);
27     printf("%d diviso %d da' risultato %d\n",NUM1,NUM2,DIVISIONE);
28     printf("Il resto di %d diviso %d da' risultato %d\n",NUM1,NUM2,RESTO);
29     system("pause");
30 }
31

```

Il modulo può essere utilizzato per verificare i multipli di un numero, o se diviso per 2 è un numero pari o dispari.

ASSEGNAZIONE DI UNA VARIABILE INTERA IN FORMATO BINARIO ED ESADECIMALE.

Si riporta l'esempio della variabile NUM assegnata nei diversi formati:

```

1
2
3 /*Assegnazione di una variabile intera nei formati decimale,
4 esadecimale e binario*/
5
6 #include<stdio.h>
7 main(){
8     int NUM=25;           //inizializzazione nel formato decimale
9     int NUM=0b11001;      //inizializzazione nel formato binario
10    int NUM=0x19;         //inizializzazione in formato esadecimale
11    printf("il numero e' %d\n",NUM);
12    system("pause");    //consente di visualizzare i risultati
13 }
14

```

FUNZIONE DI INPUT scanf();

La funzione **scanf()** consente di assegnare ad una variabile un valore da digitato da tastiera. Si riporta la sintassi dove NUM è in nome identificativo della variabile. Anche con questa funzione a seconda del tipo di variabile da acquisire devo precisare il formato nel caso riportato **%d** intero.

scanf("%d",&NUM);

Per poter essere impiegata dobbiamo da subito indicare al compilatore l'utilizzo della libreria stdio tramite la direttiva **#include<stdio.h>**.

Esempio di assegnazione di variabile da tastiera.

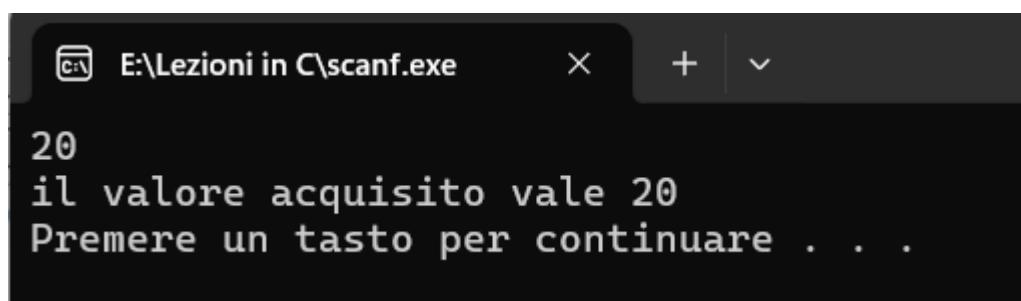
Scrivere un programma che legga da tastiera il valore intero NUM e riporti a monitor la seguente stringa “Il valore acquisito vale 20”.

\

```

1 //Scrivere un programma che legga da tastiera il valore
2 //intero NUM1 e riporti a monitor la seguente stringa "Il valore letto vale 20".
3 #include<stdio.h>
4 main(){
5     int NUM;
6     scanf("%d",&NUM);
7     printf("il valore acquisito vale %d\n",NUM);
8     system("pause");
9 }
10

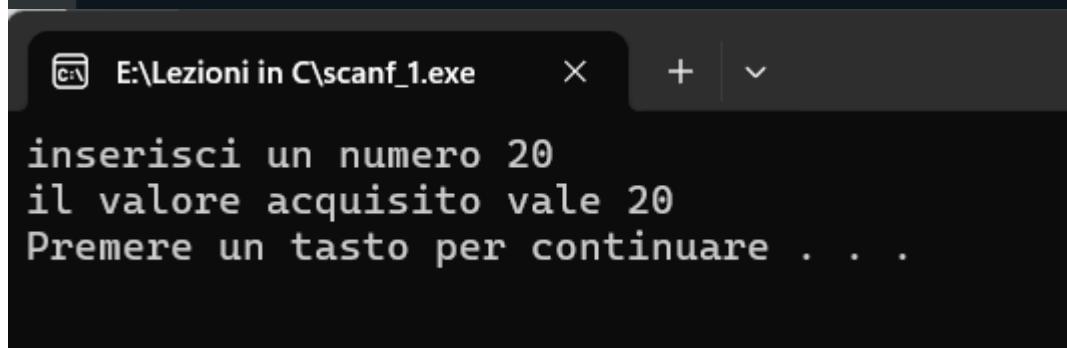
```



```
E:\Lezioni in C\scansf.exe
20
il valore acquisito vale 20
Premere un tasto per continuare . . .
```

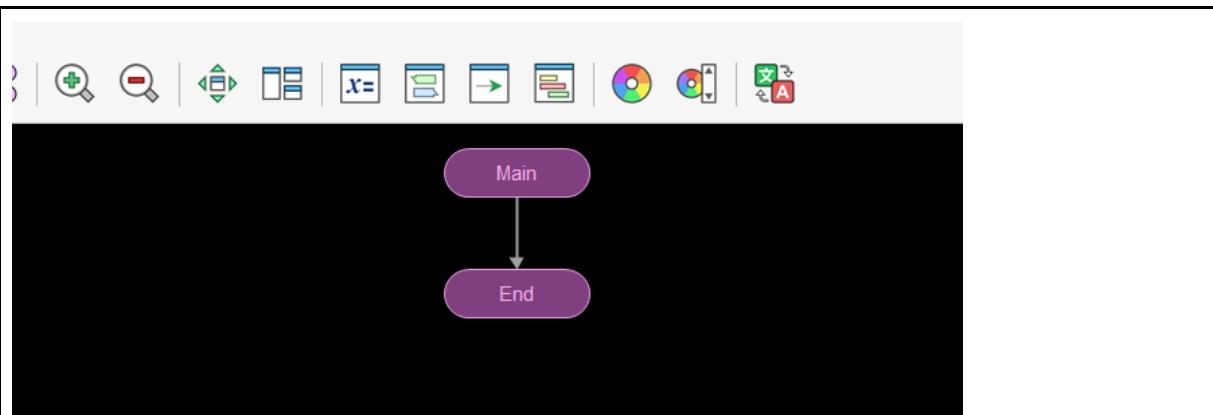
modificare il programma in modo che chieda di inserire un numero.

```
1 //Scrivere un programma che legga da tastiera il valore
2 //intero NUM1 e riporti a monitor la seguente stringa "Il valore letto vale 20".
3 #include<stdio.h>
4 main(){
5     int NUM;
6     printf("inserisci un numero ");
7     scanf("%d",&NUM);
8     printf("il valore acquisito vale %d\n",NUM);
9     system("pause");
10 }
11 |
```

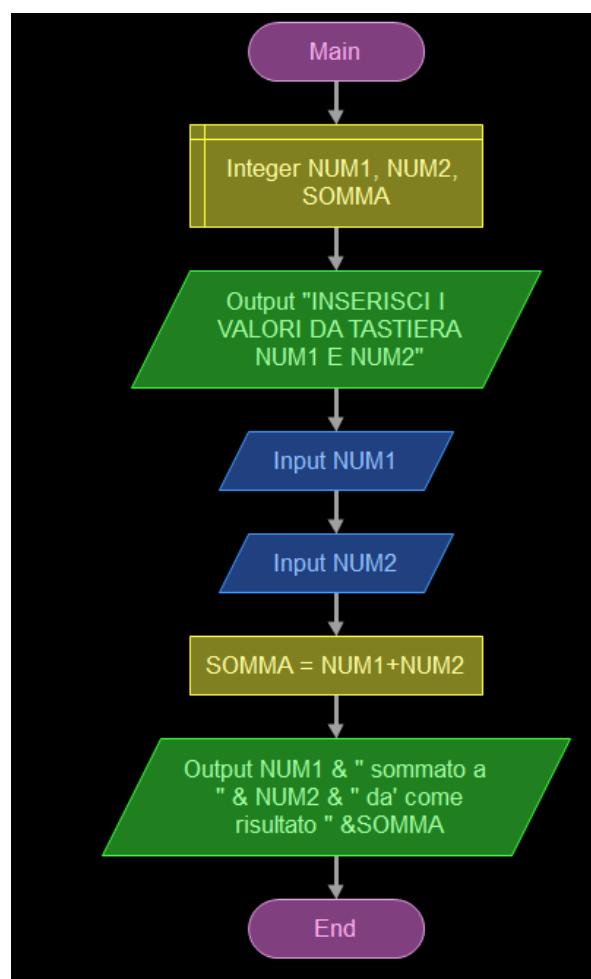


```
E:\Lezioni in C\scansf_1.exe
inserisci un numero 20
il valore acquisito vale 20
Premere un tasto per continuare . . .
```

STRUMENTO FLOWGORITHM



proviamo a realizzare il problema relativo agli operatori aritmetici



TIPI DI DATO: CARATTERE

I caratteri riconosciuti dal compilatori sono quelli del codice ASCII (American Standard Code for Information Interchange).

Al singolo carattere è associata un'area di memoria pari a 8bit.

**Decimal - Binary - Octal - Hex – ASCII
Conversion Chart**

Decimal	Binary	Octal	Hex	ASCII	Decimal	Binary	Octal	Hex	ASCII	Decimal	Binary	Octal	Hex	ASCII	Decimal	Binary	Octal	Hex	ASCII
0	00000000	000	00	NUL	32	00100000	040	20	SP	64	01000000	100	40	@	96	01100000	140	60	~
1	00000001	001	01	SOH	33	00100001	041	21	!	65	01000001	101	41	A	97	01100001	141	61	a
2	00000010	002	02	STX	34	00100010	042	22	"	66	01000010	102	42	B	98	01100010	142	62	b
3	00000011	003	03	ETX	35	00100011	043	23	#	67	01000011	103	43	C	99	01100011	143	63	c
4	00000100	004	04	EOT	36	00100100	044	24	\$	68	01000100	104	44	D	100	01100100	144	64	d
5	00000101	005	05	ENQ	37	00100101	045	25	%	69	01000101	105	45	E	101	01100101	145	65	e
6	00000110	006	06	ACK	38	00100110	046	26	&	70	01000110	106	46	F	102	01100110	146	66	f
7	00000111	007	07	BEL	39	00100111	047	27	'	71	01000111	107	47	G	103	01100111	147	67	g
8	00001000	010	08	BS	40	00101000	050	28	(72	01001000	110	48	H	104	01101000	150	68	h
9	00001001	011	09	HT	41	00101001	051	29)	73	01001001	111	49	I	105	01101001	151	69	i
10	00001010	012	0A	LF	42	00101010	052	2A	*	74	01001010	112	4A	J	106	01101010	152	6A	j
11	00001011	013	0B	VT	43	00101011	053	2B	+	75	01001011	113	4B	K	107	01101011	153	6B	k
12	00001100	014	0C	FF	44	00101100	054	2C	,	76	01001100	114	4C	L	108	01101100	154	6C	l
13	00001101	015	0D	CR	45	00101101	055	2D	-	77	01001101	115	4D	M	109	01101101	155	6D	m
14	00001110	016	0E	SO	46	00101110	056	2E	.	78	01001110	116	4E	N	110	01101110	156	6E	n
15	00001111	017	0F	SI	47	00101111	057	2F	/	79	01001111	117	4F	O	111	01101111	157	6F	o
16	00010000	020	10	DLE	48	00110000	060	30	0	80	01001000	120	50	P	112	01100000	160	70	p
17	00010001	021	11	DC1	49	00110001	061	31	1	81	01001001	121	51	Q	113	01100001	161	71	q
18	00010010	022	12	DC2	50	00110010	062	32	2	82	01001010	122	52	R	114	01100010	162	72	r
19	00010011	023	13	DC3	51	00110011	063	33	3	83	01001011	123	53	S	115	01100011	163	73	s
20	00010100	024	14	DC4	52	00110100	064	34	4	84	010010100	124	54	T	116	01101000	164	74	t
21	00010101	025	15	NAK	53	00110101	065	35	5	85	010010101	125	55	U	117	01101010	165	75	u
22	00010110	026	16	SYN	54	00110110	066	36	6	86	010010110	126	56	V	118	01101010	166	76	v
23	00010111	027	17	ETB	55	00110111	067	37	7	87	010010111	127	57	W	119	01101011	167	77	w
24	00011000	030	18	CAN	56	00111000	070	38	8	88	010011000	130	58	X	120	01110000	170	78	x
25	00011001	031	19	EM	57	00111001	071	39	9	89	010011001	131	59	Y	121	01110001	171	79	y
26	00011010	032	1A	SUB	58	00111010	072	3A	:	90	010011010	132	5A	Z	122	01110010	172	7A	z
27	00011011	033	1B	ESC	59	00111011	073	3B	:	91	010011011	133	5B	[123	01110011	173	7B	{
28	00011100	034	1C	FS	60	00111100	074	3C	<	92	010011100	134	5C	\	124	01111100	174	7C	
29	00011101	035	1D	GS	61	00111101	075	3D	=	93	010011101	135	5D]	125	01111101	175	7D	}
30	00011110	036	1E	RS	62	00111110	076	3E	>	94	010011110	136	5E	^	126	01111110	176	7E	-
31	00011111	037	1F	US	63	00111111	077	3F	?	95	010011111	137	5F	_	127	01111111	177	7F	DEL

Istruzione di dichiarazione di una variabile del tipo carattere **char xxx;**

dove x è in nome della variabile,

Istruzione di assegnazione **xxx='a';** con a carattere assegnato da tastiera compreso tra quelli del codice ascii.

Istruzione di inizializzazione **char xxx='a';**

E' possibile dichiarare un carattere non presente da tastiera utilizzando il valore espresso in esadecimale:

char xxx = '\x40' equivale a **char xxx='@'**

Esercizio:

inizializzare una variabile LETTERA assegnando il carattere @ da tastiera, visualizzare la stringa in output utilizzando la funzione printf(). Ripetere l'esercizio assegnando il corrispondente codice esadecimale \x40.

```

1 /*Inizializzare una variabile tipo char utilizzando come identificatore
2 //LETTERA assegnando il carattere @ da tastiera,
3 //visualizzare la stringa: La lettera è @
4 Ripetere l'esercizio assegnando il corrispondente codice esadecimale e binario.*/
5
6 #include<stdio.h>
7 main(){
8     char LETTERA='@';           //inizializzazione da tastiera
9     char LETTERA='\x40';       //inizializzazione con codice esadecimale
10    printf("la lettera e' %c\n",LETTERA);
11    system("pause");          //consente di visualizzare i risultati
12 }

```

Modificare il programma in modo da compiere una lettura da tastiera del carattere @. Quando inserisco il valore da tastiera non servono gli apici.

```

1 /*Inizializzare una variabile tipo char utilizzando come identificatore
2 //LETTERA, acquisire il carattere @ assegnandolo da tastiera,
3 //visualizzare la stringa: il carattere è @ */
4
5 #include<stdio.h>
6 main(){
7     char LETTERA;
8     printf("inserisci un carattere ");
9     scanf("%c",&LETTERA);
10    printf("il carattere e' %c\n",LETTERA);
11    system("pause");          //consente di visualizzare i risultati
12 }

```

DATI DEL TIPO FLOAT DETTI A VIRGOLA MOBILE:

L'istruzione per dichiarare che il dato è reale è la seguente:

float xxx;

dove *xxx* è *in nome della variabile*.

Per rappresentare numeri reali si attribuiscono 32 bit secondo la CODIFICA IEEE754 IN VIRGOLA MOBILE A SINGOLA PRECISIONE.



Il valore del numero rappresentato è calcolabile come:

$$N = (-1)^S \times 2^{(E)} \times ((1 \times 2^0) + \text{Mantissa})$$

Il campo S specifica il segno del numero: 0 per i numeri positivi, 1 per i numeri

negativi.

L'esponente E è compreso tra: $-126 < E < 127$

Mantissa = $(0 \text{ o } 1) * 2^{-1} + (0 \text{ o } 1) * 2^{-2} + \dots + (0 \text{ o } 1) * 2^{-23}$

La mantissa è un valore compreso tra 0 e 1

si rimanda al foglio in google calcoli.

Realizzare un programma che acquisisca da tastiera un valore reale e lo visualizzi a monitor “il valore inserito vale 25.5”.

```
1 #include<stdio.h>      //dichiaro utilizzo libreria stdio per poter
2                                //utilizzare printf() e scanf()
3 main(){
4     float NUM;          //dichiarazione variabile reale
5     printf("inserisci un valore ");
6     scanf("%f",&NUM);
7     printf("il valore NUM vale %f\n",NUM);
8     system("pause");
9 }
```

```
E:\Lezioni in C\float.exe
inserisci un valore 25.5
il valore NUM vale 25.500000
Premere un tasto per continuare . . . |
```

LE COSTANTI

Per definire un valore costante si utilizza la direttiva

#define XXX VALORE;

con XXX l'identificatore, sono valide le stesse regole delle variabili

```

1 //programma che calcola l'area di un cerchio
2 #include<stdio.h>           //Libreria per input e output
3 #define PIGRECO 3.141593    //costante
4
5 main(){
6     float diametro,area;
7     printf("inserisci il diametro del cerchio in mm ");
8     scanf("%f",&diametro);
9     area=PIGRECO*diametro*diametro/4;
10    printf("L'area vale %f mmq\n",area);
11    system("pause");
12 }
13

```

E:\Lezioni in C\Costanti.exe

inserisci il diametro del cerchio in mm 2
L'area vale 3.141593 mmq
Premere un tasto per continuare . . . |

Esercizio n°1 Scrivere a monitor: ciao mondo!

Esercizio n°2 L'utente inserisce 2 numeri interi e il programma calcola: la somma, la differenza, il prodotto, il quoziente e il resto.

Esercizio n°3: L'utente inserisce quattro numeri reali e il programma calcola la media.

Esercizio n°4: L'utente inserisce base e altezza di un rettangolo e il programma calcola il perimetro e l'area.

Esercizio n°5: L'utente inserisce il lato del quadrato e il programma calcola l'area e il perimetro.

Esercizio n°6: Dati 3 numeri interi considerarli come hh:mm:ss di un orario determinato. Calcola i secondi trascorsi dalla mezzanotte del giorno prima.

Esercizio n°7: Sapendo che in un parcheggio la prima ora costa 2€ tutte le successive costano 1€.

Scrivere un programma che richieda il numero complessivo delle ore di parcheggio e visualizzi il totale da pagare

PROGRAMMAZIONE IN LINGUAGGIO C

- LIBRERIA MATH
- FUNZIONI FMIN,FMAX,SQRT,ABS,POW,FLOOR,CEIL
- OPERATORI RELAZIONALI, LOGICI E ORIENTATI AL BIT

LIBRERIA MATH

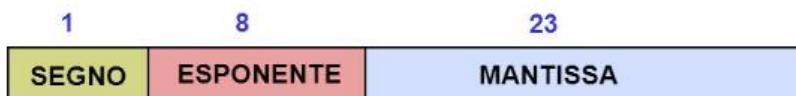
La libreria math racchiude al suo interno diverse funzioni che possono agevolare diversi calcoli matematici e procedimenti logici.

Per poterla utilizzare dobbiamo inserire in cima al nostro programma il comando `#include <math>`. I dati che vengono passati alla funzione devono essere inizializzati del tipo “float”.

TIPO DATO FLOAT

Per rappresentare numeri reali si attribuiscono 4byte equivalenti a 32 bit secondo la CODIFICA IEEE 754 IN VIRGOLA MOBILE A SINGOLA PRECISIONE.

- Semplice precisione a 32 bit:



- Doppia precisione a 64 bit



FUNZIONI DELLA LIBRERIA MATH

Elenchiamo dunque alcune funzioni della libreria math. Attenzione i parametri passati alle funzioni devono essere di tipo float.

- `fmin (num1, num2)` – serve a trovare il valore minimo tra i due numeri passati come parametri alla funzione.
- `fmax (num1, num2)` – serve a trovare il valore massimo tra i due numeri passati come parametri alla funzione.
- `abs (num)` – serve a trovare il valore assoluto del numero passato come parametro.
- `pow (numero, esponente)` – serve a calcolare l'elevamento a potenza. In pratica viene calcolato il valore di base elevato all'esponente passati come parametri.

- `sqrt (num)` – serve a calcolare la radice quadrata del numero passato come parametro.
- `floor (num)` – serve ad arrotondare il numero per difetto. In pratica restituisce la parte intera inferiore del numero.
- `ceil (num)` – serve ad arrotondare un numero per eccesso. In pratica restituisce la parte intera superiore del numero.

```

1 // Realizzare un programma utilizzando la libreria <math.h>
2 //che chieda a monitor tramite visualizzazione di inserire due numeri
3 // che dati due numeri NUM1,NUM2 calcoli il minimo e il massimo,
4 //e visualizzi "il valore minimo vale" e "il valore massimo vale".
5
6 #include<stdio.h>
7 #include<math.h>
8
9 ■ main(){
10     float NUM1,NUM2,minimo,massimo;
11     printf("inserisci il primo valore ");
12     scanf("%f",&NUM1);
13     printf("inserisci il secondo valore ");
14     scanf("%f",&NUM2);
15     minimo=fmin(NUM1,NUM2);
16     massimo=fmax(NUM1,NUM2);
17     printf("Il valore minimo tra %f e %f vale %f\n",NUM1,NUM2,minimo);
18     printf("Il valore massimo tra %f e %f vale %f\n",NUM1,NUM2,massimo);
19     system("pause");
20 }
```

```

mathmin.c [ * ] math pow,sqrt.c
1 //Scrivere un programma che dato un numero in ingresso valuti
2 //la radice quadrata, il numero elevato alla terza, la radice cubica
3 //riportando a monitor la richiesta di inserire un numero e visualizzando
4 //il risultato ottenuto.
5
6 #include <stdio.h>
7 #include<math.h>
8
9 ■ main(){
10     float x,y,z,w;
11     printf("inserisci un numero ");
12     scanf("%f",&x);
13     y=sqrt(x);
14     z=pow(x,2);
15     w=pow(x,1.0/3);    //attenzione alla divisione!!!!!
16     printf("la radice quadrata di %f vale %f\n",x,y);
17     printf("l'elevazione al quadrato di %f vale %f\n",x,z);
18     printf("la radice cubica di %f vale %f\n",x,w);
19     system("pause");}
```

```

1 //Scrivere un programma che dato un numero in ingresso lo arrotondi per difetto,
2 //per eccesso e ne calcoli il valore assoluto
3
4 #include <stdio.h>
5 #include<math.h>
6
7 main() {
8     float x,y,z,w;
9     printf("inserisci un valore ");
10    scanf("%f",&x);
11    y=floor(x);
12    z=ceil(x);
13    w=abs(x);
14    printf("il valore arrotondato per difetto vale %f\n",y);
15    printf("il valore arrotondato per eccesso vale %f\n",z);
16    printf("il valore assoluto vale %f\n",w);
17    system("pause");
18 }
```

Esercizi:

Scrivere gli algoritmi in C che permettano di risolvere la seguenti espressioni con le tre incognite inserite dall'utente a , b ,c.

$$1. \quad X = \frac{a+b+c}{2}$$

$$2. \quad Y = \frac{b*c*\sin(a)}{2} \quad \text{sin}(a) \text{ a viene considerato in radianti}$$

$$3. \quad Z = \frac{a(b-c)}{a^3 + 2a^2}$$

$$4. \quad H1 = \frac{b - \sqrt{b^2 - 4ac}}{2a} \quad H2 = \frac{b + \sqrt{b^2 - 4ac}}{2a}$$

$$5. \quad F = \sqrt[3]{\frac{a^4 + 3a^3}{c(a+5b)}}$$

ATTENZIONE ALLE DIVISIONI!!!!!!!!!!!!!!

Regola: quando devo compiere una divisione e il risultato è con la virgola (quindi dichiarato come float) o il divisore o il dividendo deve essere dichiarato reale.

```
[*] attenzionedivisione.c
1 //Attenzione alla divisione tra numeri.
2 //Attenzione all'ordine delle operazioni (potenze,moltiplicazioni,
3 //divisioni,addizione e sottrazioni)
4
5 #include<stdio.h>
6
7 main(){
8     float rapporto;
9     //primo tentativo rapporto tra due numeri interi,
10    //secondo tentativo modifico il 3 in 3.0 o il 2 in 2.0 o entrambi
11    rapporto=3/2;
12    printf("il rapporto vale %f\n",rapporto);
13    system("pause");
14 }
```

OPERATORI RELAZIONALI

pone in confronto due valori, il risultato è TRUE o FALSE.

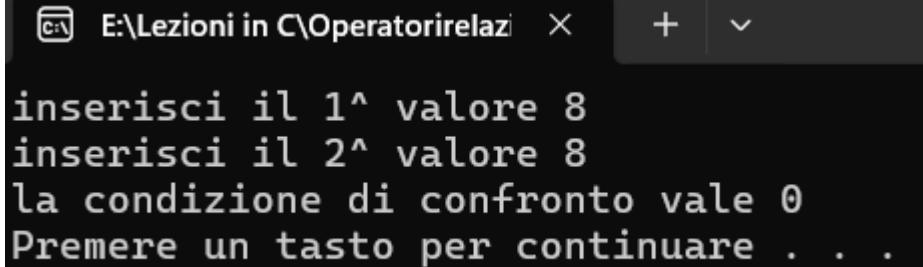
- > maggiore
- < minore
- \geq maggiore uguale
- \leq minore uguale
- \equiv uguale
- \neq diverso

Esercizio: dati due numeri NUM1 e NUM2, assegnare alla variabile x il risultato della comparazione. La variabile assumerà il valore pari a 1 se il risultato dato dal confronto è TRUE, 0 se il risultato del confronto è FALSE.

```

1 //Scrivere un programma che acquisiti da tastiera due numeri interi,
2 //utilizzando gli operatori relazionali, verifichi se num1>num2.
3 //Assegnare il risultato alla variabile conf
4 //è visualizzare tramite scritta il risultato dato dal confronto.
5
6 #include<stdio.h>
7 main(){
8     int num1,num2,conf;
9     printf("inserisci il 1^ valore ");
10    scanf("%d",&num1);
11    printf("inserisci il 2^ valore ");
12    scanf("%d",&num2);
13    conf=num1>num2;
14    printf("la condizione di confronto vale %d\n",conf);
15    system("pause");
16 }

```



```

inserisci il 1^ valore 8
inserisci il 2^ valore 8
la condizione di confronto vale 0
Premere un tasto per continuare . . .

```

OPERATORI LOGICI

NOT operatore di negazione

`!(espressione logica)` negazione dell'espressione logica

ad esempio: num1=10 e num2=5

risultato = !(num1>num2);

```

1 //Scrivere un programma che acquisiti da tastiera due numeri interi,
2 //utilizzando gli operatori relazionali, verifichi se num1>num2.
3 //Negare il risultato tramite l'operatore Logico NOT.
4 //Assegnare il risultato alla variabile conf
5 //è visualizzare tramite scritta il risultato dato dal confronto.
6
7 #include<stdio.h>
8 main(){
9     int num1,num2,conf;
10    printf("inserisci il 1^ valore ");
11    scanf("%d",&num1);
12    printf("inserisci il 2^ valore ");
13    scanf("%d",&num2);
14    conf=!(num1>num2);
15    printf("la condizione di confronto vale %d\n",conf);
16    system("pause");
17 }

```

```
E:\Lezioni in C\Operatorilogic × + ▾  
inserisci il 1^ valore 8  
inserisci il 2^ valore 4  
la condizione di confronto vale 0  
Premere un tasto per continuare . . . |
```

```
E:\Lezioni in C\Operatorilogic × + ▾  
inserisci il 1^ valore 8  
inserisci il 2^ valore 8  
la condizione di confronto vale 1  
Premere un tasto per continuare . . .
```

AND

(espressione logica 1) && (espressione logica 2)

esempio

```
risultato = (num1>=num2 && num1>=num3);
```

il risultato per essere vero (1) entrambe le espressioni logiche devono essere vere.

```
1 //Scrivere un programma che acquisiti da tastiera tre numeri interi,  
2 //utilizzando gli operatori relazionali, verifichi se num1>=num2 e num1>=num3  
3 //Assegnare il risultato alla variabile conf  
4 //è visualizzare tramite scritta il risultato dato dal confronto.  
5  
6 #include<stdio.h>  
7 main(){  
8     int num1,num2,num3,conf;  
9     printf("inserisci il 1^ valore ");  
10    scanf("%d",&num1);  
11    printf("inserisci il 2^ valore ");  
12    scanf("%d",&num2);  
13    printf("inserisci il 3^ valore ");  
14    scanf("%d",&num3);  
15    conf=(num1>=num2)&&(num1>=num3);  
16    printf("la condizione di confronto vale %d\n",conf);  
17    system("pause");  
18 }
```

```
E:\Lezioni in C\operatori logici × + ▾  
inserisci il 1^ valore 8  
inserisci il 2^ valore 4  
inserisci il 3^ valore 2  
la condizione di confronto vale 1  
Premere un tasto per continuare . . .
```

```
E:\Lezioni in C\operatori logici × + ▾  
inserisci il 1^ valore 8  
inserisci il 2^ valore 5  
inserisci il 3^ valore 10  
la condizione di confronto vale 0  
Premere un tasto per continuare . . .
```

OR

(espressione logica 1) || (espressione logica 2)

esempio

```
risultato = (num1>=num2 || num1>=num3);
```

il risultato è vero (1) se almeno una delle espressioni logiche è vera.

```
1 //Scrivere un programma che acquisiti da tastiera tre numeri interi,
2 //utilizzando gli operatori relazionali, verifichi se num1>=num2 o num1>=num3
3 //Assegnare il risultato alla variabile conf
4 //è visualizzare tramite scritta il risultato dato dal confronto.
5
6 #include<stdio.h>
7 main(){
8     int num1,num2,num3,conf;
9     printf("inserisci il 1^ valore ");
10    scanf("%d",&num1);
11    printf("inserisci il 2^ valore ");
12    scanf("%d",&num2);
13    printf("inserisci il 3^ valore ");
14    scanf("%d",&num3);
15    conf=(num1>=num2)|| (num1>=num3);
16    printf("la condizione di confronto vale %d\n",conf);
17    system("pause");
18 }
```

```
inserisci il 1^ valore 5
inserisci il 2^ valore 9
inserisci il 3^ valore 2
la condizione di confronto vale 1
Premere un tasto per continuare . . . |
```

OPERATORI ORIENTATI AL BIT

consentono di fare operazioni logiche tra numeri binari .

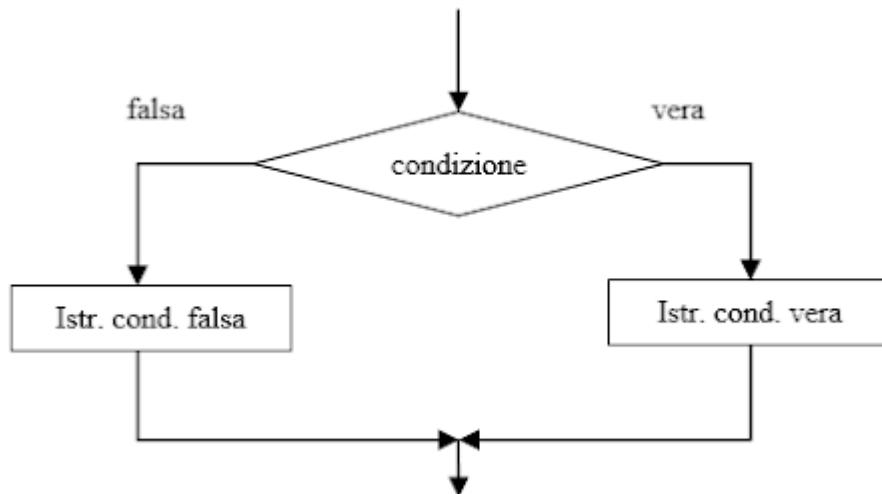
Esercizi 3 e 4 pag 105 del libro di testo
Esercizi pag 179 da 1 a 6.

- PROGRAMMAZIONE IN C: LE STRUTTURE CONDIZIONALI

- IF-ELSE
- IF
- IF-ELSE nidificati
- SWITCH

STRUTTURA CONDIZIONALE MULTIPLA IF-ELSE

Il ciclo if è condizionale, perchè il programma esegue istruzioni vere se è vera la condizione logica posta, le istruzioni false se è falsa la condizione logica.



if(espressione logica) istruzioni vera;
else istruzioni falsa;

Esempio teorico.

Dato un valore intero richiesto a monitor “scrivi un valore”: se è verificata la condizione per la quale un numero è minore o uguale a 10, scrivere a monitor “il valore è minore uguale a 10” altrimenti scrivere “il valore è maggiore di 10”.

```

1  /*Data un valore intero assegnato da tastiera
2  se è verificata la condizione per la quale un numero è minore
3  o uguale a 10, scrivere a monitor "il valore è minore uguale a 10"
4  altrimenti scrivere "il valore è maggiore di 10.*/
5
6 #include <stdio.h>
7
8 main(){
9     int x;
10    printf("scrivi un valore ");
11    scanf("%d",&x);
12    if(x<=10)printf("il valore e' minore uguale a 10\n");
13    else printf("il valore e' maggiore di 10\n");
14    system("pause");
15 }
16

```

E:\Lezioni in C\ifelse.exe

scrivi un valore 5
il valore e' minore uguale a 10
Premere un tasto per continuare . . .

ATTENZIONE: Se le istruzioni vere o false sono più di una ricordarsi di inserire le parentesi graffe, il compilatore ci segna errore.

```

ifelse_1.c
1  /*Data un valore intero richiesto a monitor "scrivi un valore";
2  se è verificata la condizione per la quale un numero
3  è minore o uguale a 10, scrivere a monitor "il valore è minore uguale a 10"
4  e assegnare alla variabile y il valore pari a 1, altrimenti scrivere
5  "il valore è maggiore di 10" e assegnare alla variabile w il valore pari a 1.*/
6
7
8 #include <stdio.h>
9
10 main(){
11     int x,y,w;
12     printf("scrivi un valore ");
13     scanf("%d",&x);
14     if(x<=10)
15         printf("il valore e' minore uguale a 10\n");
16         y=1;
17         printf("la variabile y vale %d\n",y);
18     else
19         printf("il valore e' maggiore di 10\n");
20         w=1;
21         printf("la variabile w vale %d\n",w);
22     system("pause");
23 }
24

```

Log di Compilazione Debug Risultati Ricerca Chiudi

C:\ifelse_1.c

In function 'main':
[Error] 'else' without a previous 'if'

Dato un valore intero richiesto a monitor “scrivi un valore”: se è verificata la condizione per la quale un numero è minore o uguale a 10, scrivere a monitor “il valore è minore uguale a 10” e assegnare alla variabile y il valore pari a 1, altrimenti scrivere “il valore è maggiore di 10” e assegnare alla variabile w il valore pari a 1.

```
1 /*Dato un valore intero richiesto a monitor "scrivi un valore":  
2 se è verificata la condizione per la quale un numero  
3 è minore o uguale a 10, scrivere a monitor "il valore è minore uguale a 10"  
4 e assegnare alla variabile y il valore pari a 1, altrimenti scrivere  
5 "il valore è maggiore di 10" e assegnare alla variabile w il valore pari a 1.*/  
6  
7  
8 #include <stdio.h>  
9  
10 main(){  
11     int x,y=0,w=0;  
12     printf("scrivi un valore ");  
13     scanf("%d",&x);  
14     if(x<=10){  
15         printf("il valore e' minore uguale a 10\n");  
16         y=1;  
17         printf("la variabile y vale %d\n",y);}  
18     else{  
19         printf("il valore e' maggiore di 10\n");  
20         w=1;  
21         printf("la variabile w vale %d\n",w);}  
22     system("pause");  
23 }
```

```
E:\Lezioni in C\ifelse_1.exe      X + ▾  
scrivi un valore 12  
il valore e' maggiore di 10  
la variabile w vale 1  
Premere un tasto per continuare .
```

Provare a togliere le parentesi che racchiudono le istruzioni dopo Else e vedere cosa succede.

```
1  /*Data un valore intero richiesto a monitor "scrivi un valore":  
2   se è verificata la condizione per la quale un numero  
3   è minore o uguale a 10, scrivere a monitor "il valore è minore uguale a 10"  
4   e assegnare alla variabile y il valore pari a 1, altrimenti scrivere  
5   "il valore è maggiore di 10" e assegnare alla variabile w il valore pari a 1.*/  
6  
7  
8  #include <stdio.h>  
9  
10 main(){  
11     int x,y=0,w=0;  
12     printf("scrivi un valore ");  
13     scanf("%d",&x);  
14     if(x<=10){  
15         printf("il valore e' minore uguale a 10\n");  
16         y=1;  
17         printf("la variabile y vale %d\n",y);}  
18     else{  
19         printf("il valore e' maggiore di 10\n");  
20         w=1;  
21         printf("la variabile w vale %d\n",w);  
22     }  
23 }  
24
```

E:\Lezioni in C\ifelse_1.exe X + ▾

```
scrivi un valore 8  
il valore e' minore uguale a 10  
la variabile y vale 1  
la variabile w vale 1  
Premere un tasto per continuare . . .
```

Conclusione: il compilatore considera le istruzioni dopo la prima sempre esterne alla struttura condizionale e sempre da eseguire.

Esempio di programma pratico:

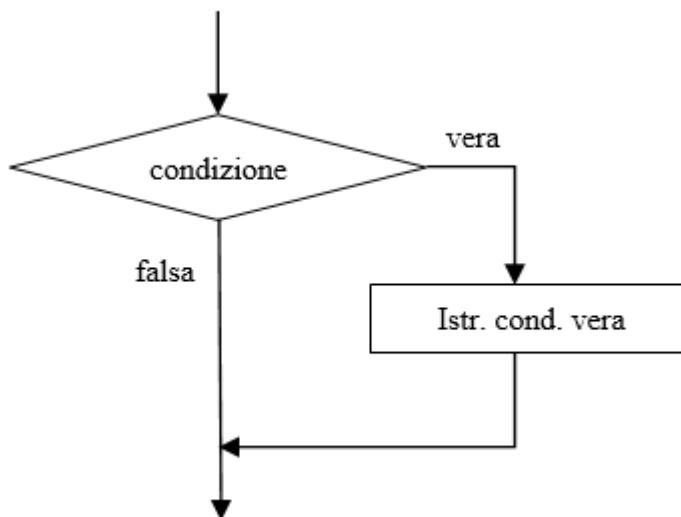
Una macchina è dotata di un selettore,
se collocato in ON la variabile y=1 (rotazione motore)
altrimenti se non è in ON y=0 (arresto motore).

```
1  /*Una macchina è dotata di un selettore,
2   se collocato in ON La variabile y=1 (rotazione motore)
3   altrimenti se non è in ON y=0 (arresto motore).*/
4
5  #include <stdio.h>
6
7 main(){
8     int x,y=0;      //variabile x definisce lo stato dell'interruttore, y lo stato del motore
9     printf("interruttore ON(1) / interruttore off(0) ");
10    scanf("%d",&x);
11    if(x==1){
12        y=1;
13        printf("motore avviato y=%d\n",y);}
14    else{
15        y=0;
16        printf("motore arrestato y=%d\n",y);}
17    system("pause");
18 }
```

```
interruttore ON(1) / interruttore off(0) 1
motore avviato y=1
Premere un tasto per continuare . . . |
```

```
interruttore ON(1) / interruttore off(0) 0
motore arrestato y=0
Premere un tasto per continuare . . . |
```

STRUTTURA CONDIZIONALE SEMPLICE IF



`if(espressione logica) istruzioni vera;`

Dato un valore intero richiesto a monitor “scrivi un valore”: se è verificata la condizione per la quale un numero è minore o uguale a 10, assegna ad una variabile il valore unitario, altrimenti non compie alcuna azione.

Esempio pratico: in un ambiente è presente un sensore che monitora la temperatura, se la temperatura scende sotto di 10° accendi la caldaia (variabile y), se la temperatura è superiore non fare nulla.

```

1  /*Dato un valore intero richiesto a monitor "scrivi un valore":  

2   se è verificata la condizione per la quale un numero è minore o uguale a 10,  

3   assegna ad una variabile il valore unitario, altrimenti non compie alcuna azione. */  

4  

5  

6  

7  #include <stdio.h>  

8  

9  main(){  

10     int x,y=0;  

11     printf("scrivi un valore ");  

12     scanf("%d",&x);  

13     if(x<=10){  

14         printf("il valore e' minore uguale a 10\n");  

15         y=1;  

16         printf("la variabile y vale %d\n",y);}  

17     system("pause");  

18 }  

19
  
```

```
scrivi un valore 8
il valore e' minore uguale a 10
la variabile y vale 1
Premere un tasto per continuare . . .
```

```
scrivi un valore 12
Premere un tasto per continuare . . . |
```

STRUTTURA CONDIZIONALE IF-ELSE NIDIFICATA O ANNIDATA

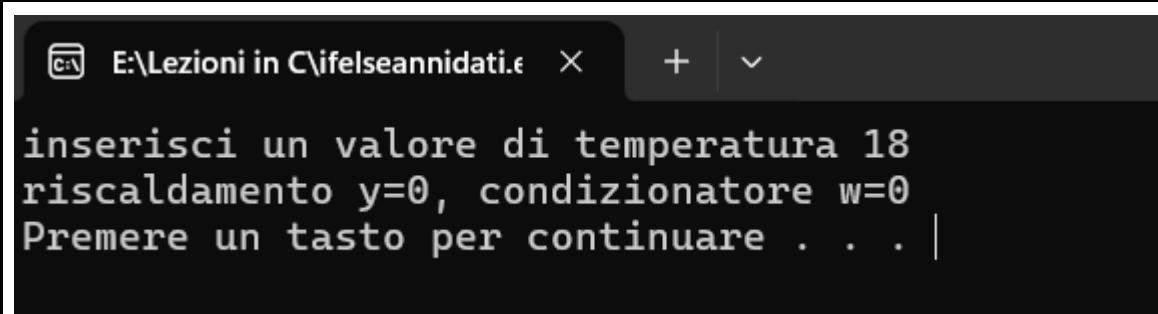
Quando la variabile deve soddisfare più di una espressione logica.

Ad esempio dobbiamo risolvere il seguente problema.

t° temperatura ambiente

se $t^{\circ} < 15^{\circ}\text{C}$ attiva in riscaldamento ($y=1, w=0$)
 mentre tra $15^{\circ}\text{C} \leq t^{\circ} \leq 20^{\circ}\text{C}$ non svolgere azioni ($y=0, w=0$)
 tra $t^{\circ} > 20^{\circ}\text{C}$ attiva condizionatore (variabile $y=0, w=1$)

```
1 /*In un ambiente è presente un sensore che monitora la temperatura:
2 se                               t<15°C attiva in riscaldamento (y=1,w=0)
3 mentre tra                      15°C <=t<= 20°C non svolgere azioni (y=0, w=0)
4 tra                            t>20°C attiva condizionatore (y=0,w=1)*/
5
6 #include <stdio.h>
7
8 main(){
9     int t,y=0,w=0;    //variabile t definisce la temperatura ambiente
10    printf("inserisci un valore di temperatura ");
11    scanf("%d",&t);
12    if(t<15){y=1;w=0;printf("riscaldamento y=%d, condizionatore w=%d \n",y,w);}
13    else if(t<=20) {y=0;w=0;printf("riscaldamento y=%d, condizionatore w=%d \n",y,w);}
14    else {y=0;w=1;printf("riscaldamento y=%d, condizionatore w=%d \n",y,w);}
15    system("pause");
16}
17
```



```

E:\Lezioni in C\ifelseannidati.c  X + ▾
inserisci un valore di temperatura 18
riscaldamento y=0, condizionatore w=0
Premere un tasto per continuare . . .

```

Esercizio pag 112 n°7

```

1 //Scrivere un programma che preveda l'inserimento di tre numeri e
2 //calcoli quanti di questi sono maggiori di 3 e quanti minori di 12
3
4 #include<stdio.h>
5 main(){
6     int num1,num2,num3,c3=0,c12=0;    //c3: contatore numeri maggiori di tre,
7     printf("inserisci num1 = ");
8     scanf("%d",&num1);
9     printf("inserisci num2 = ");
10    scanf( "%d",&num2);
11    printf("inserisci num3 = ");
12    scanf("%d",&num3);
13    if(num1>3)c3=c3+1;
14    if(num2>3)c3=c3+1;
15    if(num3>3)c3=c3+1;
16    if(num1<12)c12=c12+1;
17    if(num2<12)c12=c12+1;
18    if(num3<12)c12=c12+1;
19    printf("il numero di valori maggiori di 3 vale %d\n",c3);
20    printf("il numero di valori minori di 12 vale %d\n",c12);
21    system("pause");
22 }

```

Esercizio pag 112 n°8

```

1 //Scrivere un programma che preveda l'inserimento di tre numeri,
2 //e calcoli la somma dei due più grandi.
3
4 #include<stdio.h>
5 main(){
6     int num1,num2,num3,maggiore,intermedio;
7     float media;
8     printf("inserisci num1 = ");
9     scanf("%d",&num1);
10    printf("inserisci num2 = ");
11    scanf( "%d",&num2);
12    printf("inserisci num3 = ");
13    scanf("%d",&num3);
14    //determinazione del valore maggiore
15    if(num1>=num2)maggiore=num1;
16    else maggiore=num2;
17    if(num3>=maggiorer)e maggiore=num3;
18    printf("il valore maggiore vale %d\n",maggiorer);

```

```
19     //determinazione del valore intermedio
20     //verificando le diverse condizioni
21     //verifichiamo ordine num1>=num2>=num3
22     if(num2<=num1)
23         if(num2>=num3)intermedio=num2;
24     //verifichiamo ordine num3>=num2>=num1
25     if(num2<=num3)
26         if(num2>=num1)intermedio=num1;
27     //verifichiamo ordine num2>=num1>=num3
28     if(num1<=num2)
29         if(num1>=num3)intermedio=num1;
30     //verifichiamo ordine num3>=num1>=num2
31     if(num1<=num3)
32         if(num1>=num2)intermedio=num1;
33     //verifichiamo ordine num1>=num3>=num2
34     if(num3<=num1)
35         if(num3>=num2)intermedio=num3;
36     //verifichiamo ordine num2>=num3>=num1
37     if(num3<=num2)
38         if(num3>=num1)intermedio=num3;
39     printf("il valore intermedio vale %d\n",intermedio);
40     media=(intermedio+maggiore)/2.0;
41     printf("il valore della media tra maggiore e intermedio vale %f\n",media);
42     system("pause");
43 }
```

Provare a risolvere l'esercizio in modo differente determinando il valore minimo tra i numeri n1,n2,n3:

se minimo è uguale a n1 fai la media tra n2 e n3

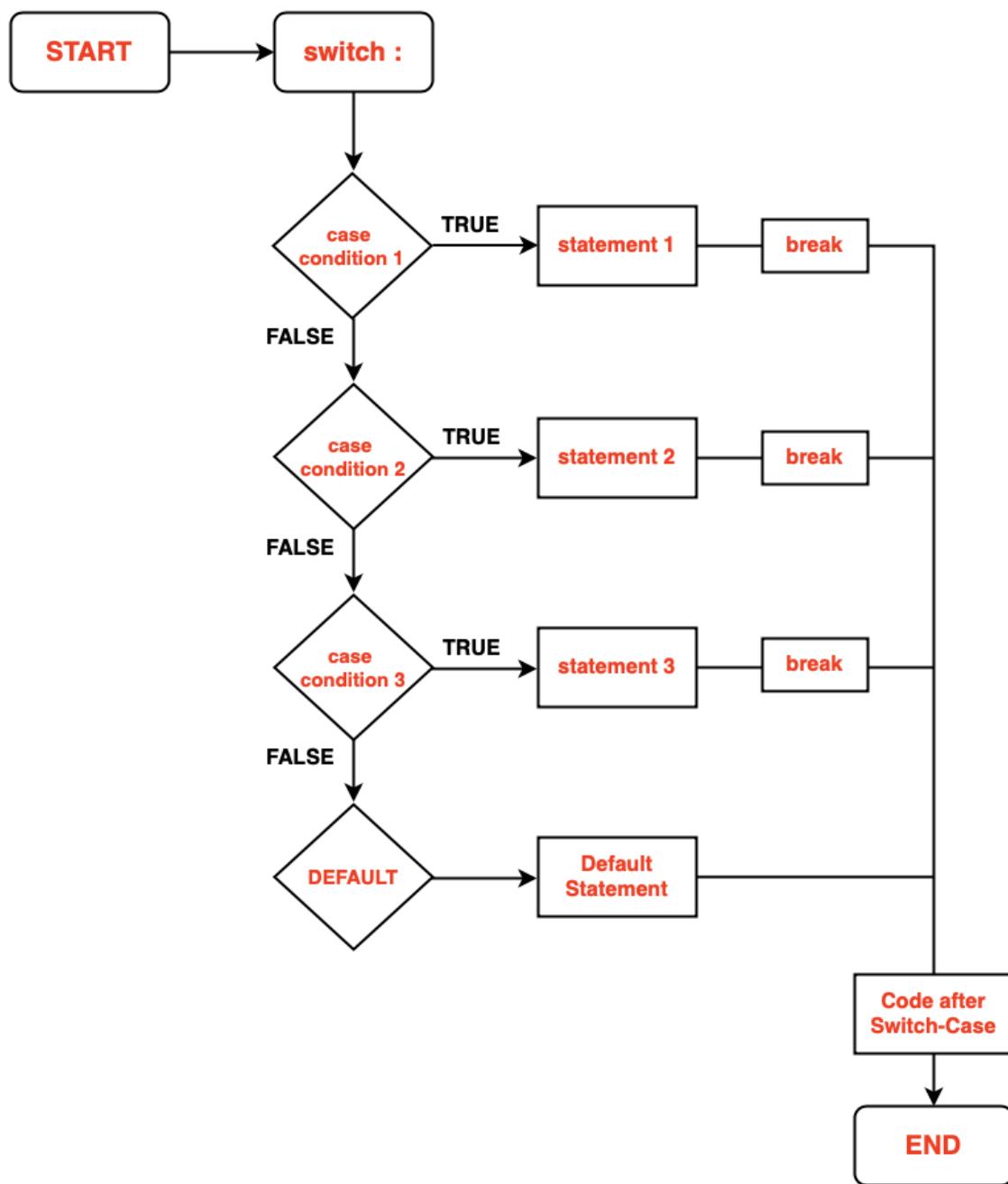
se minimo è uguale a n2 fai la media tra n1 e n3

se minimo è uguale a n3 fai la media tra n1 e n2

```
1 //Scrivere un programma che preveda l'inserimento di tre numeri,  
2 //e calcoli la somma dei due più grandi.  
3  
4 #include<stdio.h>  
5  
6 main(){  
7     int num1,num2,num3,minore;  
8     float media;  
9     printf("inserisci num1 = ");  
10    scanf("%d",&num1);  
11    printf("inserisci num2 = ");  
12    scanf( "%d",&num2);  
13    printf("inserisci num3 = ");  
14    scanf("%d",&num3);  
15    //determinazione del valore minore  
16    if(num1<=num2)minore=num1;  
17    else minore=num2;  
18    if(num3<=minore)minore=num3;  
19    printf("il valore minore vale %d\n",minore);  
20    //comparazione tra minore e i valori num1,num2,num3  
21    if(minore==num1)media=(num2+num3)/2.0;  
22    if(minore==num2)media=(num1+num3)/2.0;  
23    if(minore==num3)media=(num1+num2)/2.0;  
24    printf("il valore della media vale %f\n",media);  
25    system("pause"); }
```

STRUTTURA SWITCH

Le condizioni da verificare sono numerose cioè il dato può assumere differenti valori:



Sintassi in C:

```
switch(variabile){  
    case caso1:  
        istruzioni caso1;  
        break;  
  
    case case2:  
        istruzioni caso2;  
        break;  
  
    case caseN:  
        istruzioni casoN;  
        break;  
  
    default:  
        istruzioni nel caso non si verifichino gli altri casi;  
}
```

Esempio programma che richiede un numero tra compreso tra 0 e 4 e visualizzi a monitor il numero stampato scelto.

```
5  #include<stdio.h>
6
7  main(){
8      int num;
9  printf("inserisci un valore ");
10 scanf("%d",&num);
11
12 switch(num){
13     case 0:
14         printf("il numero scelto e'zero\n");
15         break;
16     case 1:
17         printf("il numero scelto e'uno\n");
18         break;
19     case 2:
20         printf("il numero scelto e'due\n");
21         break;
22     case 3:
23         printf("il numero scelto e'tre\n");
24         break;
25     case 4:
26         printf("il numero scelto e'quattro\n");
27         break;
28     default:
29         printf("nessun numero ammissibile e'stato scelto\n");
30     system("pause");
31 }
32
```

in alternativa scrivere un programma con tanti if in sequenza.

```
1 //Programma che consente di scegliere da tastiera un numero tra 0 e 4
2 //riporta a monitor una scritta con il numero scelto.
3
4
5 #include<stdio.h>
6
7 main(){
8     int num;
9  printf("inserisci un valore ");
10 scanf("%d",&num);
11 // in alternativa allo switch, utilizzo if in sequenza
12
13 if(num==0) printf("il numero scelto e'zero\n");
14 if(num==1) printf("il numero scelto e'uno\n");
15 if(num==2) printf("il numero scelto e'due\n");
16 if(num==3) printf("il numero scelto e'tre\n");
17 if(num==4) printf("il numero scelto e'quattro\n");
18 if((num!=0)&&(num!=1)&&(num!=2)&&(num!=3)&&(num!=4))
19 printf("il numero scelto non e' ammissibile'\n");
20 system("pause");
21 }
```

Esercizio 3 pag 180

Esercizio n°3 pag 180

```
1 /*Esercizio 3 pag 180: argomento if-else nidificati
2 Scrivere un programma che acquisisca le età di un gruppo di studenti
3 e stampi il numero di teen-agers, con età compresa tra 13 e 19 anni, e
4 la media delle età quindi 13anni<=età<=19anni
5 */
6
7
8 #include<stdio.h>
9 main(){
10     //definisco che il gruppo sia costituito da quattro componenti
11     int st1,st2,st3,st4,conteggio=0;
12     printf("inserisci l'eta' del primo studente ");
13     scanf("%d",&st1);
14     printf("inserisci l'eta' del secondo studente ");
15     scanf("%d",&st2);
16     printf("inserisci l'eta' del terzo studente ");
17     scanf("%d",&st3);
18     printf("inserisci l'eta' del quarto studente ");
19     scanf("%d",&st4);
20     if(st1>=13)if(st1<=19)conteggio=conteggio+1;
21     if(st2>=13)if(st2<=19)conteggio=conteggio+1;
22     if(st3>=13)if(st3<=19)conteggio=conteggio+1;
23     if(st4>=13)if(st4<=19)conteggio=conteggio+1;
24     printf("il numero di studenti entro il limite e'%d\n",conteggio);
25
26     system("pause");
27 }
```

Esercizio n°5 pag 180

```
1 /*Scrivere un programma che richieda il numero di pezzi acquistati
2 di un articolo e il costo unitario e calcoli il costo complessivo,
3 applicando uno sconto diverso in funzione del numero dei pezzi
4 1<=PEZZI<=3 nessuno sconto, se 3<PEZZI<=10 sconto 10% PEZZI>10 sconto 20%*/
5
6 #include<stdio.h>
7 main(){
8     int npezzi;
9     float costouni,sconto,prezzotot;
10    printf("inserisci il numero di pezzi da acquistare ");
11    scanf("%d",&npezzi);
12    printf("inserisci il costo unitario ");
13    scanf("%f",&costouni);
14    /*
15     if(npezzi>=1)      if(npezzi<=3)sconto=0;
16     |           |      else  if(npezzi<=10)sconto=0.1;
17     |           |      |      else sconto=0.2;
18     */
19    // in alternativa
20    if(npezzi>10)sconto=0.2;
21    else if(npezzi>3)sconto=0.1;
22    else sconto=0;
23    prezzotot=(npezzi*costouni)-(npezzi*costouni*sconto);
24    printf("il prezzo totale e'%f\n",prezzotot);
25
26 system("pause");}
```

Esercizio n°6 pag 180

```
1 /*Dopo aver inserito da tastiera il peso corporeo P e l'altezza A
2 di una persona, calcolare l'indice di massa corporea mediante la formula:
3 IMC=P/A^2. Successivamente stampare la condizione in funzione
4 IMC<18.5 SOTTOPESO
5 18.5<=IMC <25 NORMOPESO
6 25<=IMC<30 SOVRAPPESO
7 30<=IMC<=40 OBESITA' DI MEDIO GRADO
8 IMC>40 OBESITA' DI ALTO GRADO*/
9
10 #include<stdio.h>
11 main(){
12     float A,P,IMC;
13     printf("inserisci il peso in kg ");
14     scanf("%f",&P);
15     printf("inserisci l'altezza in metri ");
16     scanf("%f",&A);
17     IMC=P/(A*A);
18     if(IMC>40) printf("OBESITA' DI ALTRO GRADO\n");
19     else if(IMC>=30) printf("OBESITA' DI MEDIO GRADO\n");
20     else if(IMC>=25) printf("SOVRAPPESO\n");
21     else if (IMC>=18.5) printf("NORMOPESO\n");
22     else printf("SOTTOPESO\n");
23     system("pause");
24 }
```

1. Scrivere un programma che, inseriti dall'utente 2 numeri interi, li ordina in modo crescente.
2. L'utente inserisce l'età e il programma restituisce se maggiorenne o minorenne.
3. L'utente inserisce due numeri e il programma scrive se il primo è multiplo del secondo.
4. Inserire un valore e stabilire se è positivo o negativo, e se pari o dispari.
5. Inseriti 3 numeri, il programma restituisce se il secondo numero è compreso o no tra gli altri due.
6. L'utente inserisce un carattere e il programma scrive se è una vocale o

- una consonante.
7. Scrivere un programma che inserito un voto numerico dall'utente, restituisce se è sufficiente, insufficiente o gravemente insufficiente.
 8. Su una linea ferroviaria, rispetto alla tariffa piena, i pensionati usufruiscono di uno sconto del 10%, gli studenti del 15% e i disoccupati del 25%. Codificando i pensionati con 'p' gli studenti con 's' e i disoccupati con 'd' scrivere un programma che richiesto il costo del biglietto e la condizione dell'utente visualizzi l'importo.
 9. Scrivere un programma che legga da input la lunghezza dei lati di un triangolo e determini se il triangolo è equilatero, isoscele o scaleno.
 10. Realizzare un programma che legga da input tre caratteri, e li stampa in ordine alfabetico. (consultare tabella ASCII).
 11. Scrivere un programma che richieda in ingresso tre valori interi e ne determini il maggiore.
 12. Scrivere un programma che richieda i valori numerici Fascia1, Fascia2, Fascia3 dei colori delle 3 fasce di un resistore e stampi il nome dei colori e il valore resistivo.
 13. Ordinare 3 numeri in ordine CRESCENTE
 14. L'utente inserisce due numeri interi e il simbolo dell'operazione '+', '-', '*' , '/' , il programma calcola il risultato.

1. Scrivere un programma che richieda il numero del mese e visualizza il suo nome per esteso.
In caso di errore scrive: "Numero del mese inserito non valido".
2. Ricevuto in ingresso un numero intero compreso tra 0 e 9 inclusi, scrivere il numero in parole.
In caso di errore scrive: "Numero non valido".
3. Mediante un menù a scelta l'utente inserisce: 'T' per calcolare l'area del triangolo, 'R' per l'area del rettangolo, 'C' per l'area del cerchio. I dati verranno richiesti all'utente dopo la scelta.
4. In un albergo il prezzo della camera dipende dal piano in cui ci si trova secondo la seguente tabella: piano 1 → 35€ piano 2 → 45€ piano 3 → 55€ piano 4 → 65€. Calcolare il prezzo del conto sapendo il numero dei giorni di soggiorno e il piano della camera, sottraendo un buono sconto del 10% sul totale.
5. L'utente inserisce un numero intero che corrisponde al mese, e il programma visualizza la stagione corrispondente. Nel mese di marzo, giugno, settembre e dicembre richiedere anche il giorno.

PROGRAMMAZIONE IN LINGUAGGIO C: STRUTTURE ITERATIVE

- **CICLO FOR**
- **CICLO FOR NIDIFICATO**
- **WHILE**
- **DO WHILE**

STRUTTURA ITERATIVA CICLO FOR

Tale struttura iterativa si impiega quando si vuole compiere una operazione per un certo numero di volte.

Quando è noto a priori il numero di volte per cui si ripete l'azione si usa la struttura iterativa denominata ciclo for:

Segue la sintassi “con conteggio incrementale”:

`for(i=0;i<N;i=i+1) {sequenza di istruzioni;}`

i=0 inizializzazione

i<N ad ogni ciclo viene verificata la condizione

se la condizione è verificata: aggiorna la variabile i aggiungendo 1 ed esegue le istruzioni contenute tra parentesi, l’incremento della variabile i e le istruzioni vengono eseguite fino a che è vera i<N, quando falsa esce dal ciclo for. Il ciclo riportato si ripete per N volte, ciò non toglie che possa scriverlo in con conteggio in forma decrementale

`for(i=N;i>0;i=i-1) {sequenza di istruzioni;}`

la variabile i viene denominata contatore.

Programma che stampa i numeri da 0 a 4.

```
1 //Programma che stampa i numeri da 0 a 4
2
3 #include<stdio.h>
4 main(){
5     int i;
6     incremento del contatore da 0 a 4
7     for(i=0;i<5;i=i+1)
8         printf("%3d",i);
9         printf("\n");
10    /*decremento del contatore
11    for(i=4;i>=0;i=i-1)
12        printf("%3d",4-i);
13        printf("\n");*/
14
15    system("pause");
16 }
17 |
```

CICLO FOR NIDIFICATO

Si ha una struttura nidificata o annidata quando all'interno di un ciclo FOR è racchiuso un altro ciclo FOR.

```
for(i=0;i<N;i=i+1)
```

```
    for(j=0;j<M;j=j+1){istruzioni;}
```

Le parentesi grafe sono necessarie con più di una istruzione dopo il ciclo for interno.

Quando il ciclo interno (quello con variabile j) cicla per M volte, quello esterno (variabile i) compie un solo ciclo.

```

1 //Stampa della tavola pitagorica      1-2-3-4-5-6-7-8-9-10
2 //
3 //
4
5 #include<stdio.h>
6 main(){
7     int i,j;
8     // contatore esterno j definisce il numero della tabellina
9     // il contatore interno i moltiplicatore
10    for(j=1;j<=10;j=j+1){
11        for(i=1;i<=10;i=i+1)
12            printf("%3d",i*j);
13
14        printf("\n");
15
16        system("pause");
17    }
18

```

Esercizio 1 pag 181

```

1 /*Scrivere un programma che stampi i numeri interi da 0 a 49
2 su singola riga*/
3
4 #include<stdio.h>                  // direttiva che richiama la libreria con funzioni di I/O
5 main(){                                //funzione principale
6     int i;
7     for(i=0;i<50;i++)                //ciclo for iterativo per 50 volte da 0 a 49
8     printf("%3d",i);                  //stampa i valore della variabile i
9     printf("\n");                    //uscito dal ciclo il cursore va a capo.
10    system("pause");                //attende la pressione di un tasto per uscire,
11                                //consentendo di visualizzare la soluzione
12 }

```

Esercizio 2 pag 181

```

1 /*Scrivere un programma che stampi i numeri interi pari da 0 a 49
2 su singola riga*. E'abbligatorio l'uso del termine % per verificare se il numero
3 è pari*/
4
5 #include<stdio.h>                  // direttiva che richiama la libreria con funzioni di I/O
6 main(){                                //funzione principale
7     int i;
8     for(i=0;i<50;i++)
9     if(i%2==0){                      //se il resto è zero allora il numero è pari
10        printf("%3d",i);              //stampa il valore, altrimenti non fare nulla.
11        printf("\n");                //uscito dal ciclo il cursore va a capo.
12        system("pause");            //attende la pressione di un tasto per uscire,
13                                //consentendo di visualizzare la soluzione
14    }
15
16 // controllo della sintassi:
17 // le variabili sono tutte dichiarate?
18 // il tipo di variabili (int,float,char,string) è coerente alle funzioni printf e scanf?
19 // ci sono tutti i punti e virgola?
20 // le parentesi sono corrette?
21 // la frase "attendi la pressione di un tasto è andata a capo"?
22 // i numeri sono distanziati?
23

```

```

1  /*Scrivere un programma che stampi i numeri interi da 0 a 99
2  andando a capo ogni volta che un numero finisce per 3, verificando
3  quindi che il resto della divisione tra il numero e 10 dia 3 */
4
5  #include<stdio.h>           // direttiva che richiama la libreria con funzioni di I/O
6  main(){                      //funzione principale
7      int i;                  //dichiarazione variabile intera i
8      for(i=0;i<100;i++)       //ciclo for iterativo per 100 volte da 0 a 99
9      if(i%10!=3)             //se numero non termina per 3 stampa in linea, altrimenti
10     printf("%3d",i);        //va a capo.
11     else printf("%3d\n",i);
12     printf("\n");           //uscito dal ciclo il cursore va a capo.
13     system("pause");        //attende la pressione di un tasto per uscire,
14     | | | | | | | | | | | | //consentendo di visualizzare la soluzione
15 }

```

```

0  1  2  3
4  5  6  7  8  9  10 11 12 13
14 15 16 17 18 19 20 21 22 23
24 25 26 27 28 29 30 31 32 33
34 35 36 37 38 39 40 41 42 43
44 45 46 47 48 49 50 51 52 53
54 55 56 57 58 59 60 61 62 63
64 65 66 67 68 69 70 71 72 73
74 75 76 77 78 79 80 81 82 83
84 85 86 87 88 89 90 91 92 93
94 95 96 97 98 99

Premere un tasto per continuare . . .

```

Esercizio 4 pag 181

```

1  /*Scrivere un programma che acquisisca 5 numeri e non appena acquisiti li stampi.
2  In altre parole il programma deve richiedere "inserisci un numero" e una volta
3  inserito, deve stampare "hai inserito il valore"
4
5  */
6
7  #include<stdio.h>           // direttiva che richiama la libreria con funzioni di I/O
8  main(){                      //funzione principale
9      int i,NUM;               //dichiarazione variabile intera i e NUM
10     for(i=0;i<5;i++){
11         printf("inserisci un numero ");
12         scanf("%d",&NUM);
13         printf("hai inserito il valore %d\n",NUM);
14         printf("\n");
15         system("pause");
16     }
17 }

```

```

inserisci un numero 1
hai inserito il valore 1
inserisci un numero 2
hai inserito il valore 2
inserisci un numero 3
hai inserito il valore 3
inserisci un numero 4
hai inserito il valore 4
inserisci un numero 5
hai inserito il valore 5

Premere un tasto per continuare . . .

```

Esercizio 5 pag 181

```

1  /*Scrivere un programma che acquisisca 5 numeri e non appena acquisiti stampi solo quelli pari
2  */
3
4  #include<stdio.h>
5  main(){
6      int i,NUM;
7      for(i=0;i<5;i++){
8          printf("inserisci un numero ");
9          scanf("%d",&NUM);
10         if(NUM%2==0)
11             printf("hai inserito il valore %d\n",NUM);
12         printf("\n");
13         system("pause");
14     }
15 }
16

```

Esercizio 6 pag 181

```

1  /*Scrivere un programma che acquisisca 5 numeri e non appena acquisiti stampi solo quelli pari
2  e che scriva "inserisci il 1^ numero"
3  */
4
5  #include<stdio.h>
6  main(){
7      int i,NUM;
8      for(i=0;i<5;i++){
9          printf("inserisci il %d^ numero ",(i+1));
10         scanf("%d",&NUM);
11         if(NUM%2==0)
12             printf("hai inserito il valore %d\n",NUM);
13         printf("\n");
14         system("pause");
15     }
16 }
17

```

```

1 /*Scrivere un programma che acquisisca 5 numeri e calcoli e stampi la somma dei numeri pari*/
2 */
3
4 #include<stdio.h>
5 main(){
6     int i,NUM,SOMMAPARI=0;           // direttiva che richiama la libreria con funzioni di I/O
7     for(i=0;i<5;i++){               //funzione principale
8         printf("inserisci il %d^ numero ",(i+1)); //dichiarazione variabile intera i,NUM,SOMMAPARI
9         scanf("%d",&NUM);            //ciclo for iterativo per 5 volte da 0 a 4
10        if(NUM%2==0){              //visualizza la stringa richiamando la variabile i
11            SOMMAPARI=SOMMAPARI+NUM; //associa il valore alla variabile NUM
12        }                           //se pari è vero somma NUM a SOMMAPARI, else non fare nulla
13    }                             //usciti dal ciclo stampa SOMMAPARI
14    printf("\n");                 //uscito dal ciclo il cursore va a capo.
15    system("pause");             //attende la pressione di un tasto per uscire,
16 }                            //consentendo di visualizzare la soluzione
17

```

```

inserisci il 1^ numero 1
inserisci il 2^ numero 2
inserisci il 3^ numero 3
inserisci il 4^ numero 4
inserisci il 5^ numero 5
il valore della somma pari vale 6
Premere un tasto per continuare . . .

```

Esercizio 8 pag 181

```

1 /*Scrivere un programma che acquisisca 5 numeri e calcoli e stampi la somma dei numeri pari
e stampi quelli pari*/
2
3
4 #include<stdio.h>                   // direttiva che richiama la Libreria con funzioni di I/O
5 main(){
6     int i,NUM,SOMMAPARI=0;           //funzione principale
7     for(i=0;i<5;i++){               //dichiarazione variabile intera i,NUM,SOMMAPARI
8         printf("inserisci il %d^ numero ",(i+1)); //ciclo for iterativo per 5 volte da 0 a 4
9         scanf("%d",&NUM);            //visualizza la stringa richiamando la variabile i
10        if(NUM%2==0){              //associa il valore alla variabile NUM
11            SOMMAPARI=SOMMAPARI+NUM; //se pari è vero somma NUM a SOMMAPARI e stampa num pari
12            printf("hai inserito il valore pari %d\n",NUM);}}
13    printf("il valore della somma pari vale %d",SOMMAPARI); //usciti dal ciclo stampa SOMMAPARI
14    printf("\n");          //uscito dal ciclo il cursore va a capo.
15    system("pause");      //attende la pressione di un tasto per uscire,
16 }                            //consentendo di visualizzare la soluzione
17

```

```
inserisci il 1^ numero 1
inserisci il 2^ numero 2
hai inserito il valore pari 2
inserisci il 3^ numero 3
inserisci il 4^ numero 4
hai inserito il valore pari 4
inserisci il 5^ numero 5
il valore della somma pari vale 6
Premere un tasto per continuare . . . |
```

STRUTTURA ITERATIVA WHILE

while (condizione) {gruppo di istruzioni}

Se la condizione è VERA il programma esegue ciclicamente le istruzioni contenute tra parentesi graffe.

Se la condizione è FALSA il programma esce dal ciclo e continua con la parte di programma successiva.

Se da subito la condizione è FALSA, il programma salta alle istruzioni

successive, senza entrare nel ciclo.

```
1 //Programma che somma i numeri positivi ed uguali a zero
2 //ed esce quando inserisco un numero negativo
3
4
5 #include<stdio.h>
6
7 main(){
8     int num,somma=0;
9     printf("inserisci un valore ");
10    scanf("%d",&num);
11    while(num>=0){
12        somma=somma+num;
13        printf("inserisci un valore ");
14        scanf("%d",&num);}
15        printf("La somma vale %d\n",somma);
16        system("pause");
17 }
```

STRUTTURA ITERATIVA DO-WHILE

do {gruppo di istruzioni} while (condizione)

Il programma esegue almeno una volta il gruppo di istruzioni anche se la condizione è Falsa. Il programma poi verifica la condizione.

Se condizione è VERA il programma ritorna ad eseguire ciclicamente le istruzioni contenute tra parentesi graffe.

Se la condizione è FALSA il programma esce dal ciclo e continua con la parte di programma successiva.

```
1 //Somma numeri positivi ed esce quando La somma supera
2 //il valore 10
3
4
5 #include<stdio.h>
6
7 main(){
8     int num,somma=0;
9     do{
10         printf("inserisci un valore ");
11         scanf("%d",&num);
12         somma=somma+num;}
13         while(somma<=10);
14         printf("La somma vale %d\n",somma);
15         system("pause");
16 }
```

Esercizi con While e do While

Esercizio 1 pag 181

CICLO WHILE

1. Scrivere un programma che stampi i primi 100 numeri interi.
2. Scrivere i numeri naturali compresi tra 1 e un numero letto in input.

3. Calcolare la somma dei primi 100 numeri.
Poi dei primi n numeri, con n inserito dall'utente.
4. Calcolare la somma dei numeri pari fino a 100
5. Calcolare la media e la somma di 10 numeri inseriti a scelta dall'utente.
6. Scrivere un programma che stampi i numeri pari da 0 a 49.
7. L'utente inserisce un numero naturale n e scrive in output un numero n di '+'.
8. Scrivere un programma che legga 5 numeri da tastiera e ne restituisca il minore.
9. Scrivere la tavola pitagorica. (la tavola delle tabelline).
10. Leggere in input due numeri naturali b e h e in output far scrivere un rettangolo di "*" con b asterischi come base e h per l'altezza.
11. Visualizzare un rettangolo con cornice "+" e interno "=" di misure A e B, decise dall'utente.
12. Scrivere un programma che stampi i multipli pari di 5 fino a 100 compreso.
13. Dividere le cifre di un numero inserito dall'utente in cifra delle centinaia delle decine e delle unità.
14. Scrivere un programma che stampi i numeri da 0 a 99 andando a capo ogni volta che un numero finisce con 3. Suggerimento: Verificare che il numero diviso per 10 dia resto 3.
15. Convertire in lettere un numero inferiore a 999 (es. 123 centoventitre).
16. Scrivere un programma che verifichi se un numero è primo(senza usare il while).
17. Scrivere un programma che stampi i multipli di 3 fino a 99, andando a capo ogni 4 numeri.

1. Il programma richiede all'utente di inserire numeri interi finché la loro somma supera 100.
2. L'utente inserisce un numero maggiore di uno e il programma continua a calcolare e visualizzare le potenze del numero inserito fino a quando una potenza supera 5000. es. $2^2 - 2^3 - 2^4 \dots$
3. Il programma deve continuare a leggere numeri naturali e a calcolarne la somma, fermandosi quando legge uno zero.
4. L'utente inserisce una serie di numeri in ordine crescente. All'immissione di un numero minore rispetto al precedente, il programma si deve fermare e restituire la media dei valori inseriti.
5. L'utente inserisca un valore intero e il programma restituisca in output da quante cifre è composto il numero.
6. Il programma continua a leggere numeri naturali finchè l'utente non inserisce 0 e scrive quanti numeri pari e quanti numeri dispari sono stati inseriti.
7. Scrivere un programma che letto un numero naturale N compreso tra 0 e 126 inclusi, scriva in output il carattere ASCII corrispondente, fino a quando il numero inserito è uguale a 0. Avvisare l'utente che i caratteri tra 0 e 31 non sono stampabili. esempio: per stampare il carattere codificato con 89 nella tabella ASCII devo fare il casting

```
char car=char(89);
```

8. Scrivere un programma che stampi una volta uno, due volte due tre volte tre ecc.. fino a 9.

1

22

333...

9. Scrivere un programma che estraie numeri casuali da 0 a 30 e li stampa a schermo fino a quando estraе il numero 18.

10. Scrivere un programma "Indovina numero". All'inizio viene generato un numero casuale da 0 a 5 inclusi. Successivamente dentro un ciclo while viene richiesto all'utente di inserire un numero. Se i due numeri coincidono, ovvero se il giocatore indovina il numero generato dal computer, il ciclo while termina. All'uscita dal ciclo il programma stampa "Bravo, hai indovinato" oppure "Ritenta".

11. Modificare il programma 10 in modo tale che il PC all' uscita del while comunichi il numero di tentativi effettuati.

12. Scrivere un programma che richieda dei numeri interi all'utente e calcoli la somma di quelli negativi e la somma di quelli positivi. Il programma deve interrompersi se la somma dei negativi è <= a -20 oppure se la somma dei positivi è > di 20. Il programma deve comunicare se il ciclo si è interrotto per esubero dei numeri positivi o di quelli negativi.

PROGRAMMAZIONE IN LINGUAGGIO C++

ARRAY AD UN DIMENSIONE

ARRAY AD UNA DIMENSIONE

Un array ad una dimensione è una lista di variabili che hanno tutte lo stesso nome e tipo, ma che si distinguono in base ad un indice intero.

Per definire un array ad esempio formato da 5 variabili intere

andrò ad indicarlo come:

int a[5]; le singole variabili saranno identificate con:

a[0], a[1], a[2], a[3], a[4]

fare attenzione il primo indice vale 0.

idem se l'array è del tipo float

float v[5]; le singole saranno verranno identificate con:

v[0], v[1], v[2], v[3], v[4]

fare attenzione il primo indice vale 0.

Istruzione di assegnazione dell'intero vettore:

Esempio: dato un array intero di 5 variabili, memorizzare i primi 5 numeri naturali.

int v[5]={0,1,2,3,4};

Istruzione di assegnazione per singola variabile:

int v[0]=0, v[1]=1, v[2]=2, v[3]=3, v[4]=4;

1. Memorizzare in un array di dieci posizioni i primi dieci numeri naturali.
2. L'utente inserisce valori interi in un array di 10 elementi e il programma: stampa l'array, calcola la somma e la media dei valori.
3. Leggere e memorizzare in un array di 8 celle, 8 numeri reali randomici da 0 a 10, dopo averli memorizzati calcolare la somma e la media.
4. Leggere e memorizzare in un array 7 numeri, dopo averli memorizzati contare quante volte è stato memorizzato lo zero.
5. Memorizzare in un array di 100 posti i primi cento numeri naturali in ordine inverso.
6. Dopo aver letto e memorizzato 8 numeri in un array, calcolare la somma di quelli negativi e memorizzare zero al loro posto. Visualizzare l'array e la somma.
7. Scrivere un programma in C++ che inverta la posizione degli elementi di un vettore di interi di cui sia fornita la dimensione e il valore di ogni elemento.
8. Inserire in un array 100 numeri casuali compresi tra 0 e 18, visualizzando 10 numeri per riga.
9. Inserimento random di valori float tra 0 e 10 in un array di 50 elementi e visualizzare l'array.
10. Creare un array di interi con 10 posti, inserire zeri in tutte le celle; leggere in che posizione inserire un 1 e visualizzare l'array.
11. Riempire un array con 5 elementi inseriti dall'utente che siano maggiori di zero, se viene inserito un numero negativo viene fatto reinserire fino a quando rispetta la condizione. Scrivere a schermo l'array inserito.
12. Leggere un array di interi di 5 posizioni e dopo averlo letto cercare al suo interno il valore massimo e scrivere il valore e la posizione della casella in cui è memorizzato.
13. Leggere un array di interi random compresi tra 0 e 10, di 6 elementi, leggere un ulteriore numero intero copreso tra 0 e 10 inclusi e dire quanti numeri memorizzati nell' array sono inferiori, quanti superiori e quanti uguali all'ultimo numero letto.
14. Leggere un array di interi random compresi tra 0 e 50 di 10 posizioni.
Visualizzare in quali posizioni è memorizzato un numero pari. Escludendo lo zero che per C++ è pari.
15. I codici a barre dei prodotti sono composti da 13 cifre, comprese tra 0 e 9 inclusi, di cui l'ultima è la cifra di controllo che si determina a

partire dalle prime 12 con le seguenti regole:
moltiplicare per 3 tutte le cifre in posizione dispari (la prima, la terza,...fino all'undicesima);

moltiplicare per 2 tutte le cifre in posizione pari (la zero, la seconda, la quarta...fino alla dodicesima);
sommare i 12 valori così ottenuti prendere il resto della divisione, della somma per 10 ed infine visualizzare l'array con il valore della 13ma posizione.

I primi 12 valori sono casuali e il tredicesimo è inizializzato a zero.

16. Leggere un array di interi randomici compresi tra 0 e 3 di 10 posizioni.

Stampare il valore che compare più volte all'interno dell'array, qualora ci siano più numeri che compaiono lo stesso numero di volte stampare quello con valore minore.

17. Leggere un array di 10 interi random compresi tra 1 e 5 inclusi, stampare solo i numeri che appaiono nell' array una volta soltanto.

18. Riempire un array di 10 caselle con valori casuali compresi tra 0 e 30 e ordinarlo in ordine crescente utilizzando l'algoritmo di ordinamento Selection Sort.

PROGRAMMAZIONE IN LINGUAGGIO C++

LE MATRICI

LE MATRICI

Una matrice a due dimensioni è una lista di variabili tutte dello stesso tipo ordinate per righe e colonne che si distinguono in base a due indici interi.

Per definire una matrice ad esempio formata da 10 variabili intere ordinate su 2 righe e 5 colonne andrà ad indicarla come segue:

int mat[2][5]; le singole variabili saranno identificate con:

mat[0][0], mat[0][1], mat[0][2], mat[0][3], mat[0][4],

mat[1][0], mat[1][1], mat[1][2], mat[1][3], mat[1][4],

Istruzione di assegnazione dell'intera matrice:

Esempio: data una matrice intera di 10 variabili, memorizzare i primi 10 numeri naturali.

int mat [2],[5]= {0,1,2,3,4,5,6,7,8,9};

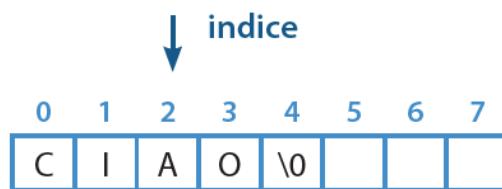
Istruzione di assegnazione della singola variabile:

int mat[0][0]=0, mat[0][1]=1, mat[0][2]=2, mat[0][3]=3,
mat[0][4]=4, mat[1][0]=5, mat[1][1]=6, mat[1][2]=7, mat[1][3]=8, mat[1][4]=9;

1. Memorizzare in una matrice di 5 righe e 4 colonne i primi venti numeri naturali.
2. Scrivere un programma che permetta all'utente di inserire dei numeri interi in una matrice 2x2, e visualizzare la matrice.
3. Scrivere un programma che trovi il numero massimo all'interno di una matrice 3x3 di numeri positivi casuali da 0 a 24.
4. Scrivere un programma che calcoli la somma delle righe di una matrice 2x3 di numeri interi positivi casuali da 0 a 9.

ARGOMENTO:	PROGRAMMAZIONE IN LINGUAGGIO C++
TITOLO:	LE STRINGHE
TEORIA:	<p>LE STRINGHE</p> <p>Una stringa è una variabile che raggruppa una sequenza di caratteri, pertanto è uno speciale vettore di caratteri.</p> <p>Nel complesso una stringa può accogliere singole parole o intere frasi.</p> <p>A differenza del vettore è conclusa dal carattere speciale '\0' che ne delimita la fine, detto carattere terminatore o NULL.</p> <p>Grazie al carattere terminatore è possibile:</p> <ol style="list-style-type: none">1. delimitare la fine della stringa in tutte le situazioni, senza alcuna ambiguità;2. dichiarare una sola volta la stringa con una lunghezza prestabilita, ma memorizzare sequenze di caratteri più corte della effettiva lunghezza della stringa. <p>La stringa, come il vettore, può essere rappresentata visivamente da un insieme di caselle, numerate a partire da 0.</p> <p>Nella figura sotto è rappresentata una stringa di 8 caselle così caratterizzata:</p> <ol style="list-style-type: none">1. le caselle sono numerate da 0 a 7;2. ogni casella è identificata da un indice, un numero che esprime la sua posizione d'ordine;3. per esempio la terza casella è individuata da INDICE = 2;

- il suo contenuto è la lettera 'A';
4. la stringa è conclusa dal carattere terminatore;
 5. le caselle 5, 6, 7 sono vuote oppure, se hanno un contenuto, questo è ininfluente.



La dichiarazione di una stringa prevede di specificare:

1. il tipo della variabile, che è un **char**;
2. il nome della variabile;
3. l'estensione, cioè il numero di caselle.

ogni singola casella memorizza un carattere alfanumerico al quale si assegnano 8bit di memoria.

Esempio:

Dichiara una stringa di nome PAROLA, costituita da 8 caselle.

char PAROLA[8]

- **Lettura/scrittura di ogni singola casella:**

Nella modalità lettura/scrittura di ogni singola casella l'accesso alla casella avviene, esattamente come in un vettore, specificando il nome della stringa con l'indice racchiuso tra parentesi quadre.

Esempio:

Assegna la lettera 'O' alla casella 3 di PAROLA.

PAROLA[3] = 'O';

	<p>Esempio:</p> <p>Assegna a CAR (variabile di tipo char) il contenuto della casella numero 2.</p> <p>CAR = PAROLA[2];</p> <p>ESEMPIO</p> <ul style="list-style-type: none">• Lettura/scrittura dell'intera stringa. <p>Assegnare direttamente i caratteri alle caselle.</p> <p>Il carattere terminatore deve essere giustapposto alla fine.</p> <p>char PAROLA[8] = {'C','I','A','O','\0'};</p> <p>Assegnare l'intera stringa.</p> <p>Il carattere terminatore viene inserito automaticamente alla fine della stringa.</p> <p>char PAROLA[8] = "CIAO";</p> <p>La scrittura a video può essere fatta impiegando un ciclo for.</p> <pre>for(i=0;i<=8;i=i+1){ cout<<PAROLA[i]}</pre>
ESERCIZI	<ol style="list-style-type: none">1. Leggi una stringa e determina quanto è lunga.2. Leggi una stringa e un carattere e conta quante volte quel carattere è contenuto nella stringa.3. Leggi una stringa e determina quante vocali contiene.4. Leggi due stringhe e verifica quale è più lunga.

--	--

ARGOMENTO:	PROGRAMMAZIONE IN LINGUAGGIO C++
TITOLO:	LE FUNZIONI
TEORIA:	<p>LE FUNZIONI</p> <p>Volendo definire una funzione da usare nel programma principale procediamo nel seguente modo:</p> <p>anteporre al main il prototipo della funzione che vogliamo definire che avrà la seguente struttura:</p> <p>tipo della funzione nome della funzione (tipo degli argomenti)</p> <p>posporre al main al main la testata della funzione che avrà la seguente struttura:</p> <p>tipo della funzione nome della funzione (tipo e nome degli argomenti) seguita dal corpo della funzione che vogliamo definire racchiuso tra parentesi graffe.</p>

ESERCIZI	<ol style="list-style-type: none">1. L'utente inserisce due numeri interi e il programma tramite una funzione trova il minore e lo stampa.2. L'utente inserisce quanti voti vuole inserire. Il programma tramite due funzioni trovamin e trovamax stampa il valore del voto minimo e del voto massimo inserito.3. Scrivere una funzione che accetti due numeri Num1 e Num2 e un simbolo '+', '-' , '*' , '/'. La funzione restituisce il risultato dell'operazione indicata dal simbolo.4. Scrivere una funzione che accetti due lati Lato1 e Lato2 e restituisca un carattere: 'q' se i due lati sono uguali (quadrato) 'r' se i due lati sono diversi (rettangolo). <p>Il programma principale deve richiamare la funzione e stampare se si tratta di un quadrato o di un rettangolo.</p>

Software: CADESIMU 4962 e PCSIMU 9966

Ciclo automatico A+, A- con valvola bistabile.

- 1) realizzazione della configurazione hardware: START, STOP, a0, a1
 - IO inserisco il plc S7-1200 alla metà dell'altezza del foglio
 - inserisco l'alimentazione del PLC (24V,0V e massa) andando nei connettori.

CORSO DI SOLIDWORKS ELECTRICAL**CREARE UN NUOVO PROGETTO**

- Doppioclick icona presente su desktop “SolidWorks Electrical” si entra nella finestra “Gestione dei progetti Elettrici”.
- Click su Nuovo.
- Scegliamo il modello IEC “simboli secondo gli standard della “Commissione Elettrotecnica Internazionale”, e la lingua Italiano.
- Si procede alla compilazione (alcuni campi presenti verranno richiamati nei dati presenti nei cartigli).
- Compilare i campi:

Titolo: Formazione 2024/25

numero contratto: 0001

Cliente:

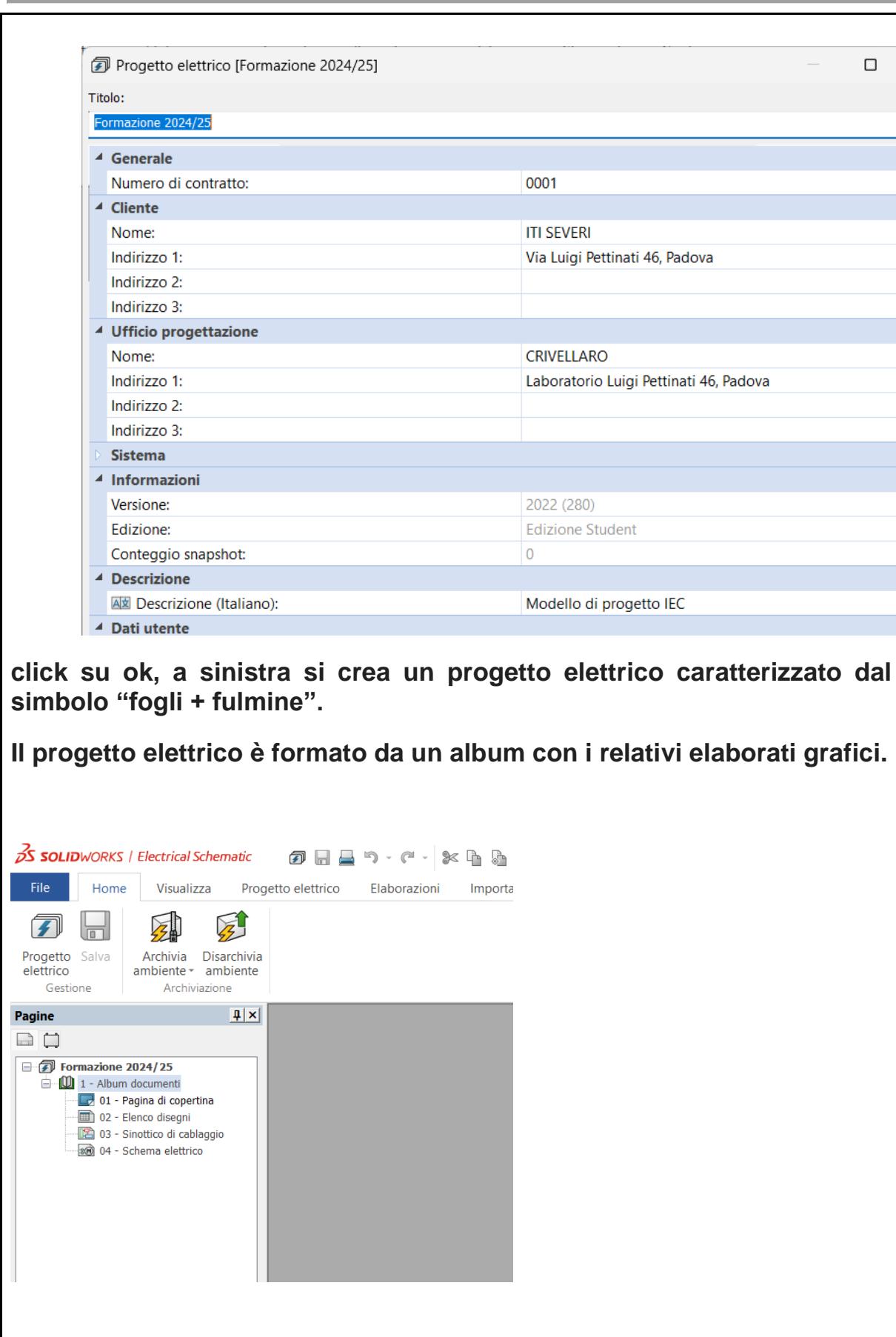
nome: ITI SEVERI

indirizzo: Via Luigi Pettinati 46, Padova

Ufficio Progettazione:

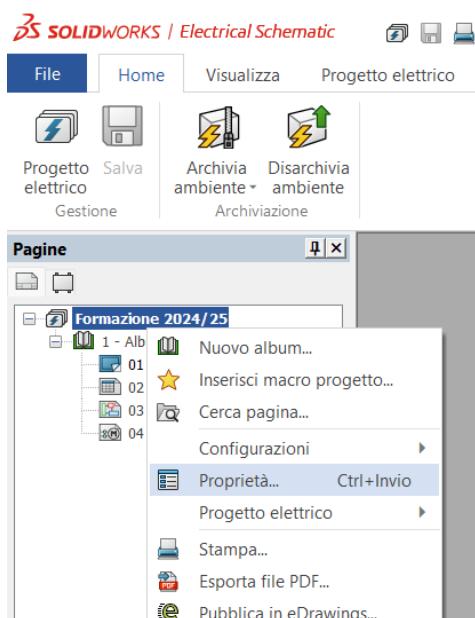
Nome: Crivellaro Andrea

indirizzo: Laboratorio Luigi Pettinati 46, Padova



Se volessi tornare a modificare o integrare informazioni esempio modifica indirizzo cliente, per tornare alla finestra:

Tasto destro mouse su progetto elettrico selezionare proprietà.



Come si vede per adesso il progetto è formato da:

- **pagina di copertina**
- **elenco dei disegni**
- **sinottico**
- **schema elettrico**

Facendo doppio click si apre la relativa tavola grafica.

Si osserva che alcuni dati “proprietà progetto” sono stati impiegati per compilare i cartigli. Ci sono altri campi che riportano proprietà riferite alla singola tavola (revisione e modifiche) o ad esempio alla data di creazione.

CORSO DI SOLIDWORKS ELECTRICAL

GESTIONE CARTIGLI

**Per conoscere quali cartigli sono associati alle tavole grafiche, tasto destro sul progetto “Formazione 2024/25” → configurazioni → progetto elettrico.
Spostarsi tra le schede e selezionare Cartiglio.**

The screenshot shows the 'Configurazione progetto elettrico: Formazione 2024/25' window. At the top, there is a toolbar with icons for Generale, Grafico, Simbolo, Attributo, Testo, Contrassegno, Cartiglio, and Libreria e palette. Below the toolbar is a table titled 'Cartiglio' with three columns: 'Tipo di disegno', 'Cartiglio associato', and 'Descrizione'. The table lists various drawing types and their associated cartiglio names and descriptions.

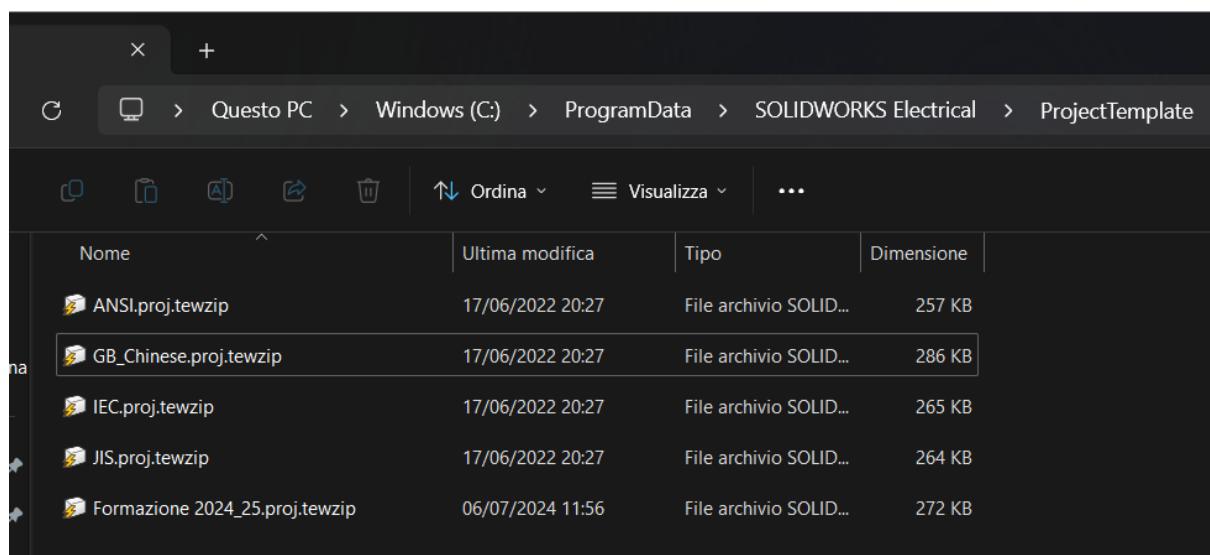
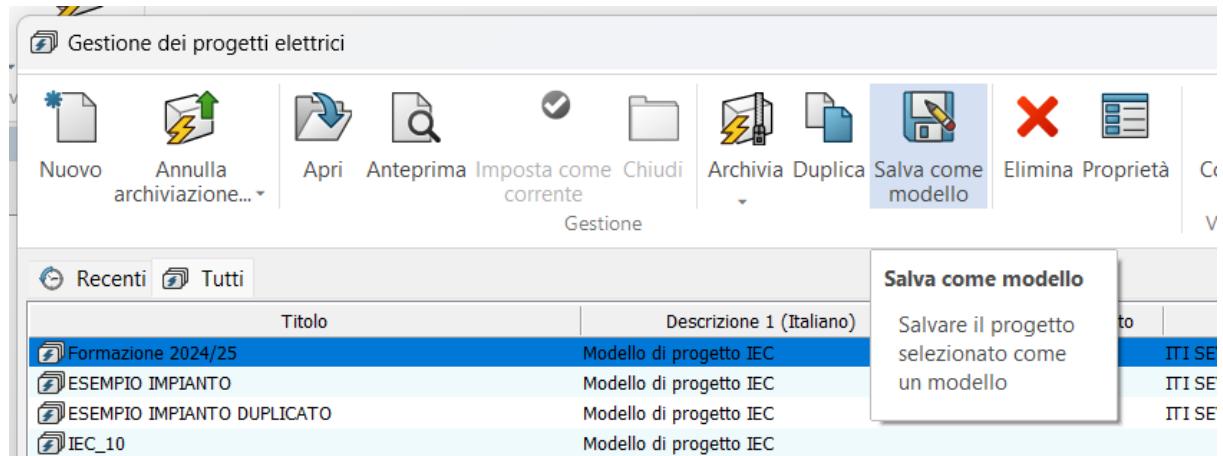
Tipo di disegno	Cartiglio associato	Descrizione
Pagina di copertina:	TR_FDP_BASEA3_PDG_EN	Pagina di copertina (titoli in inglese)
Schema:	TR_FDP_BASEA3CA_EN	10 colonne (titoli in inglese)
Sinottico:	TR_FDP_BASEA3CA_EN	10 colonne (titoli in inglese)
Schema misto:	TR_FDP_BASEA3CA_EN	10 colonne (titoli in inglese)
Disposizione ad armadio 2D:	TR_FDP_BASEA3_SCALE_EN	Con scala (titoli in inglese)
Disegno 2D da 3D:	TR_FDP_BASEA3_SCALE_EN	Con scala (titoli in inglese)
Appiattimento routing:	TR_FDP_BASEA3_SCALE_EN	Con scala (titoli in inglese)
Disegno della morsettiera:	TR_FDP_BASEA3_EN	Senza colonne (titoli in inglese)
Rapporto:	TR_FDP_BASEA3_EN	Senza colonne (titoli in inglese)
Regola di progettazione:	TR_FDP_BASEA3_EN	Senza colonne (titoli in inglese)

Possiamo controllare l’associazione, tasto destro sulla singola tavola grafica→ cartiglio→ apri.

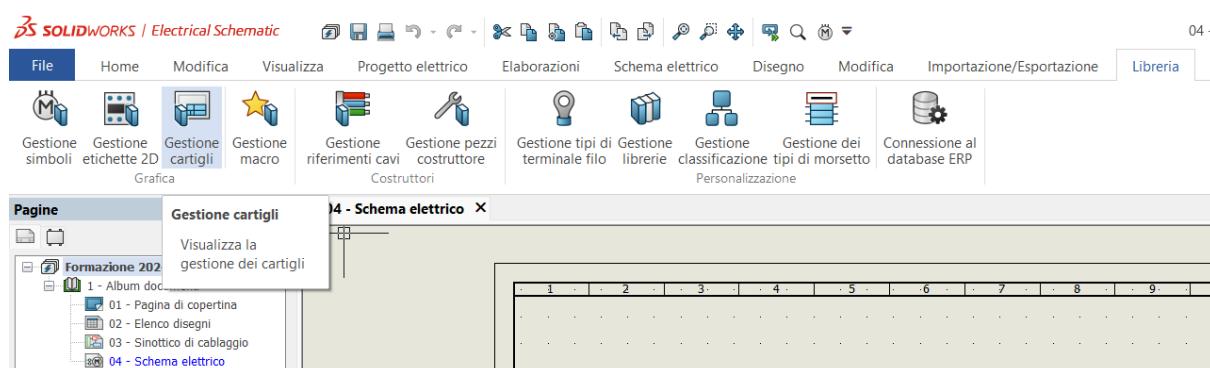
The screenshot shows the 'Pagine' (Pages) window with a tree view of project documents. A schematic page titled '04 - Schema elettrico' is selected. A context menu is open over the page, with the 'Cartiglio' option highlighted. A secondary submenu for 'Cartiglio' is open, showing options like 'Aggiorna...', 'Sostituisci...', 'Rimuovi...', and 'Apri...'. The main menu bar at the top includes 'File', 'Edita', 'Progetto', 'Disegni', 'Strumenti', 'Visualizza', 'Opzioni', 'Aiuto', and 'SolidWorks'.

Una volta settato il tutto coerentemente al disegno posso creare un modello, per poter riutilizzare le impostazioni in progetti futuri.

Uscire dal progetto e selezionare Salva come modello.



Per creare un cartiglio, devo selezionare Libreria → gestione cartigli



Esercizio: duplicare di un cartiglio in formato ISO 216 - A3 - 420x297 mm.

Duplico un cartiglio esistente per poi modificarlo. Selezionare 10 colonne (titolo in inglese) → copiare e incollare → modificare nome. Tasto destro del mouse sul cartiglio copiato → proprietà e modifico la descrizione (10 colonne) titolo in inglese prova). Entrare nella tendina del progetto, associare allo schema elettrico il nuovo cartiglio di prova.

Dalla Libreria facendo doppio click Aprire il cartiglio di prova.

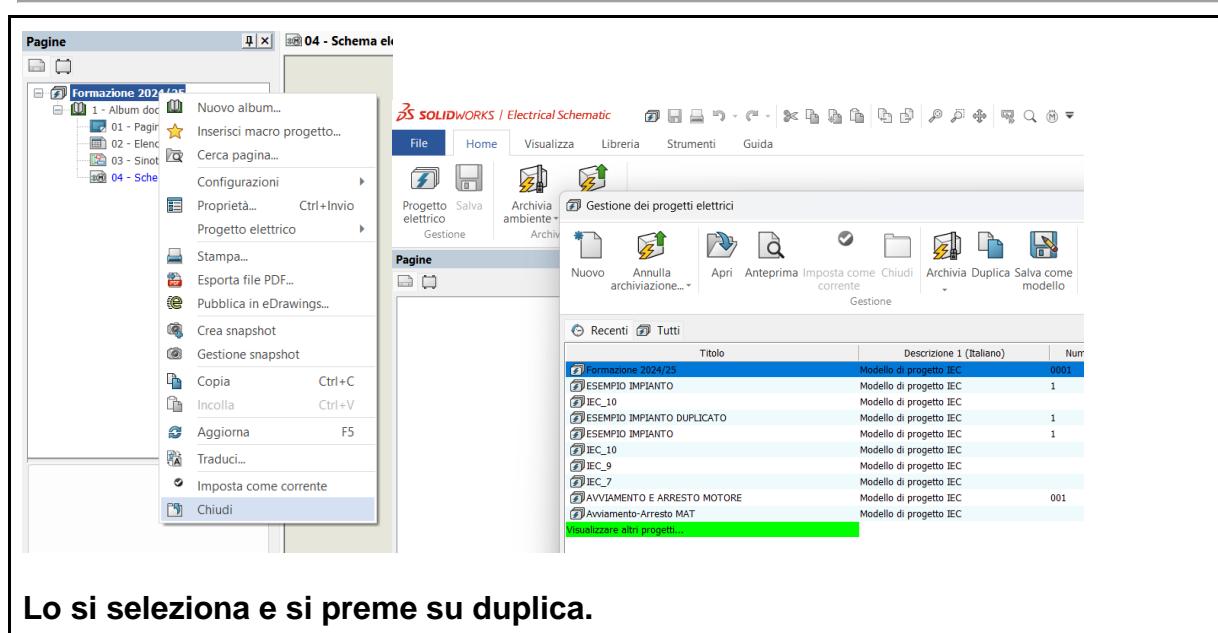
Se osservo il cartiglio vedo delle scritte con anteposto il simbolo #, questi si chiamano attributi.

Per capire a quale proprietà si riferiscono devo aprire la tendina di progetto alla sinistra. Quelli colorati in blu sono già inseriti e possono riferirsi all'intero progetto, all'album o al singolo disegno. Per inserire una nuova proprietà, trascinare quelle in nero all'interno dell'area del cartiglio.

CORSO DI SOLIDWORKS ELECTRICAL

DUPLICARE UN PROGETTO

Prima di duplicarlo, chiudere il progetto



Lo si seleziona e si preme su duplica.

CORSO DI SOLIDWORKS ELECTRICAL

SINOTTICO DI CABLAGGIO

Il sinottico di cablaggio ci serve per rappresentare i percorsi esterni ai quadri dei cavi. Nel sinottico è importante delineare i quadri, le apparecchiature esterne, i dispositivi di manovra che realizzano collegamenti esterni tramite cavi.

DEFINIZIONE DELLE UBICAZIONI

La prima fase consiste nel definire le ubicazioni:

Componenti

Formazione 2024/25

- Nuovo
- Cerca componente...
- Configurazioni
- Proprietà... Ctrl+Invio
- Progetto elettrico
- Crea snapshot
- Gestione snapshot
- Visualizzazione
- Taglia Ctrl+X
- Copia Ctrl+C
- Incolla Ctrl+V
- Aggiorna F5
- Traduci...
- Imposta come corrente**
- Chiudi

Proprietà ubicazione: +L1

Proprietà Pezzo costruttore

Contrassegno	
Modalità:	<input checked="" type="radio"/> Automatico <input type="radio"/> Manuale
Contrassegno:	L1
Radice:	L
Numero:	1
Gerarchia	
Principale:	
Album associato:	1 - Album documenti
Descrizione	
Descrizione (Italiano):	QUADRO PRINCIPALE
Dati utente	
Dati utente 1:	
Dati utente 2:	
Dati traducibili	
Dati traducibili 1 (Italiano):	
Dati traducibili 2 (Italiano):	

Componenti

- Formazione 2024/25
 - L1 - QUADRO PRINCIPALE
 - L2 - PANNELLO DI CONTROLLO
 - L3 - SALA MACCHINE

INSERIRE I SIMBOLI:

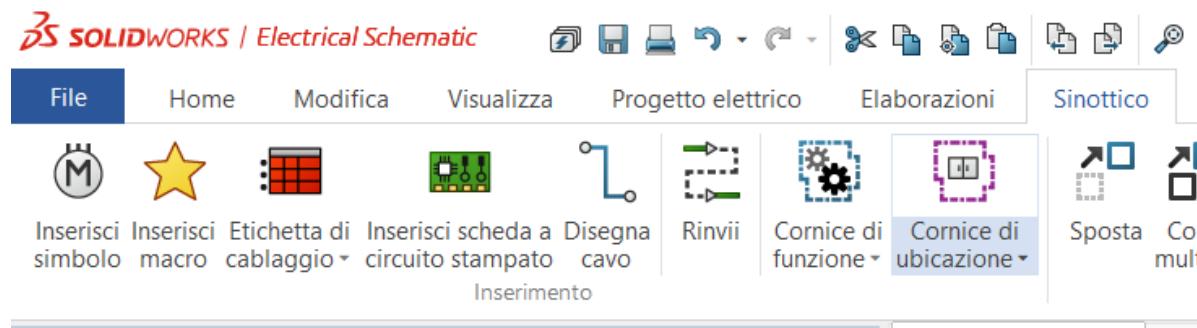
Nella scheda sinottico, selezionare inserisci simbolo:

INIZIAMO DAL MOTORE: quando lo inseriamo modifichiamo la posizione e lo inseriamo nell'ubicazione precedentemente definita SALA MACCHINE.

INSERIAMO LA LAMPADA DI SEGNALAZIONE ROSSA H1 e VERDE H2 modifichiamo la posizione in L2 PANNELLO DI CONTROLLO.

INSERIAMO LE MORSETTIERE: generalmente nel quadro principale distinguono di due circuiti, quindi una morsettiera per il circuito di potenza verso i motori e una per quello di comando verso il pannello di controllo. Il pannello di controllo ha una sua morsettiera.

CORNICI DI UBICAZIONE: Contornare i componenti con le cornici di ubicazione.

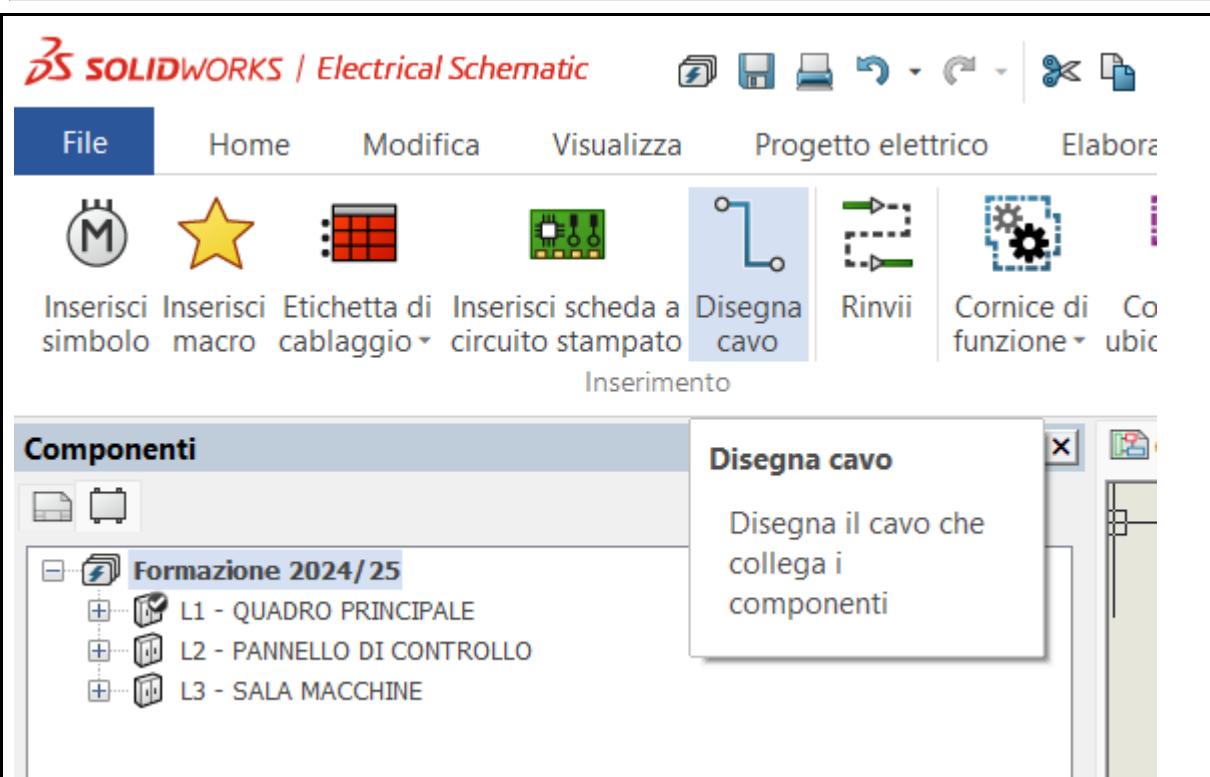


DISEGNA COLLEGAMENTI TRA UBICAZIONI:

creare dei collegamenti tra le diverse zone realizzando uno schema UNIFILARE.

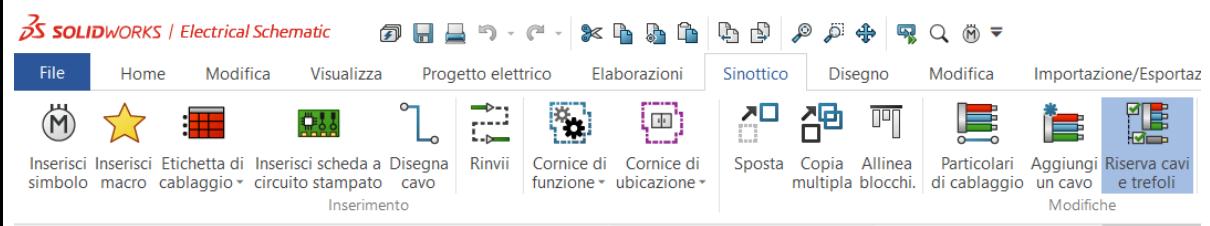
SCHEDA SINOTTICO: Selezionare Disegna Cavo, differenziare gli stili se sono cavi differenti.

Nell'esempio faremo lo stile F01 rosso per il circuito di potenza, e F03 blu per quello di comando.



DEFINIZIONE DEL TIPO DI CAVO E ASSOCIAZIONE:

**Scheda SINOTTICO → Selezionare Riserva Cavi e trefoli e poi selezionare sul disegno il percorso disegnato a cui assegnare un tipo di cavo.
Click su nuovo cavo si apre il database, selezionarne uno.**

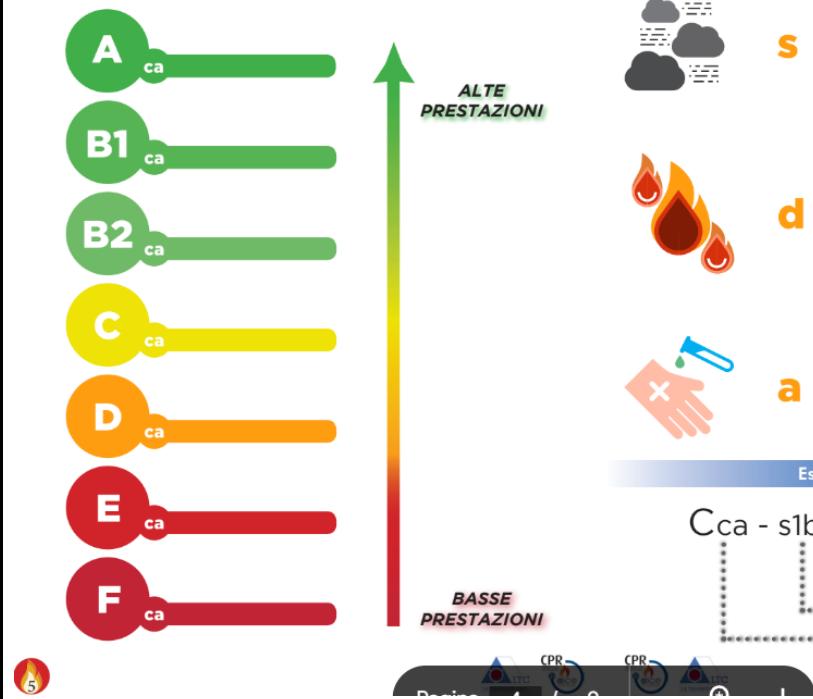


CREAZIONE DI UN TIPO DI CAVO:

In base alla normativa Il Regolamento Prodotti da Costruzione (CPR), devono essere sicuri nei confronti della resistenza all'incendio.

CPR**Classificazione CPR**

I cavi sono stati classificati in 7 classi di Reazione al Fuoco identificate dalle lettere da «F» a «A» e dal pedice "ca" (cable) in funzione delle loro prestazioni crescenti

**CPR****Parametri addizionali CPR**

Oltre a questa classificazione principale, le Autorità Europee hanno regolamentato anche l'uso dei seguenti **parametri addizionali**:



S Opacità dei fumi
(s1 – s2 – s3 / s1a – s1b)



d Gocciolamento di particelle incandescenti
(d0 – d1 – d2)



a Acidità che definisce la pericolosità dei gas e fumi per le persone e la corrosività per le cose
(a1 – a2 – a3)

Esempio di classificazione

C_{ca} - s1b, d1, a1



Acidità = a1

Gocciolamento = d1

Opacità = s1b

Reazione al fuoco = C_{ca}

**CPR****Tabella di correlazione**

LUOGHI DI IMPIEGO (EDIFICI ED ALTRE OPERE DI INGEGNERIA CIVILE)	LIVELLO DI RISCHIO
• AEREOSTAZIONI • STAZIONI FERROVIARIE • STAZIONI MARITTIMI • METROPOLITANE in tutto o in parte soffitte e soffitte • GALLERIE STRADALI di lunghezza superiore ai 500m • FERROVIE superiori a 1000m	ALTO
• STRUTTURE SANITARIE che erogano prestazioni in regime di ricovero ospedaliero e/o residenziale a ciclo continuativo e/o diurno • CASE DI RIPOSO per anziani con oltre 25 posti letto • STRUTTURE SANITARIE che erogano prestazioni di assistenza specialistica in regime ambulatoriale, ivi comprese quelle riabilitative, di diagnostica strumentale e di laboratorio • LOCALI DI SPETTACOLO E DI INTRATTENIMENTO in genere impianti e centri sportivi, palestre, sia di carattere pubblico che privato • ALBERghi • PENSIONI • MOTEL • VILLAGGI ALBERGO • RESIDENZE TURISTICO-ALBERGHIERE • STUDENTATI • VILLAGGI TURISTICI • ALLOGGI AGRITURISTICI • OSTELLi per la giovinezza • RIFUGI ALPINI • BED & BREAKFAST • DORMITORI • CASE PER FERIE con oltre 25 posti letto • STRUTTURE TURISTICO-RICETTIVE nell'aria aperta (campeggi, villaggi turistici, ecc.) con capacità ricettiva superiore a 400 persone • SCUOLE di ogni ordine, grado e tipo, collegi, accademie con oltre 100 persone presenti • ASILI NIDO con oltre 30 persone presenti • LOCALI adibiti ad esposizione e/o vendita all'ingrosso o al dettaglio, fiere e quartieri fieristici • AZIENDE ED UFFICI con oltre 300 persone presenti • BIBLIOTECHE • ARCHIVI • MUSEI • GALLERIE • ESPOSIZIONI • MOSTRE • EDIFICI destinati ad uso civile, con altezza antincendio superiore a 24m	MEDIO
• EDIFICI destinati ad uso civile ed industriale, con altezza antincendio inferiore a 24m • SALE D'ATTESA • BAR • RISTORANTI • STUDI MEDICI	BASSO (posa a fascio)
• ALTRE ATTIVITÀ: installazioni non previste negli edifici di cui sopra e dove non esiste rischio di incendio e pericolo per persone r/o cose	BASSO (posa singola)

CPR**Tabella di correlazione**

DESIGNAZIONE ATTUALE	DESIGNAZIONE CPR	CLASSE DI PRESTAZIONE
FG10OM1- 0,6/1 kV	FG18OM16 - 0,6/1 kV	B2 _{ca} -s1a, d1, a1
FG7OM1 - 0,6/1 kV N07G9-K (H07Z1-K/U/R type 2)	FG16OM16 - 0,6/1 kV FG17 - 450/750 V (H07Z1-K/U/R type 2)	C _{ca} -s1b, d1, a1
FG7OR - 0,6/1 kV N07V-K	FG16OR16 - 0,6/1 kV FS17 - 450/750 V	C _{ca} -s3, d1, a3
H07RN-F	H07RN-F	E _{ca}

Il rischio incendio nel caso di applicazioni industriali è BASSO utilizzeremo cavi del tipo FG16OR-06/1kV o FS17-450/750V solo unipolare.

Il cavo FS17-450/750V è indicato per i collegamenti interni ai quadri in quanto

unipolare, mentre il cavo FG16OR-06/1kV è indicato per eseguire i collegamenti tra quadretti collocati in differenti ubicazioni.

Cavo che collega quadro principale al motore:

SCELTA DEL MOTORE e DETERMINAZIONE CAVO

Dati: • Potenza meccanica richiesta dal carico $P_m = 5 \text{ kW}$ •

Velocità di rotazione massima richiesta dal carico $n_m = 1400 \text{ rpm}$

Consultando le tabelle del costruttore:

Da tabella 2 si sceglie il motore tipo 132 S che presenta le seguenti grandezze caratteristiche:

Tensione nominale $V = 400 \text{ V}$ trifase

Potenza nominale (meccanica) $P_n = 5,5 \text{ kW}$

Velocità nominale $n = 1450 \text{ rpm}$

Rendimento $\eta = 84\% (0,84)$

Fattore di potenza $\cos \phi = 0,85$

Corrente di spunto $I_s = 7$

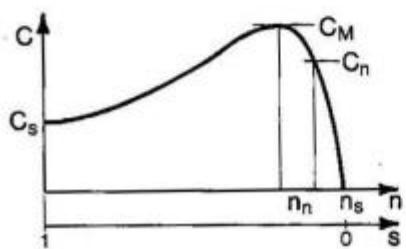
Potenza elettrica $P_e = P_n / \eta = 5,5 / 0,84 = 6,55 \text{ kW} = 6550 \text{ W}$ •

Corrente nominale $I_n = P_e / (1,73 * V * \cos \phi) = 6550 / (1,73 * 400 * 0,85) = 11,1 \text{ A}$

Grand. costrutt.	Potenza nominale (kW)	Potenza nominale (CV)	Velocità nominale (min ⁻¹)	Rendimento (%)	Fattore di poten.	Corrente nomin. 380 V (A)	Coppia nomin. (N · m)	Coppia spunto quale multiplo della coppia nomin.	Corrente spunto corrente nomin.	Coppia max. coppia nomin.	Peso (kg)
4 POLI											
100 M	2,2	3	1415	79	0,82	5,2	15	2,2	6,0	2,6	22
100 L	3	4	1415	81	0,83	6,8	20	2,7	6,2	3,0	24
112 M	4	5,5	1435	83	0,83	8,8	27	2,8	7,0	3,0	42
132 S	5,5	7,5	1450	84	0,85	11,7	36	2,2	7,0	2,9	50
132 M	7,5	10	1450	86	0,85	15,6	49	2,4	7,6	3,3	66
160 M	11	15	1460	88	0,86	22	72	2,4	7,6	3,0	92
160 L	15	20	1460	89	0,88	29	98	2,2	7,7	2,9	
180 M	18,5	35	1455	90,2	0,83	37,5	121	2,6	6,4	2,5	170
180 L	22	30	1460	91,2	0,83	44	144				190
200	30	40	1465	91,7	0,83	60	195	2,6	6,4	2,5	250

Motori asincroni trifasi**Grandezze caratteristiche dei motori ad induzione**

Denominazione	Unità di misura	Relazioni
Tensione nominale U_n	V	
Corrente nominale I_n	A	$I_n = \frac{P_n}{\sqrt{3} \cdot U_n \cdot \cos\varphi \cdot \eta}$
Frequenza f	Hz	
Fattore di potenza $\cos\varphi$		$\cos\varphi = P_a / (\sqrt{3} \cdot U_n \cdot I)$
Potenza assorbita P_a	W	$P_a = K \cdot U_n \cdot I \cdot \cos\varphi$ ($K = \sqrt{3}$ per motori trifasi; $K = 1$ per motori monofasi)
Potenza nominale P_n (resa all'albero)	W	$P_n = P_a \cdot \eta$ (η = rendimento)
Coppia $\begin{cases} \text{nominale } C_n \\ \text{di spunto } C_s \\ \text{massima } C_M \end{cases}$	N · m	$C_n = \frac{9,55 \cdot P_n}{n_n}$
Velocità $\begin{cases} \text{di sincronismo } n_s \\ \text{nominale } n_n \\ \text{effettiva } n \end{cases}$	N · m giri al minuto	$n_s = 120 f/p$ (p = numero poli) $n = (1 - s) \cdot n_s$ (s = scorrimento)

Caratteristica meccanica

La coppia risulta massima a circa $(0,8 \div 0,9) n_s$.

La potenza resa diminuisce all'aumentare della temperatura ambiente.

La coppia influisce anche sulle dimensioni del motore, maggiore è la coppia richiesta maggiori sono le dimensioni del motore a parità di potenza resa.

Il cavo per sopportare dovrà avere un valore nominale di corrente maggiore di quella assorbita dal motore.

Entrando nella tabella del costruttore dei cavi: FG16OR-06/1kV

condizione di posa: tubo in aria a 30°

cavo quadripolare 4G (G indica che ci sono 3 conduttori di linea + gialloverde).

Colori delle anime

UNIPOLARE



BIPOLARE



TRIPOLARE



QUADRIPOLARE



PENTAPOLARE



Quadripolari

Formazione	\varnothing indicativo conduttore	Spessore medio isolante	Spessore medio guaina	\varnothing esterno max	Resistenza elettrica max a 20°C	Peso indicativo cavo	Portata di corrente A					
							in aria a 30°C	in tubo in aria a 30°C	interrato a 20°C	tubo interrato a 20°C		
n° x mm ²	mm	mm	mm	mm	Ω/km	kg/km	K = 1	K = 1,5	K = 1	K = 1,5		
4 x 1,5	1,5	0,7	1,8	13,4	13,3	170	23	19,5	23	22	20	19
4 x 2,5	2,0	0,7	1,8	14,6	7,98	220	32	26	30	29	27	25
4 x 4	2,5	0,7	1,8	16,0	4,95	295	42	35	39	37	34	32
4 x 6	3,0	0,7	1,8	17,5	3,30	385	54	44	50	47	43	41
4 x 10	4,0	0,7	1,8	19,8	1,91	575	75	60	67	63	58	55
4 x 16	5,0	0,7	1,8	22,4	1,21	795	100	80	88	83	76	72
4 x 25	6,2	0,9	1,8	26,8	0,780	1205	127	105	113	107	99	93

Scelgo un cavo 4Gx1.5mmq con In=19.5 A

AGGIUNGERE UNA TIPOLOGIA DI CAVI:

LIBRERIA → GESTIONE RIFERIMENTI CAVI

I cavi che utilizzeremo più spesso li associamo ad una nuova classe.

**SELEZIONO GESTIONE CLASSIFICAZIONE → NUOVA CLASSE
assegno il nome PERSONALIZZATA**

RITORNARE IN GESTIONE RIFERIMENTO CAVO → NUOVO RIFERIMENTO

Nuovo riferimento del cavo

Proprietà **Dati utente** **Trefoli cavo**

Generale

Riferimento:	
Costruttore:	
Classe	PERSONALIZZATA
Numero di articolo:	
ID esterno:	
Libreria:	<Nessuna libreria>
Famiglia:	
Norma:	
Serie:	
Descrizione (Italiano):	<input type="button" value="+"/>

Fornitore

Nome fornitore:	
Numero stock:	

Informazioni

Autore creazione:	Utente
Creato da:	crive
Data di creazione:	07/07/2024
Modificato da:	crive
Data di modifica:	07/07/2024

Personalizza... **OK** **Annulla**

Proprietà riferimento cavo

Proprietà **Dati utente** **Trefoli cavo**

Generale

Riferimento:	FG16OR16 4Gx1.5
Costruttore:	TRIVENETA CAVI
Classe	PERSONALIZZATA
Numero di articolo:	
ID esterno:	
Libreria:	<Nessuna libreria>
Famiglia:	
Norma:	CPR
Serie:	
Descrizione (Italiano):	<input type="button" value="+"/>

Fornitore

Nome fornitore:	
Numero stock:	

Informazioni

Autore creazione:	Utente
Creato da:	crive
Data di creazione:	07/07/2024
Modificato da:	crive
Data di modifica:	07/07/2024

Caratteristiche

Tipo:	Energia
Standard dimensioni:	Sezione (mm ²)
Sezione conduttori (mm ²):	1.5
Lunghezza:	0 m
Diametro:	13.4 mm
Diametro conduttore:	1.5 mm
Colore:	Grigio
Fattore raggio di curvatura:	0
Raggio di curvatura (fattore raggio di curvatura x diametro):	0 mm
Massa lineare:	
Caduta di tensione (V/A/km):	0

Nuovo riferimento del cavo

Proprietà **Dati utente** **Trefoli cavo**

Aggiungi **Inserisci** **Elimina** **Raggruppa** **Nulla raggruppame**

Numero	Descrizione (Italiano)	Tipo	Colore	Sezione (mm ²)	Diametro (mm)	Gruppo
1	1	„ Vari	Marrone	1.5	1,5	„
2	2	„ Vari	Nero	1.5	1,5	„
3	3	„ Vari	Grigio	1.5	1,5	„
4	4	„ Protezione	Giallo Verde	1.5	1,5	„

TORNO AL SINOTTICO→RISERVA CAVI E TREFOLI→ SELEZIONO IL CAVO CHE COLLEGA IL QUADRO PRINCIPALE AL MOTORE E ASSOCIO IL CAVO CREATO.

Nello stesso modo creare i seguenti cavi per il circuito di comando:

Cavo BIPOLARE 2x1.5mmq con In=19.5 A (posso copiare il precedente e modificare le proprietà), poi lo assegno al percorso disegnato nel sinottico.

Cavo UNIPOLARE 1x1.5MMQ FS17-450/750V

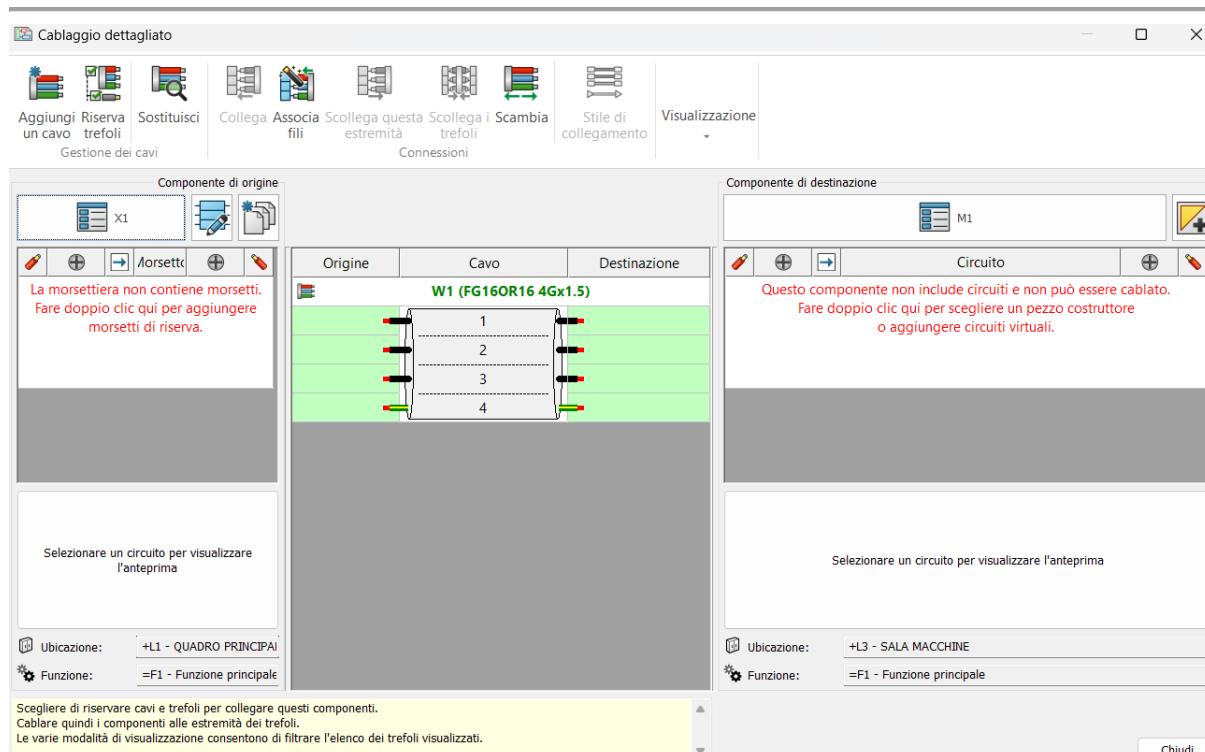
Alla fine ad ogni percorso è associato un cavo.

LISTA DEI CAVI: Per avere una prima lista dei cavi, con indicate le ubicazioni collegate andare su PROGETTO ELETTRICO→ RAPPORTI

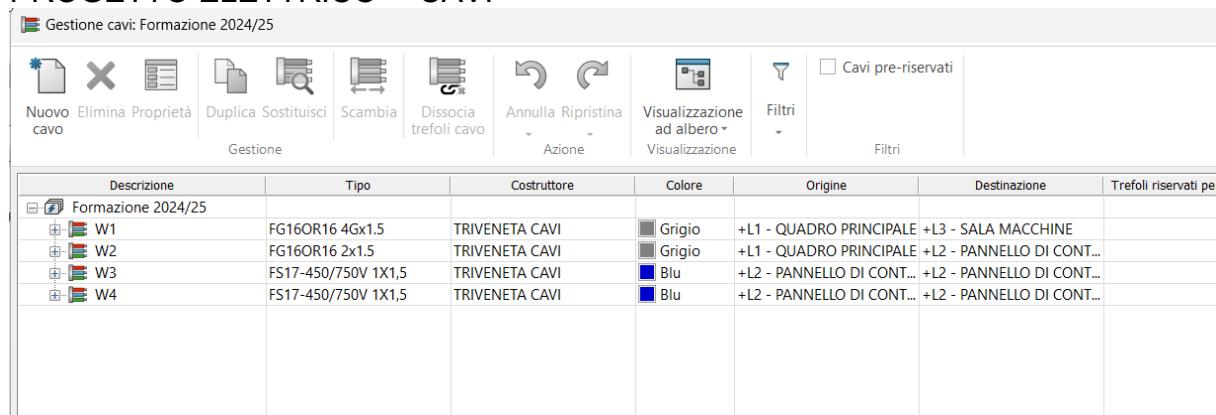
▲	Contrassegno	Descrizione	Percorso ubicaz...	Posizione di origine	Posizione di destin...	Lunghezza (...)	Riferimento	Costruttore
1	W1		+L1<>+L3	L1 - QUADRO PR...	L3 - SALA MACC...	0	FG160R16 4Gx1.5	TRIVENETA CAVI
2	W2		+L1<>+L2	L1 - QUADRO PR...	L2 - PANNELLO ...	0	FG160R16 2x1.5	TRIVENETA CAVI
3	W3		+L2<>+L2	L2 - PANNELLO ...	L2 - PANNELLO ...	0	FS17-450/750V 1X1,5	TRIVENETA CAVI
4	W4		+L2<>+L2	L2 - PANNELLO ...	L2 - PANNELLO ...	0	FS17-450/750V 1X1,5	TRIVENETA CAVI

INFORMAZIONI ANALOGHE LE POSSO OTTENERE DA:

SINOTTICO→ PARTICOLARI DI CABLAGGIO



OPPURE PROGETTO ELETTRICO → CAVI



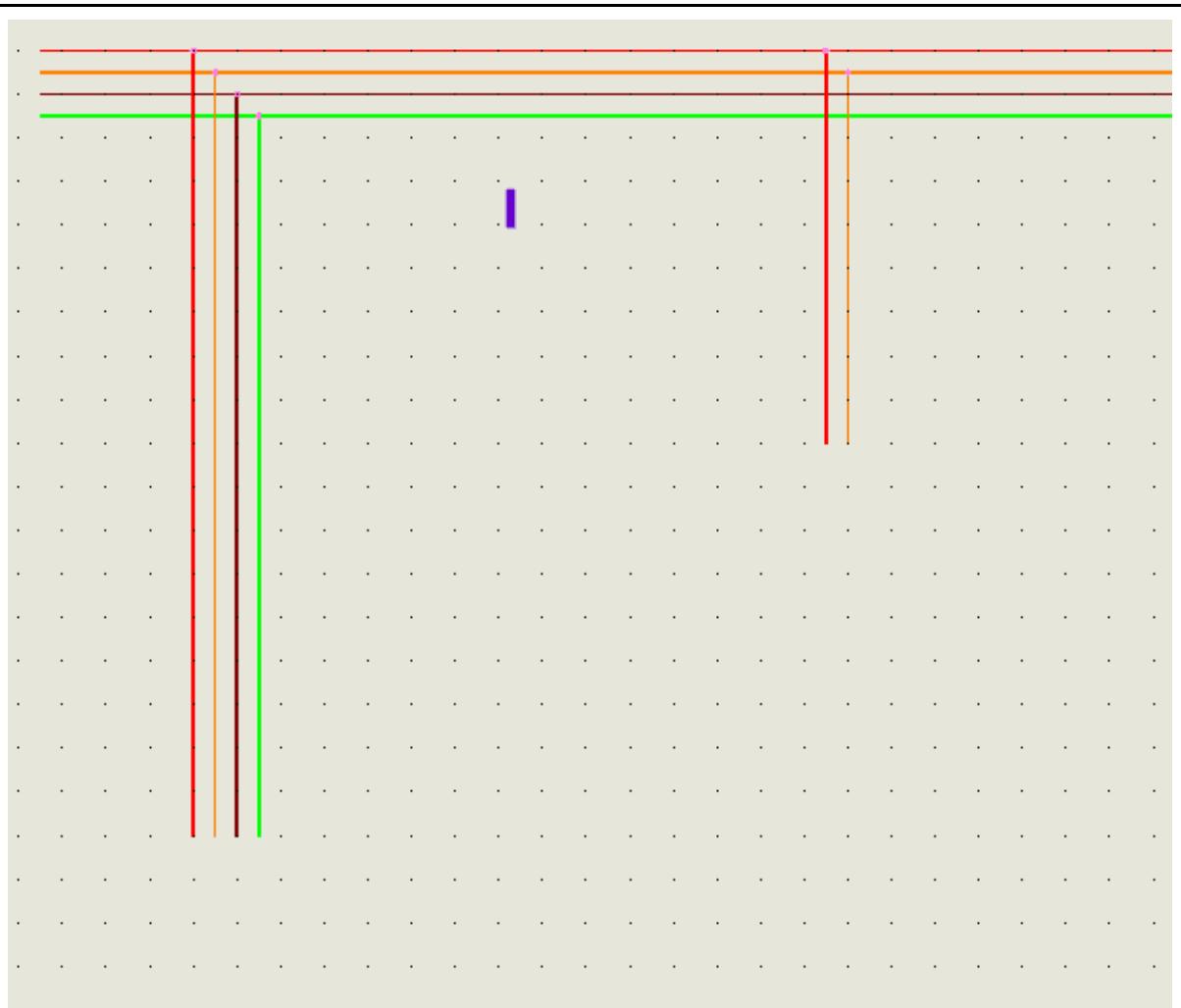
CORSO DI SOLIDWORKS ELECTRICAL

PROGETTO ELETTRICO

SELEZIONARE LA TAVOLA GRAFICA DELLO SCHEMA ELETTRICO

Selezionare “Tracciato fili multipli” e disegnare 3L++PE, disegnare la linea orizzontale di potenza e la linea che arriva al motore.

Sempre come circuito di potenza selezionando “Tracciato fili multipli” disegnare 2L.

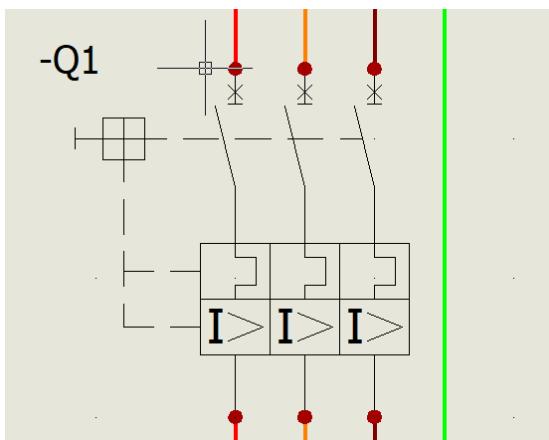


Selezionare “TRACCIATO FILO SINGOLO” selezionare 24V e realizzare il circuito di comando come in figura.

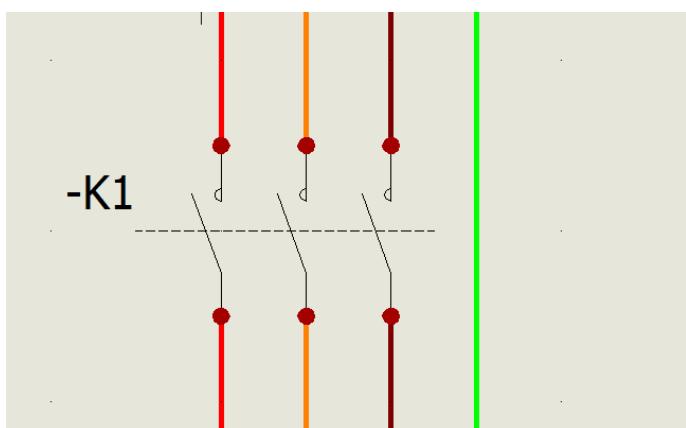
CRIVELLARO Laboratorio Luigi Pettinati 46, Padova crive.		Album documenti						REVISION 0
CONTRACT: 0001		LOCATION: +L1	QUADRO PRINCIPALE			Userdata 1	Userdata 2	SCHEME 04
			REV.	DATE	NAME	CHANGES		
		0	06/07/2024	drive				

INSERIRE I SIMBOLI:

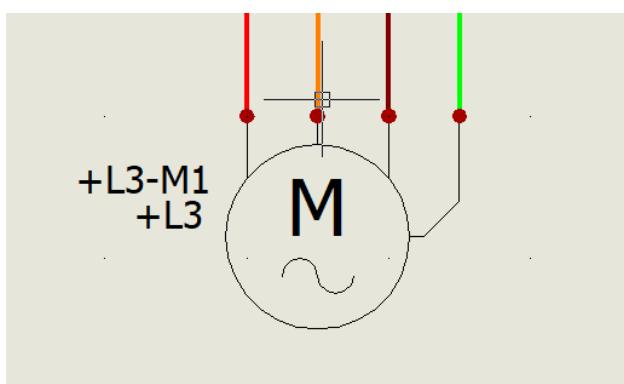
- **MAGNETOTERMICO:**
Selezionare la scheda Schema Elettrico → inserisci simbolo
Disgiuntori → Disgiuntore magnetotermico.
inserire il simbolo nella linea di potenza, controllando la corretta ubicazione (Quadro principale)



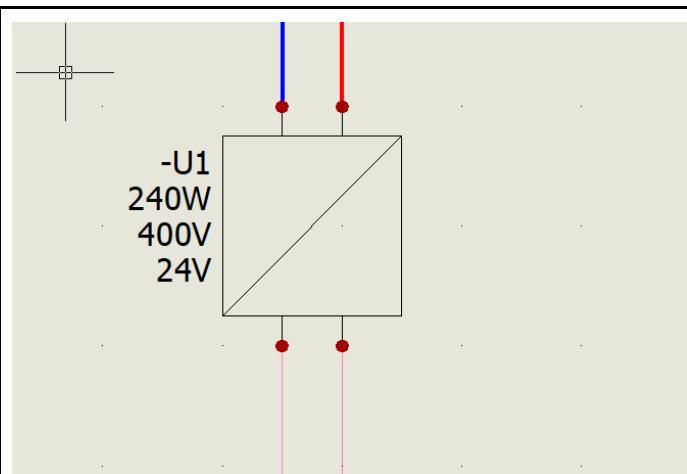
CONTATTORI,RELE': Contatto di potenza Tripolare



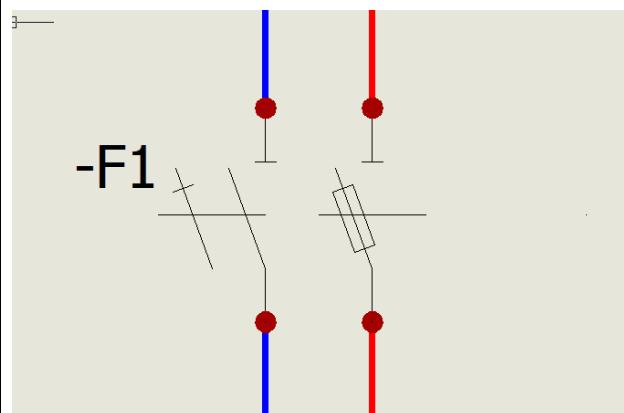
MOTORE CA TRIFASE TRE MORSETTI + PE

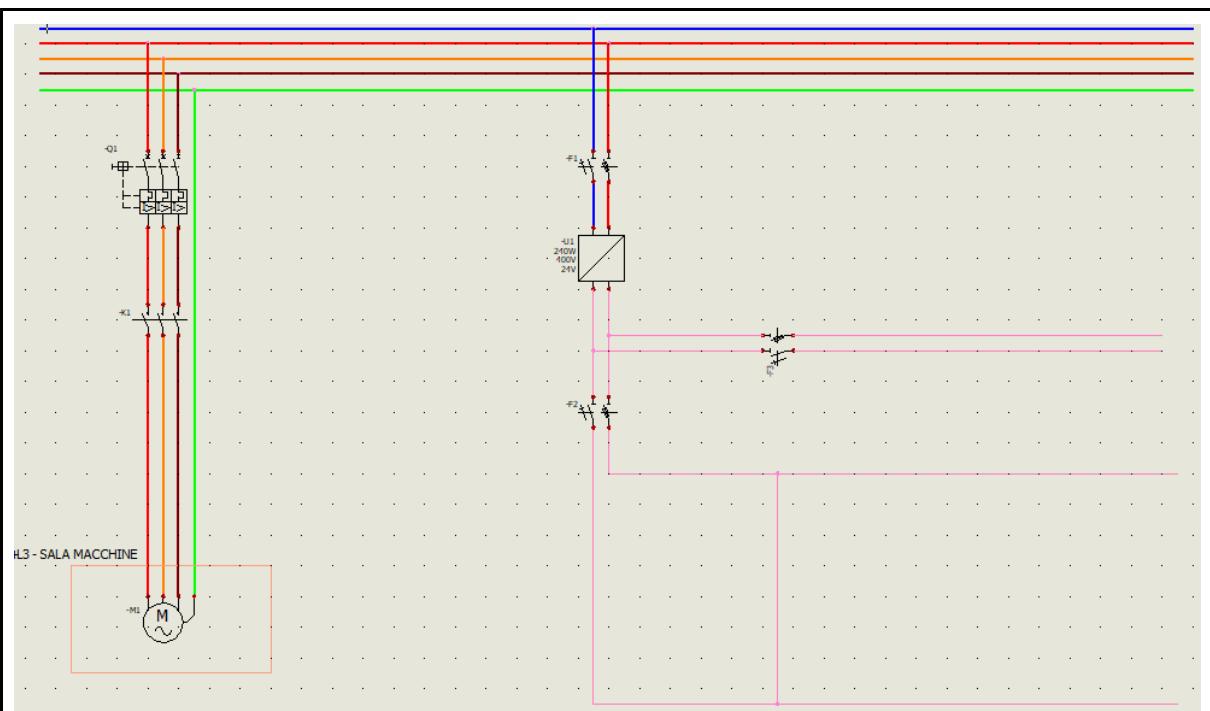


**CONVERTITORI INVERTER→CONVERTITORE STATICO
MONOFASE DA 240V/24V 5A**

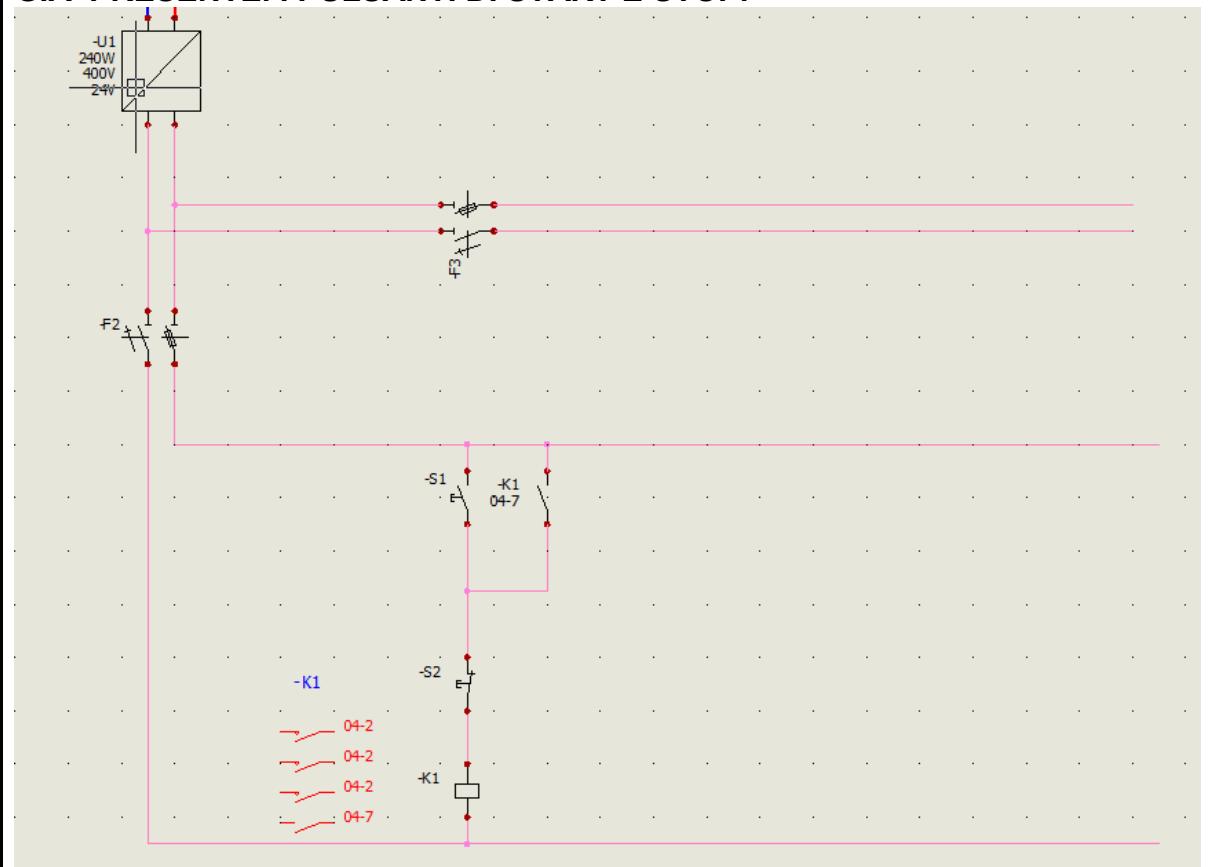


**FUSIBILE PRIMARIO ALIMENTATORE
FUSIBILE SECONDARIO ALIMENTATORE**





**ORA PASSIAMO A INSERIRE
RELE' DI COMANDO CONTATTORE: DEVO ASSOCIARLA AL CONTATTORE K1
GIA' PRESENTE. I PULSANTI DI START E STOP.**



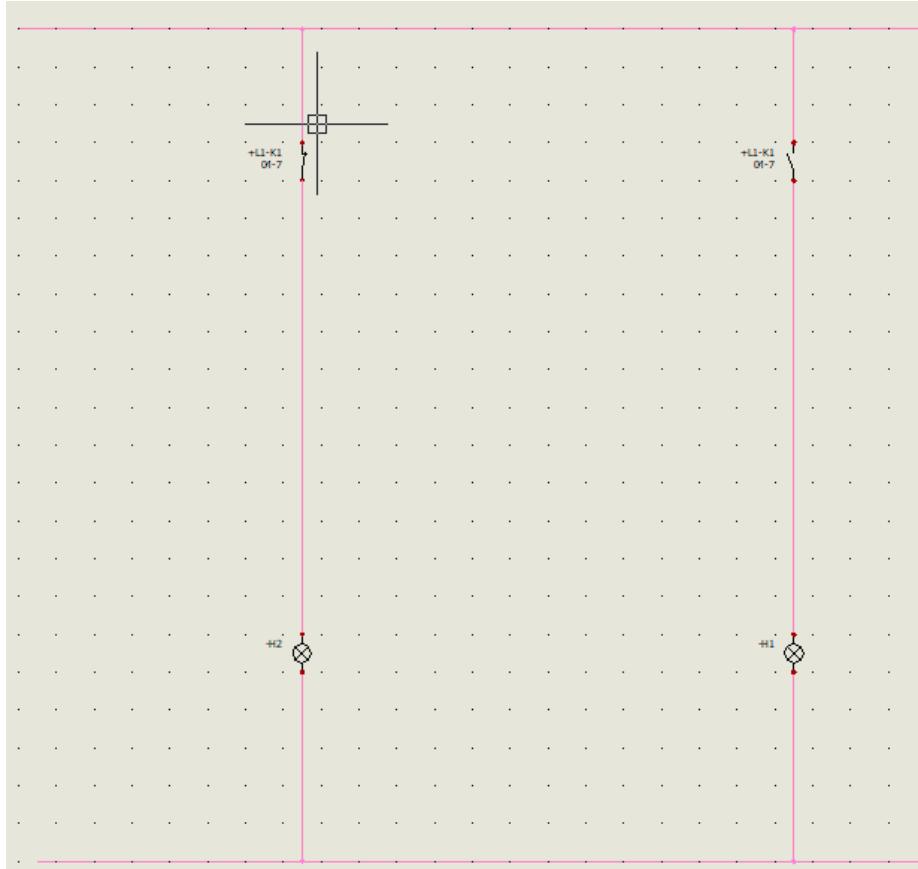
I PULSANTI DI AVVIAMENTO E STOP DEL MOTORE, E LE LAMPADE DI SEGNALAZIONE

CREAZIONE DI UN ULTERIORE SCHEMA ELETTRICO DEDICATO AL PANNELLO DI CONTROLLO:

ALBUM→ TASTO DESTRO DEL MOUSE → NUOVO SCHEMA

SU NUOVO SCHEMA→ TASTO DESTRO→ PROPRIETA'

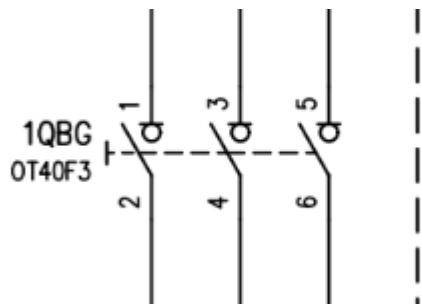
MODIFICO L'UBICAZIONE E INSERISCO NELLA DESCRIZIONE "Pannello di controllo"



SE MANCA UN SIMBOLO?????

CORSO DI SOLIDWORKS ELECTRICAL**CREARE UN SIMBOLO DA INSERIRE NELLO SCHEMA ELETTRICO**

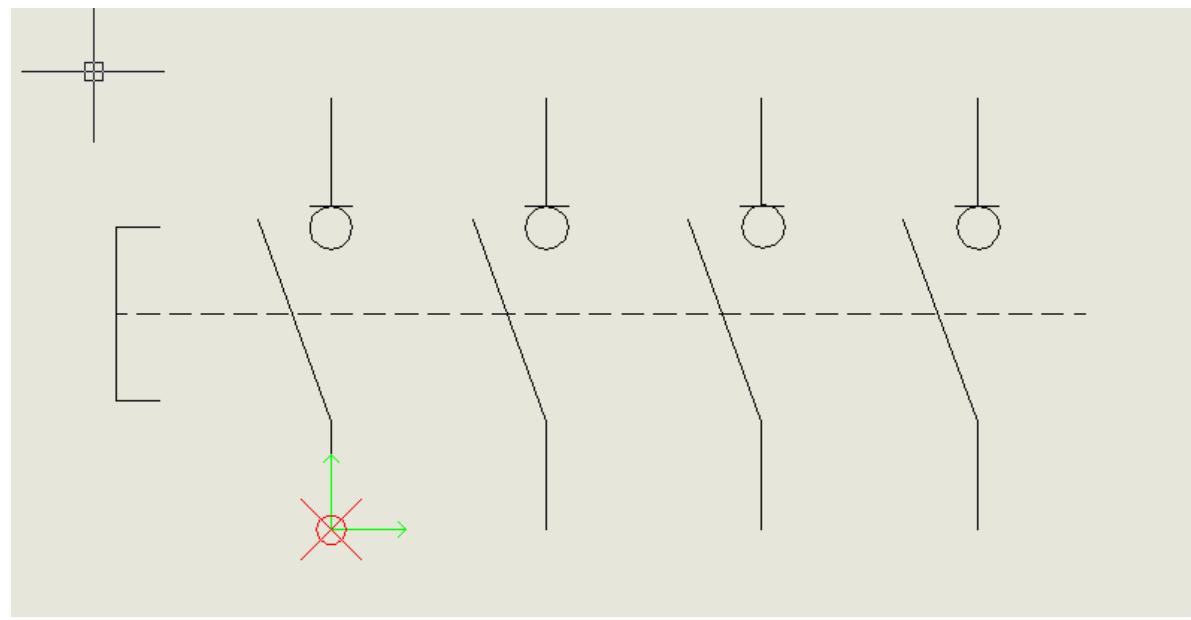
Devo creare un sezionatore manuale generale avente il simbolo sottoriportato

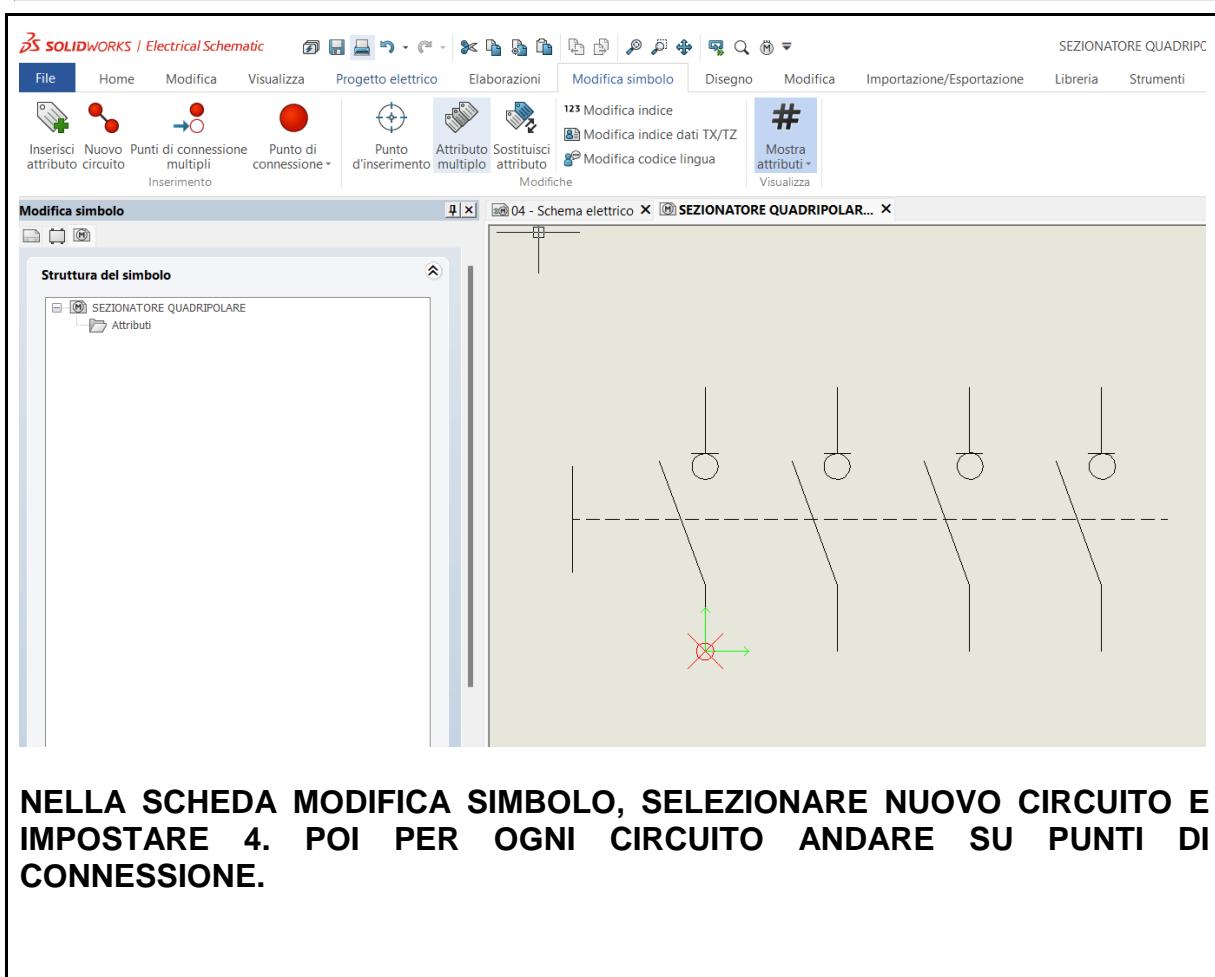


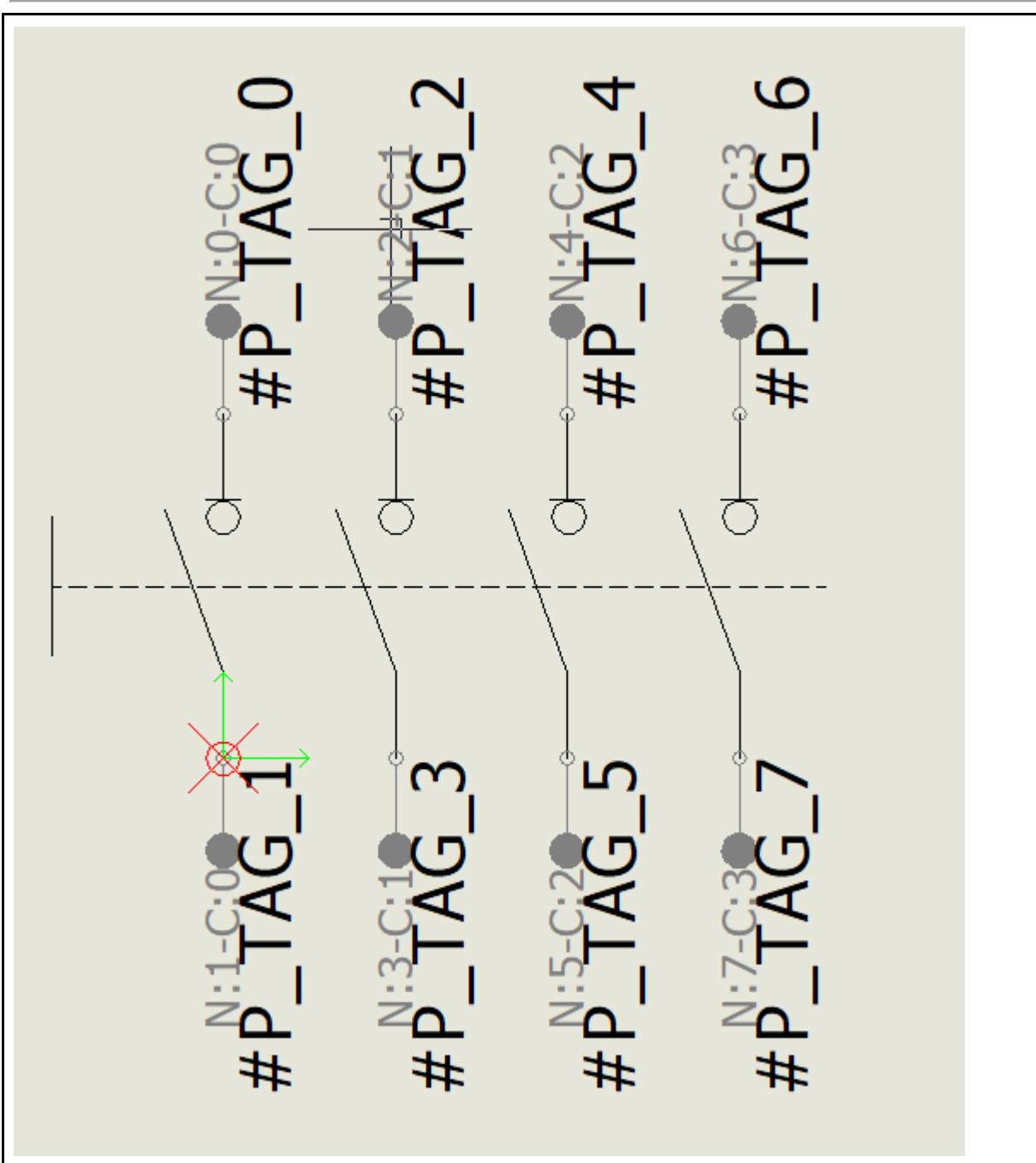
LIBRERIA→ GESTIONE SIMBOLI

SELEZIONO LA CATEGORIA (SEZIONATORI) ALLA QUALE APPARTIENE IL SIMBOLO E PREMO NUOVO.

SI DISEGNA IL SIMBOLO, PARTENDO DALLA COPIA DI UNO SIMILE.



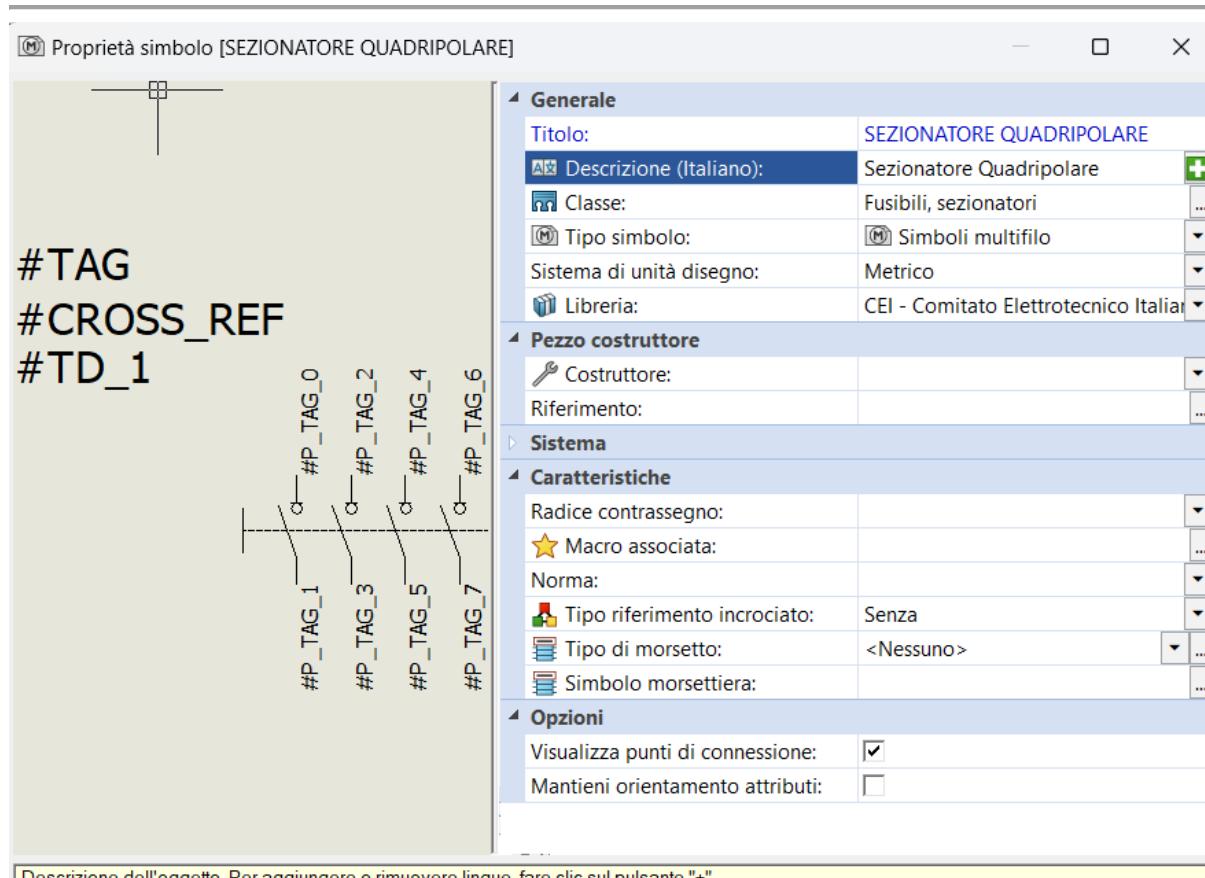




Gestione attributi

Nome	Descrizione	Valore
#TAG	Contrassegno componente	+L1=F1-A1-F1
#SHORT_TAG	Contrassegno corto	-F1
#PARENT_TAG	Contrassegno principale	+L1=F1-A1
#LOC_TAG	Contrassegno ubicazione	+L1
#FUN_TAG	Contrassegno funzione	=F1
#HAR_TAG	Contrassegno cablaggio elettronico	
#REF_MAN	Costruttore	
#REF_REF	Riferimento	
#REF_MAN_ALL	Costruttore	
#REF_REF_ALL	Riferimento	
#REF_ART_NUM	Numero di articolo	AN-001
#REF_SERIES	Serie	abc
Dati utente per il pezzo costruttore		
Tensione e frequenza pezzi costruttore		
Dati utente per il componente		
Ubicazione		
Funzione		
Cablaggio elettronico		
Dati descrittivi		
Riferimenti incrociati		
#CROSS_REF	Riferimenti incrociati	
#LAB_CROSS_REF	Riferimenti incrociati di etichette di cab...	
Dati del costruttore		
#TD_1	Corrente nominale	
#TD_2	Valore 2	
#TD_3	Grandezza	
#TD_4	Calibro	
#TD_5	Velocità	

DEFINIZIONE DELLE PROPRIETA'



CORSO DI SOLIDWORKS ELECTRICAL

COLLEGAMENTI TRA DISEGNI

Si prevede di collegare il percorso dei cavi schema elettrici realizzati su pagine diverse.

SCHEDA SCHEMA ELETTRICO → RINVII, SI APRE UN AMBIENTE CHE CONSENTE DI COLLEGARE I DIVERSI DISEGNI.

CORSO DI SOLIDWORKS ELECTRICAL

MORSETTIERE: PROGETTAZIONE E DISEGNI

Per disegnare le morsettiera: X1,X2,X3 andare in:

SCHEMA ELETTRICO→ INSERISCI MORSETTO O INSERISCI “n” MORSETTI.
Si sceglie l’orientamento della morsettiera: (verso di percorrenza dello schema da monte a valle) e si associa la morsettiera corretta in base a quelle definite da sinottico.

ASSOCIARE AD OGNI PERCORSO NUMERATO IL CAVO COMMERCIALE:
tasto destro associa trefoli cavo.

ASSOCIARE AD OGNI MORSETTIERA IL COMPONENTE COMMERCIALE

Per analizzare il corretto cablaggio delle morsettiera
PROGETTO ELETTRICO→ MORSETTIERE → selezionare la singola morsettiera.

Per produrre i disegni delle morsettiera:
PROGETTO ELETTRICO→ MORSETTIERE → DISEGNA PAGINE

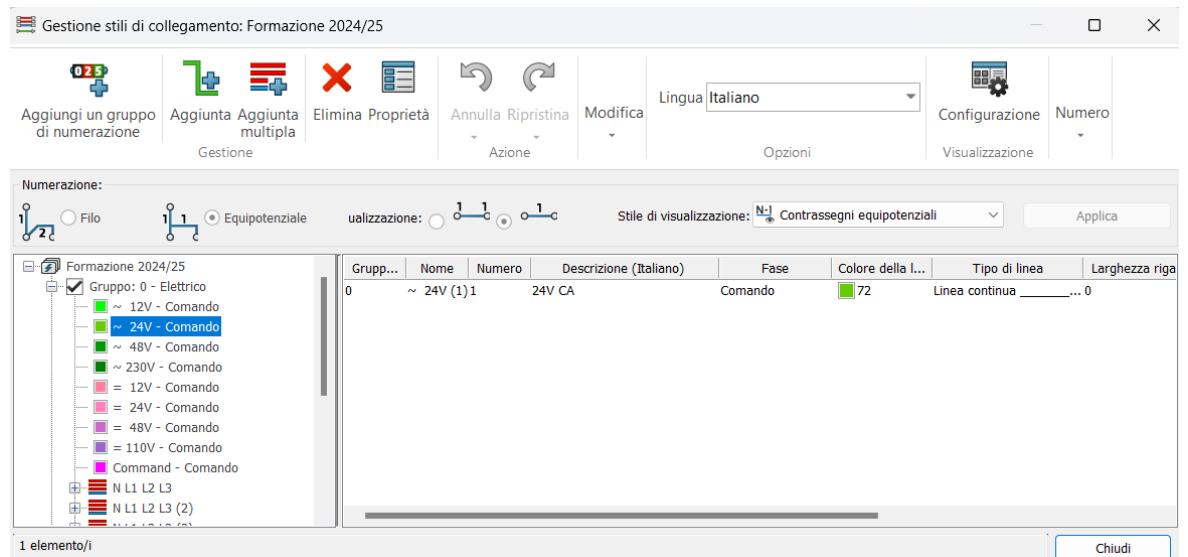
CORSO DI SOLIDWORKS ELECTRICAL

CAVI EQUIPOTENZIALI

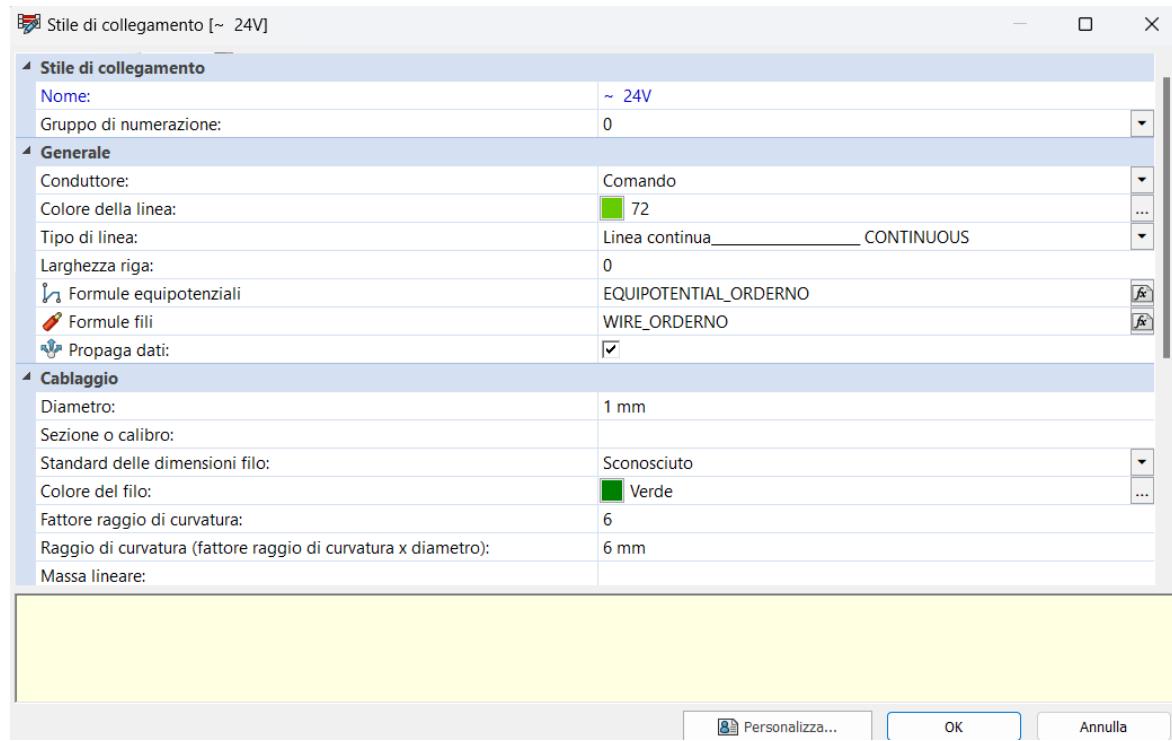
SCHEDA ELABORAZIONI → RINUMERA I NUOVI FILI

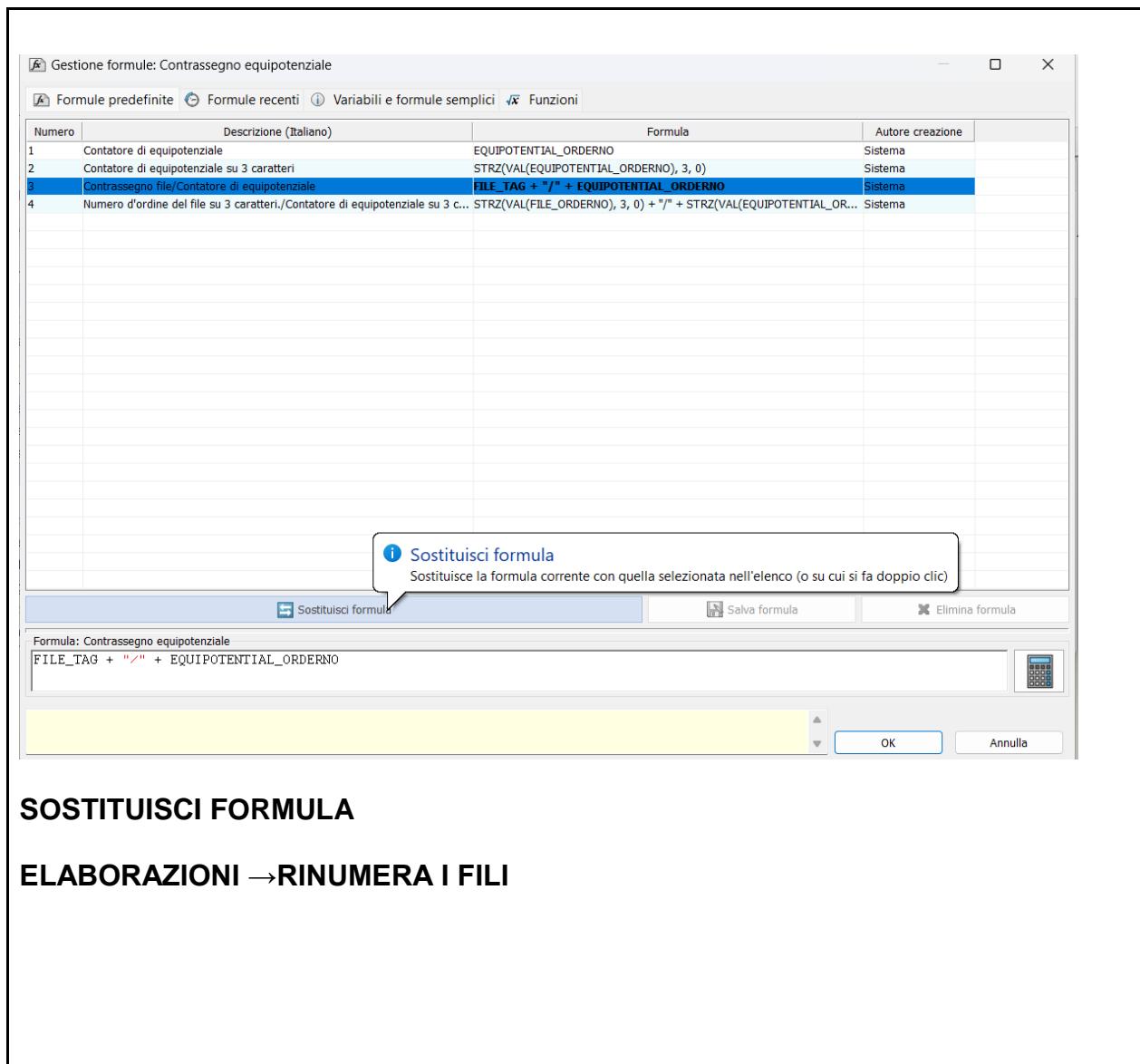
MODIFICARE LE PROPRIETA' DEI CAVI "DISEGNATI TRACCIATO FILO SINGOLO"

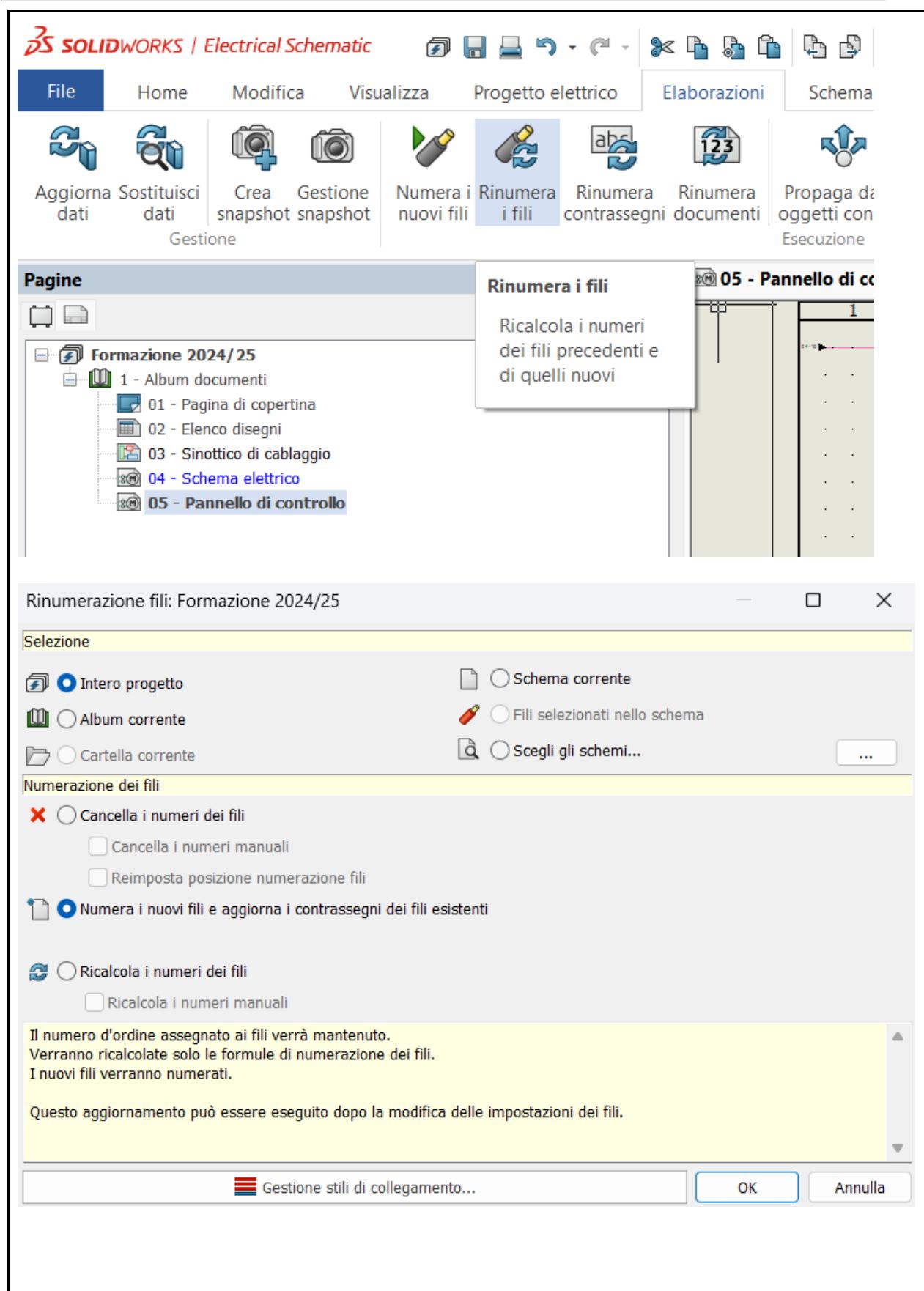
PROGETTO → CONFIGURAZIONE → STILE COLLEGAMENTO 24v



SELEZIONARE PROPRIETA'







CORSO DI SOLIDWORKS ELECTRICAL

RAPPORTI

**Per farci restituire tavole con informazioni relative al progetto
PROGETTO ELETTRICO→ RAPPORTI**

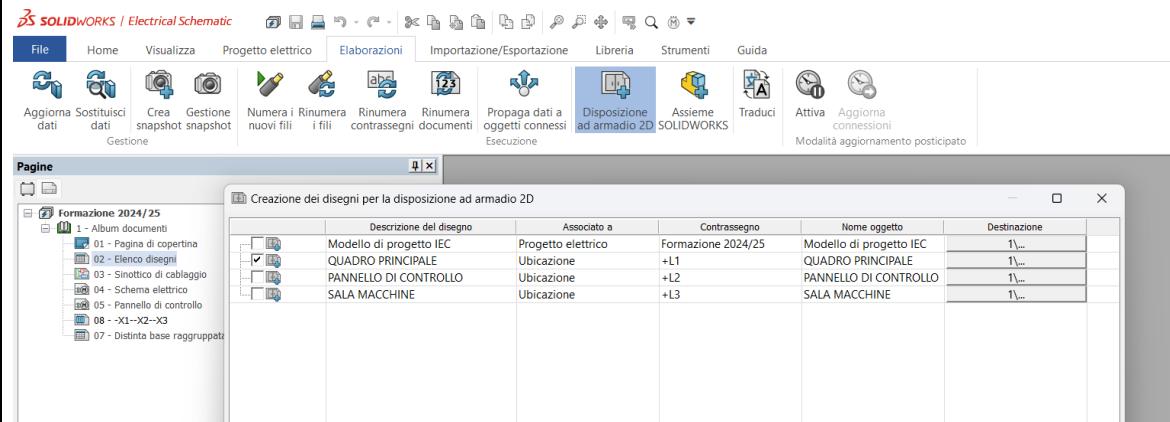
**Per produrre i disegni dei rapporti
PROGETTO ELETTRICO→ RAPPORTI→ GENERA DISEGNI**

- **DISTINTA BASE: elenco dei componenti da acquistare**
-

CORSO DI SOLIDWORKS ELECTRICAL

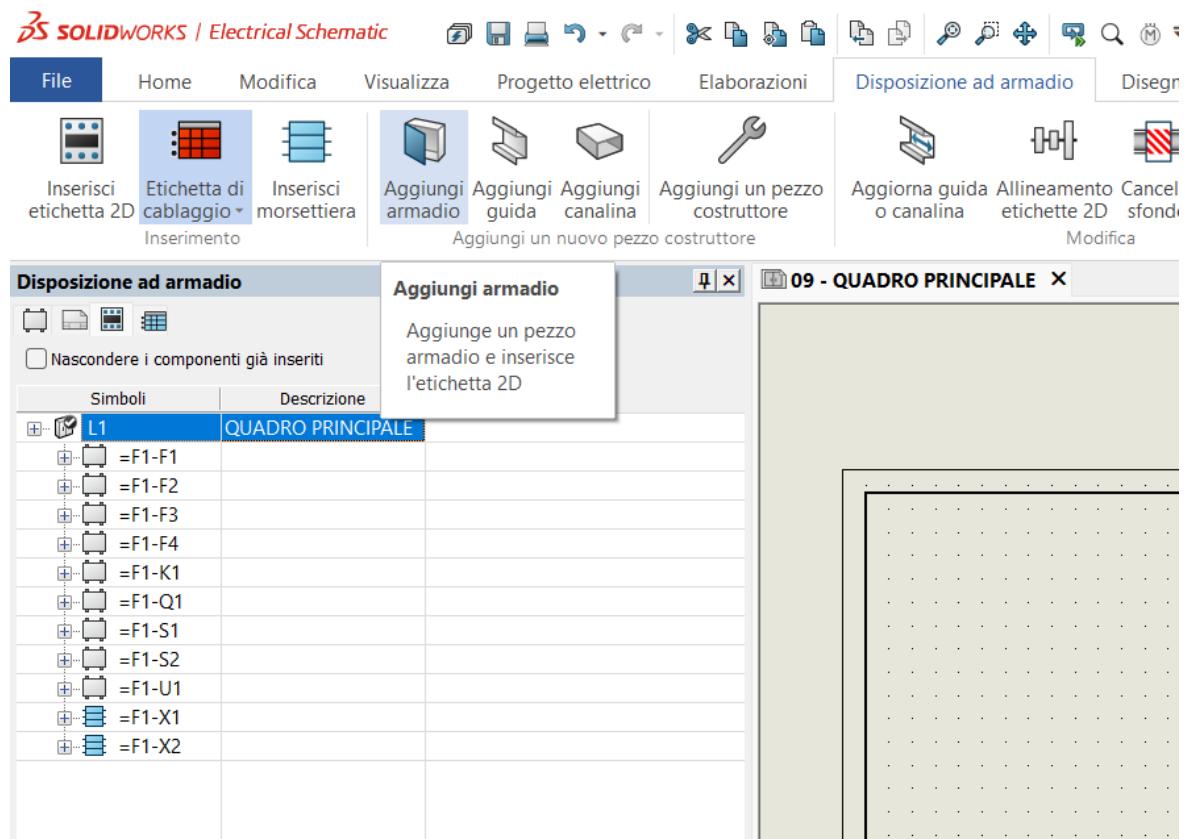
QUADRO ELETTRICO

ELABORAZIONI→DISPOSIZIONE AD ARMADIO 2D→ SELEZIONARE QUADRO ELETTRICO PRINCIPALE



**APRIRE IL NUOVO DISEGNO CREATO.
A SINISTRA SELEZIONANDO I COMPONENTI SI OSSERVA CHE SONO STATI ASSOCIATI I DISEGNI 2D DA INSERIRE NEL QUADRO.**

AGGIUNGERE L'ARMADIO: SELEZIONARE LA LOCAZIONE “QUADRO PRINCIPALE” E PREMERE AGGIUNGI ARMADIO, SELEZIONIAMO 09113 DELLA SCHNEIDER



MODIFICARE LA SCALA 1:3

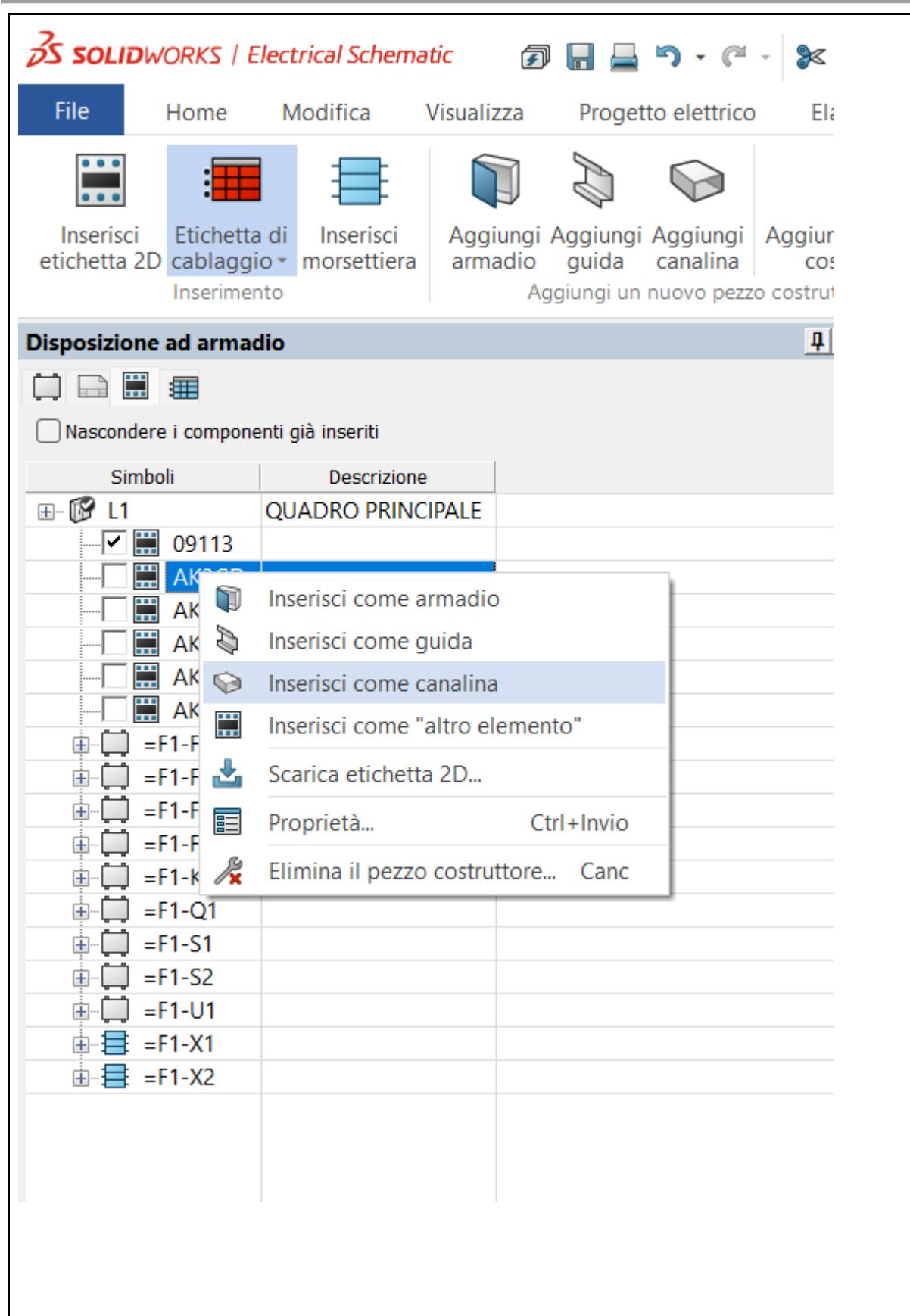
GUIDA DIN: AGGIUNGI CANALINA RICERCARE CODICE AK2GD5050

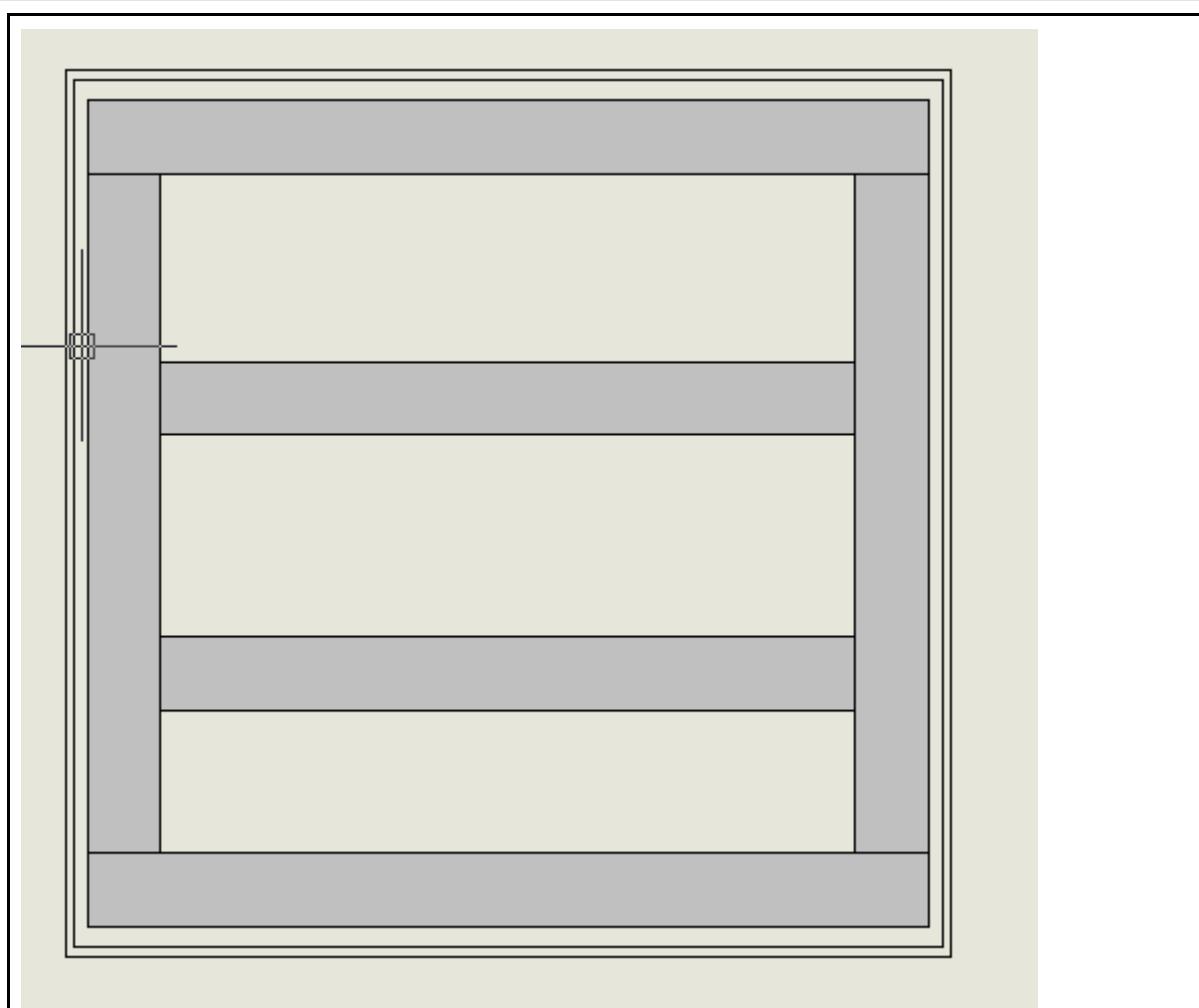
09 - QUADRO PRINCIPALE

Messaggio
Selezionare un pezzo

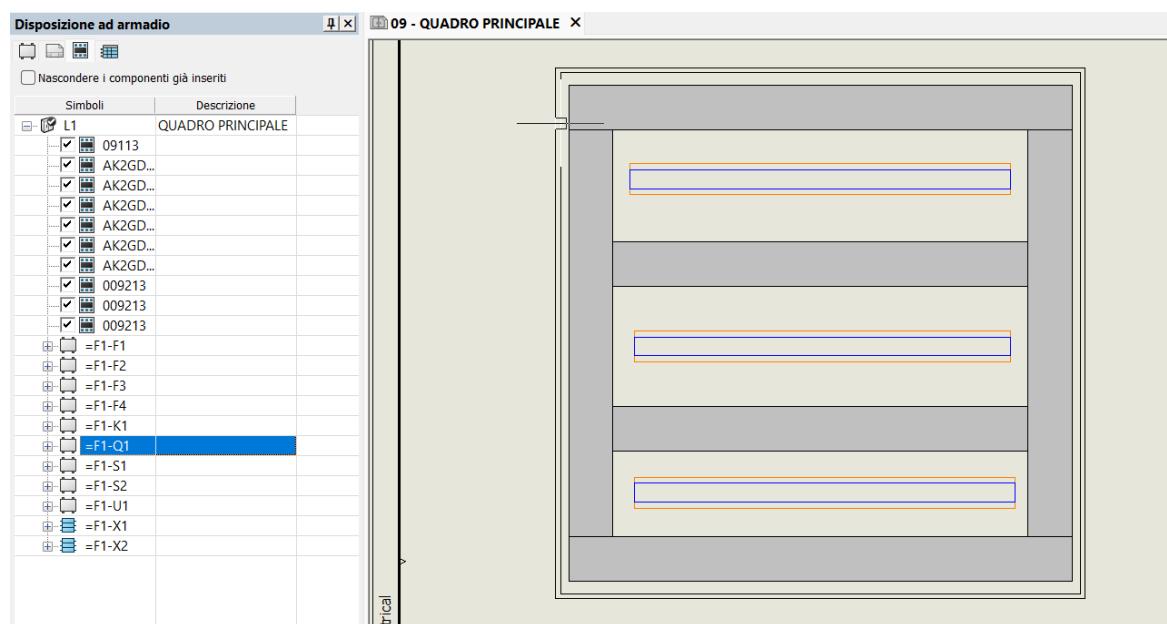
Numero	Riferimento	Descrizione (Italiano)
1	AK2GD5050	

**AGGIORNA GUIDA O CANALINA CONSENTE DI MODIFICARE LA LUNGHEZZA COSÌ DA ADATTARLA ALLA DIMENSIONE DEL QUADRO.
INSERIAMO 6 CANALINE COSÌ DA OTTENERE:**





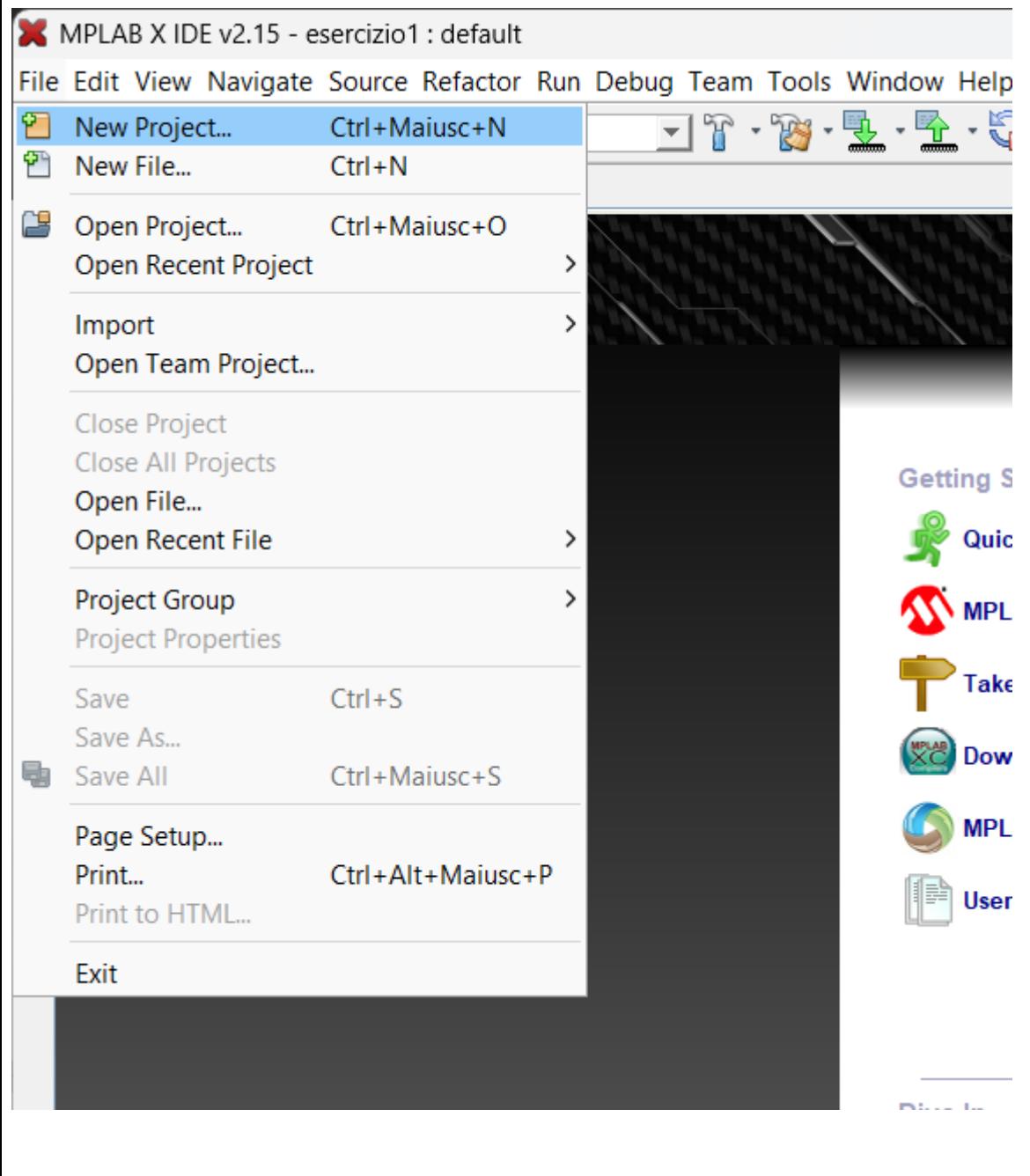
AGGIUNGERE TRE GUIDE DIN CODICE 009213

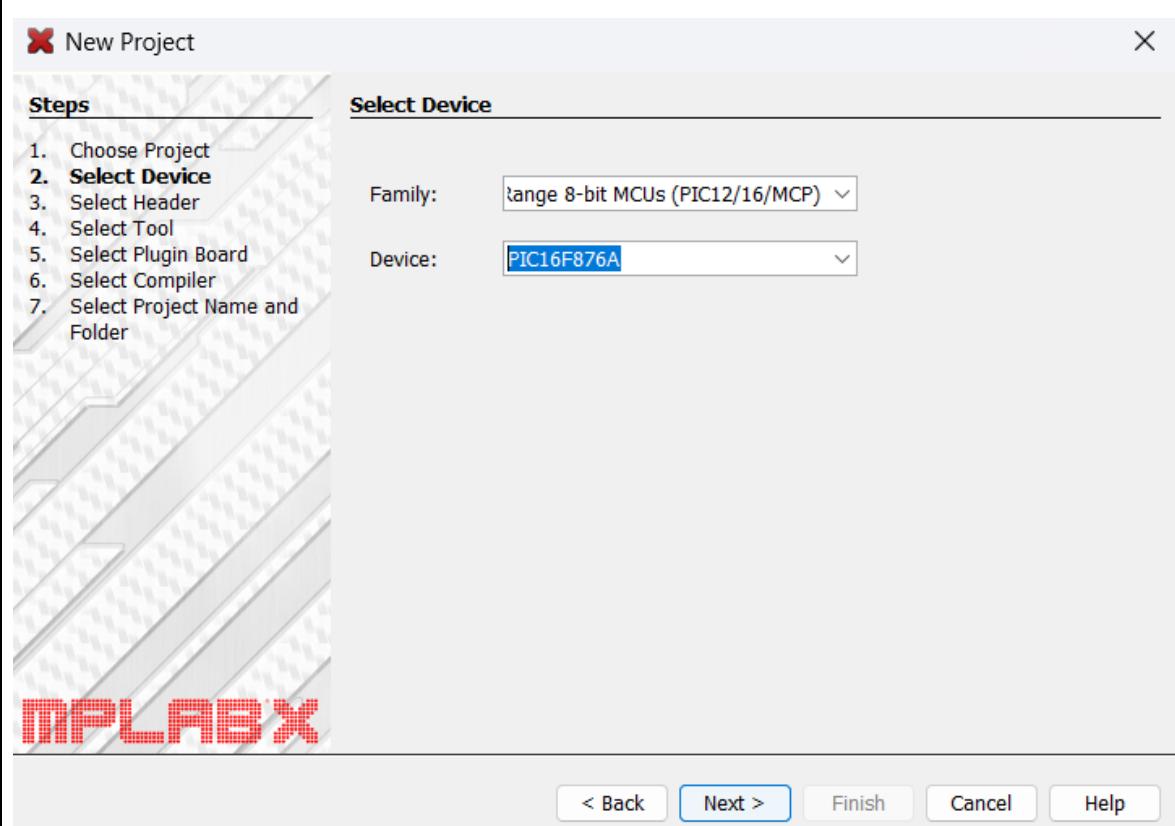
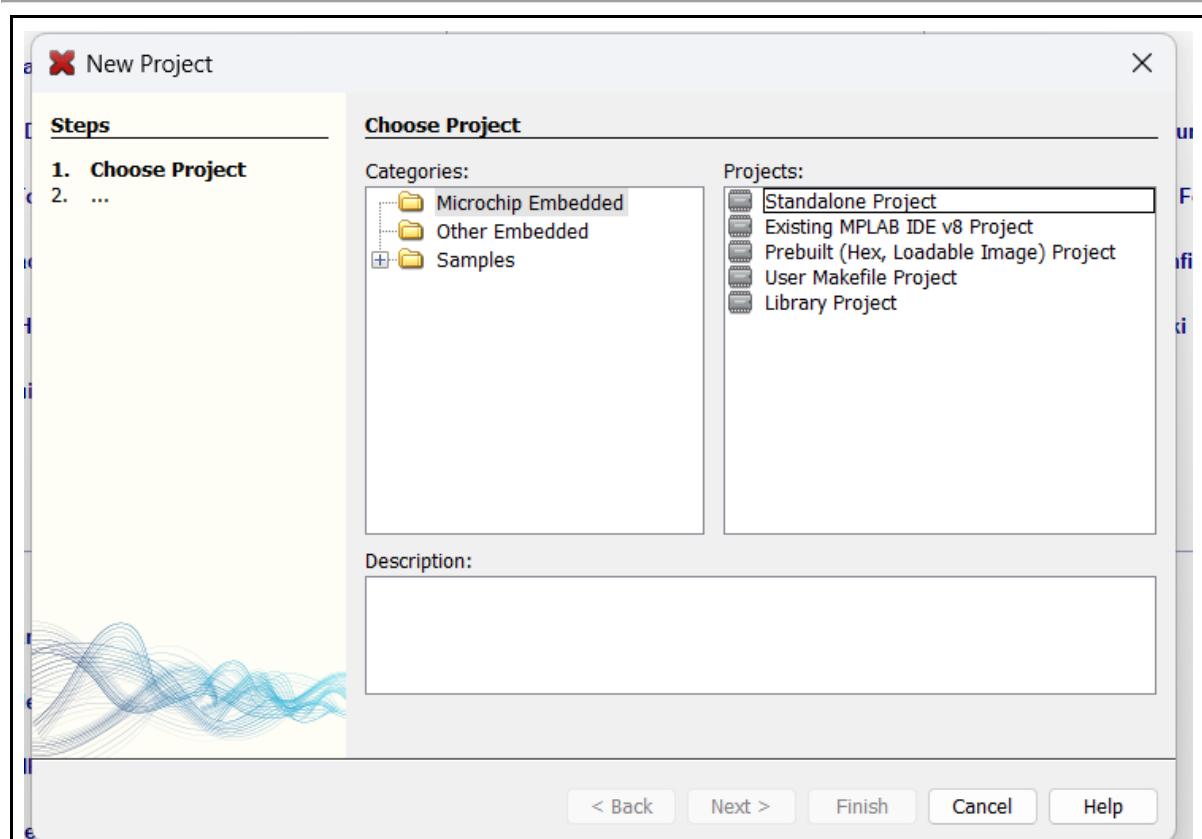


LIBRERIA FILE PER ARMADIO 2D:

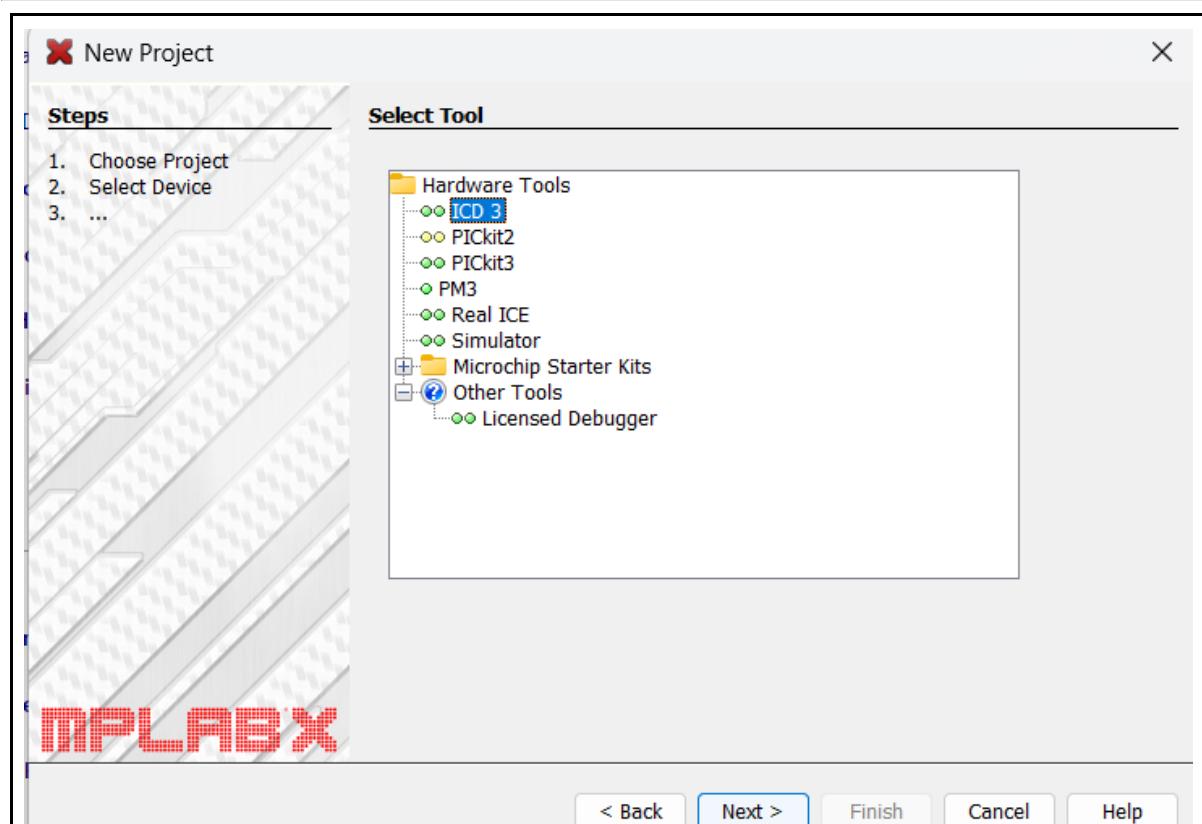
GESTIONE ETICHETTE 2D:**IMPORTAZIONE FILE DWG****MICROCONTROLLORE PIC 18F876A**

CREAZIONE DI UN NUOVO PROGETTO

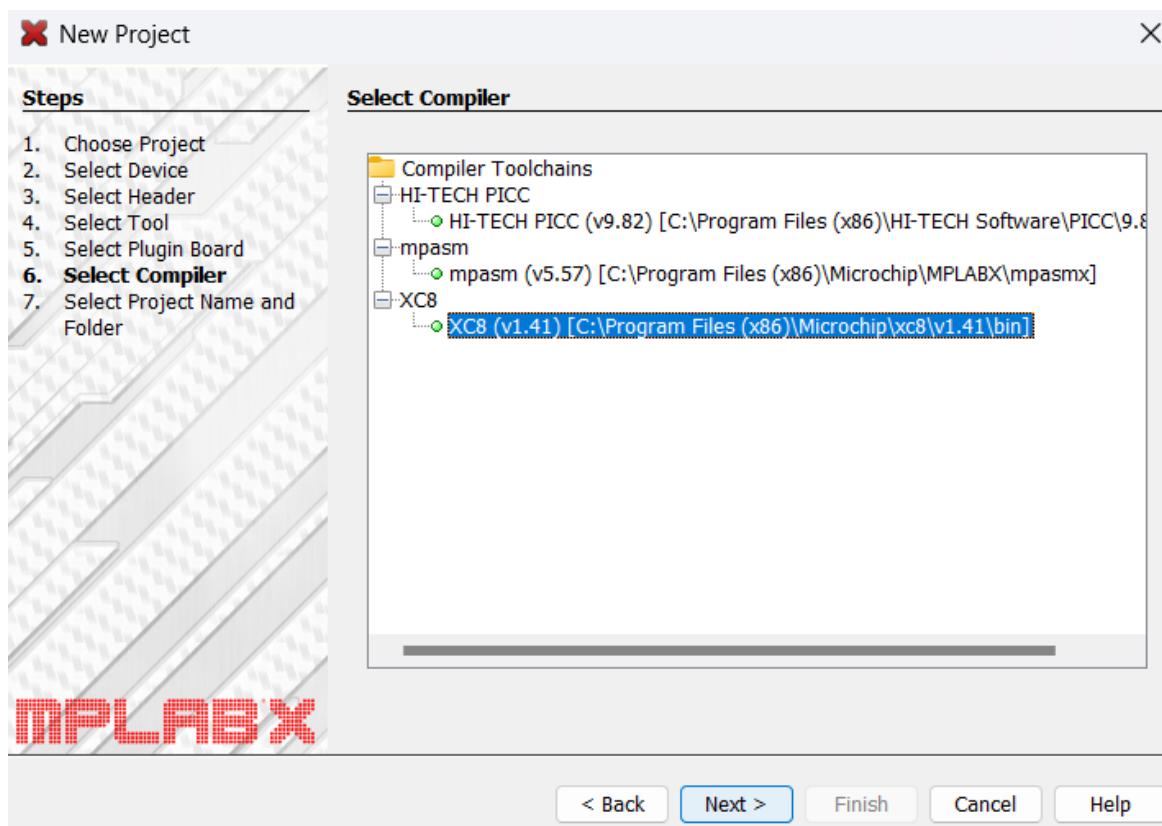




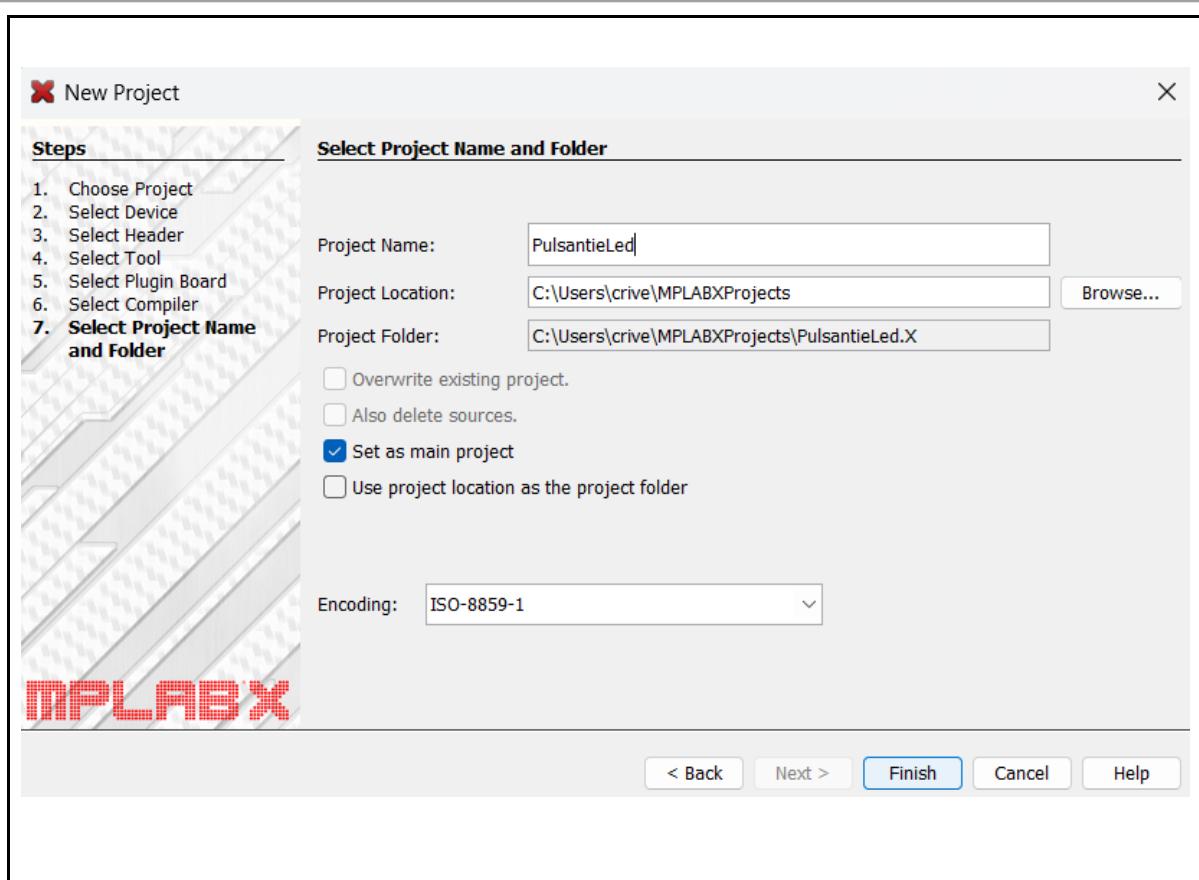
SELEZIONARE IL PROGRAMMATORE NEL MIO CASO ICD 3

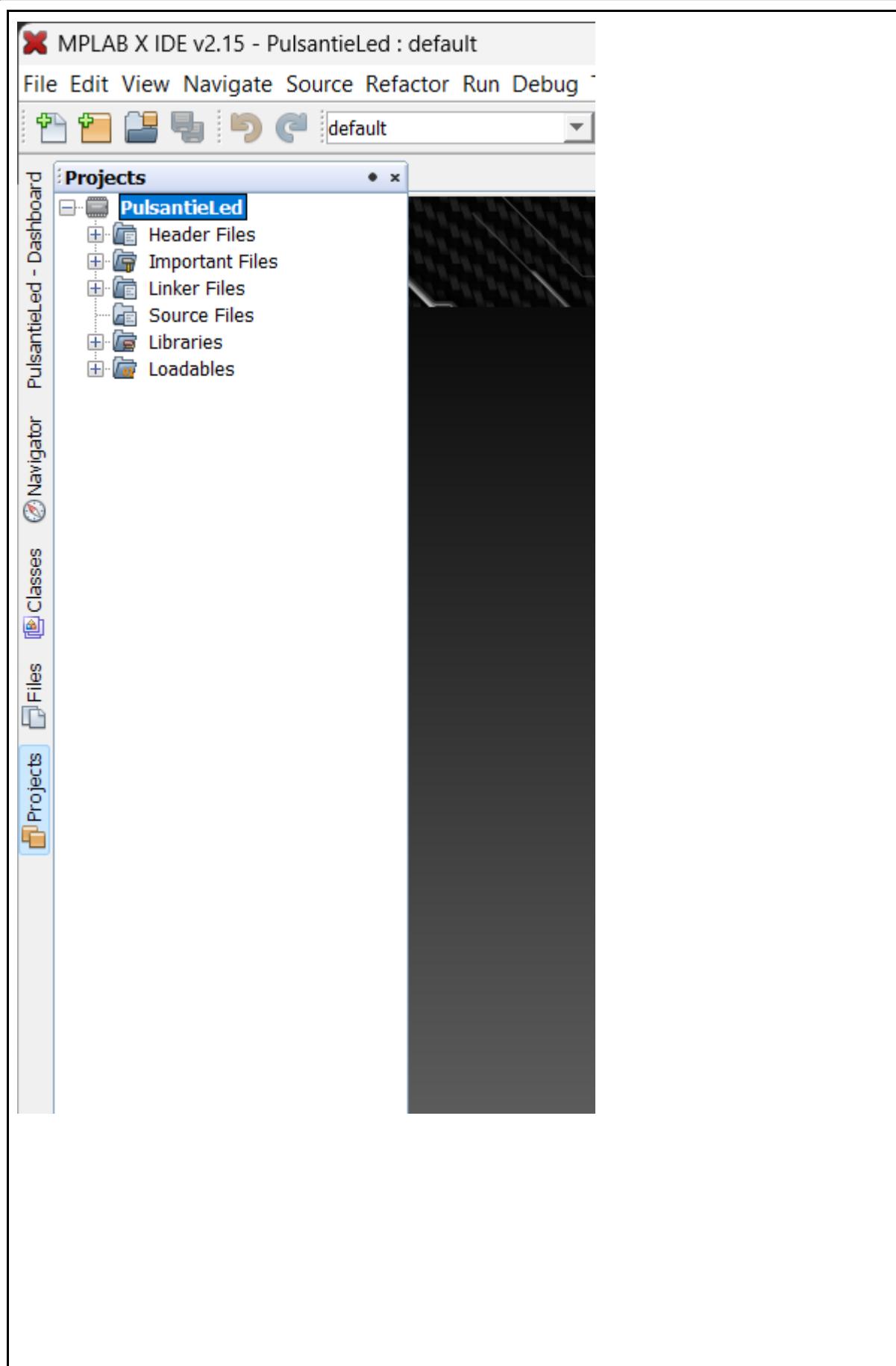


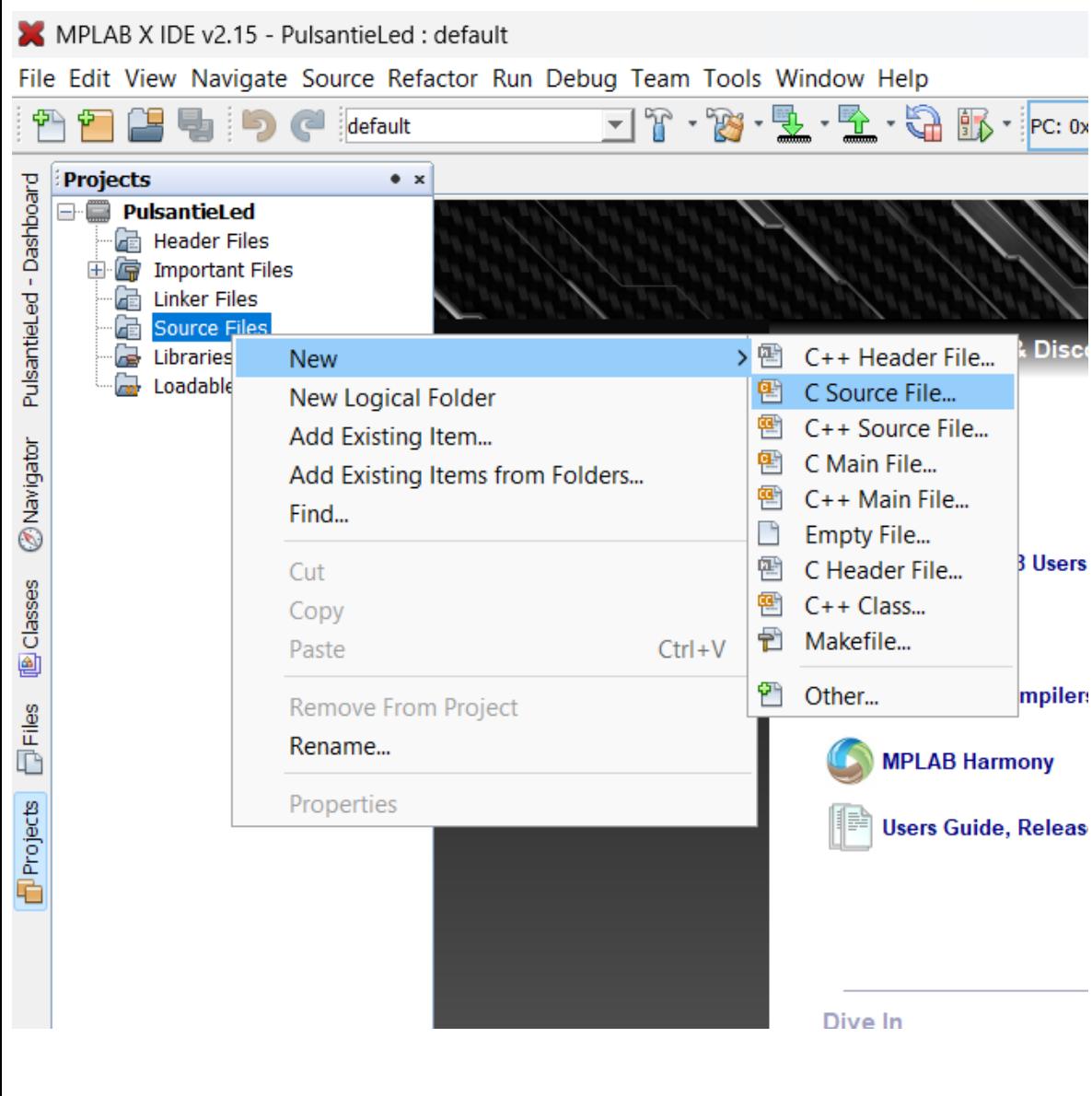
SELEZIONARE IL COMPILATORE XC8

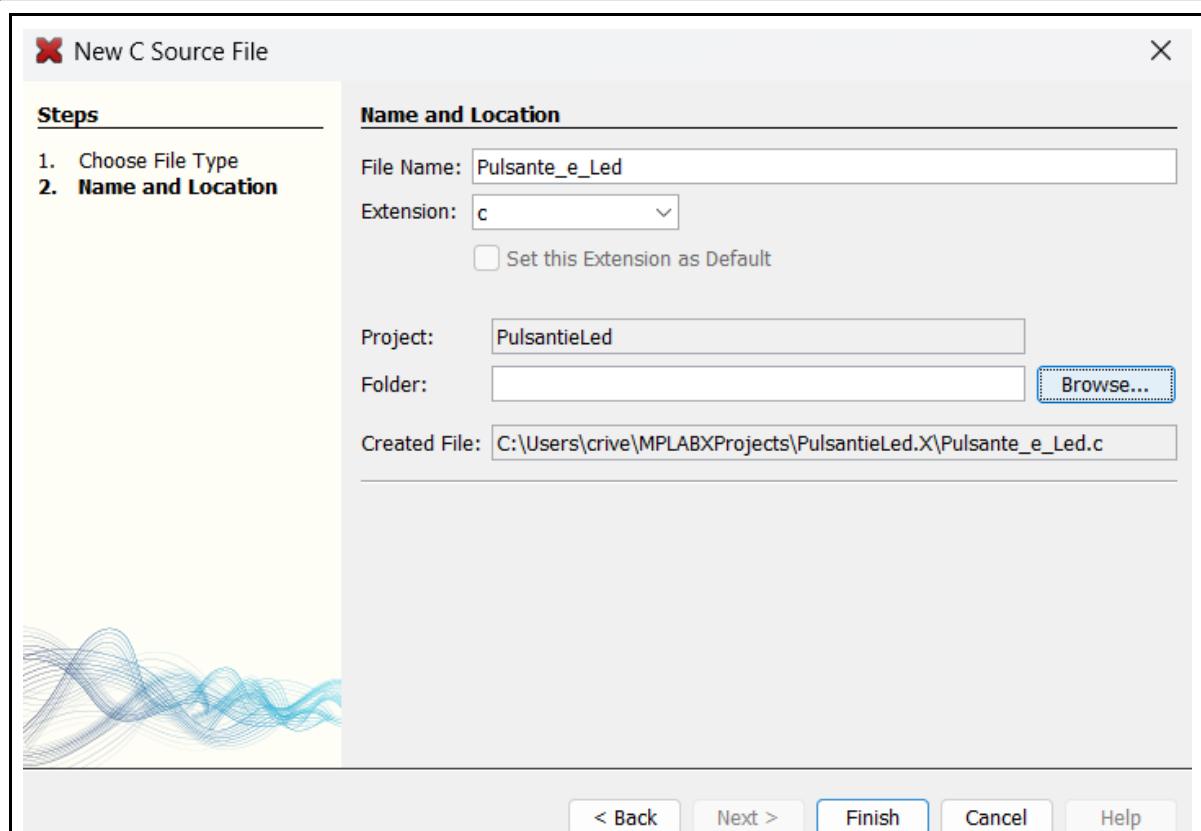


ASSEGNAZIONE DI UN NOME AL PROGETTO:





CREAZIONE DEL FILE SOURCE IN LINGUAGGIO DI PROGRAMMAZIONE C



Configurazione Bit

- 1) FOSC: HT – questo mi permette di utilizzare un oscillatore al quarzo di media potenza adatto fino a frequenze di 4MHz
- 2) WDTE: OFF – ho disabilitato il Watchdog Timer. Questo è un timer indipendente che, se attivo, ogni tanto deve essere resettato nel nostro programma altrimenti se termina il suo conteggio esegue il reset del programma.
E' una funzione di sicurezza nel caso in cui i programmi si blocchino.
- 3) PWRTE: ON – Il programma non entra in funzione fino a che la tensione di alimentazione non è stabile.
- 4) BOREN: OFF – Brown Out disabilitato. Il Brown Out è una funzione che resetta il microcontrollore nel caso in cui ci siano abbassamenti di tensione, è utile in alcune applicazioni per evitare corruzione dei dati o altri problemi.

- 5) LVP: OFF – Programmazione a bassa tensione disattivata. Questo serve per programmare il PICmicro senza fornire l'alta tensione sul pin MCLR (basta la 5V), ma in compenso bisogna utilizzare un altro pin (RB3/PGM).
- 6) CP: OFF – Protezione area codice disattivata
- 7) CPD: OFF – Protezione area dati disattivata
- 8) WRT: OFF – Protezione da auto-scrittura memoria flash disattivata

PCB CON MULTISIM ULTIBOARD

SCHEMA ELETTRICO

MORSETTIERA A VITE: Connectors - Terminal Blocks (nello schema devo collegarci i componenti virtuali Vcc e massa)

JUMPER: Connectors - Headers Test-HDR 1x10

MENU TRANSFER TO ULTIBOARD

MENU VIEW: bird air si divide la finestra in due

MENU VIEW: 3d preview

piazzole di saldatura

Layer RATSNEST: spengo e accendo i collegamenti tra componenti.

Il lato dove c'è l'etichetta è il lato dove sono presenti i morsetti

SETUP prima di eseguire lo sboglio**Scelta della larghezza delle piste****Scelta della dimensione delle piazzole****Scelta della dimensione dei fori delle piazzole stesse****E' utile l'attivazione da view del blocco spreadsheet view****Scheda net: Trace width cioè larghezza della traccia****di default il programma propone larghezze da 10mils****Trace clearance: distanza di rispetto tra pista e pista o tra pista e piazzola.****THT pads: mostra le informazioni riguardanti le piazzole di quei componenti a foro passante****SMT pads: mostra informazioni delle piazzole dei componenti saldati sul lato del componente.****COPPER LAYER: disabilito i layer dove non ci sono traccie.****minima larghezza di una pista: 0,3 mm per produzioni industriali con standard elevati, 0,5 produzione hobbistica.****per impostare in mills trasformo i mm in pollici e moltiplico per 1000****0,5mm x****1pollice/25,4mm=x/0,5mm****x=19,68 mills****possiamo impostare una larghezza delle piste di 20 mills****Clearance lasciamo 10 mills**

Diametro del foro	Diametro esterno della piazzola
< 0,8	1,9
1	2,2
1,3	2,6

Noi possiamo usare un diametro del foro da 0,8 e un diametro esterno della

**piazzola da 1,9mm
1pollice/25,4mm=x/0,8mm**

x=31,49 mills

foro da 32 mills

1pollice/25,4mm=x/1,9mm

x=74,80 mills

piazzola da 75 mills

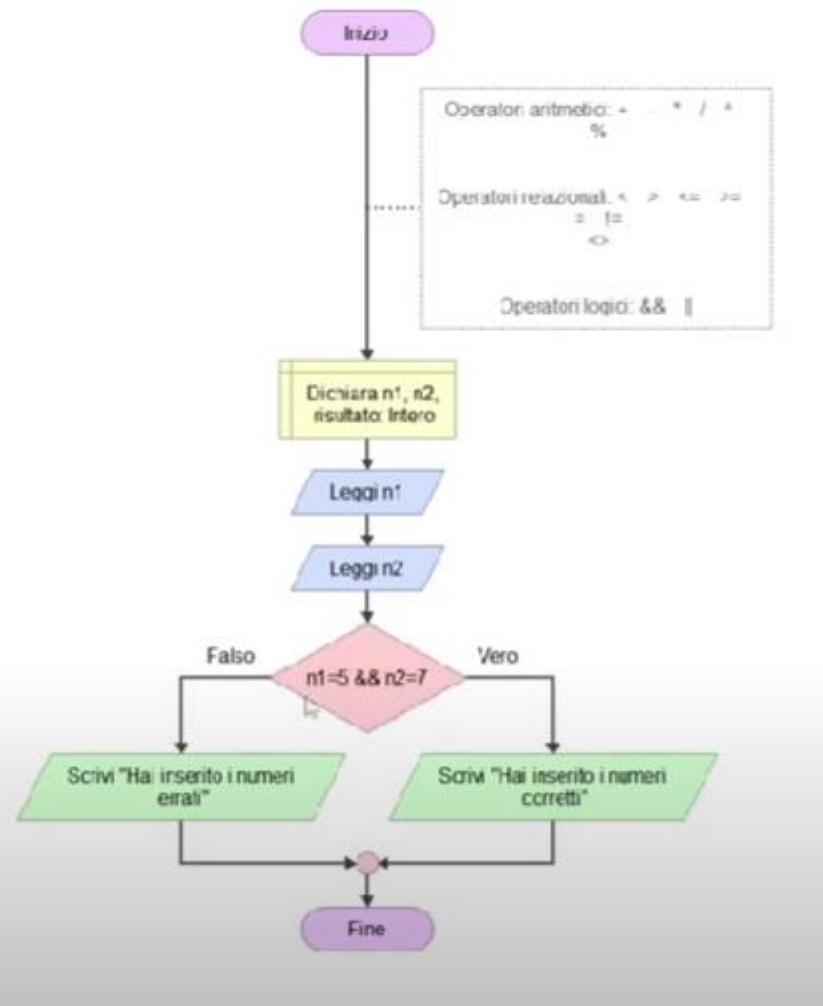
Seleziono tutte le piazzole, tasto destro proprieta' imposto foro e piazzola, e layer setting (solo lato rame autorouting)

Sbroglio automatico detto autorouting

Premo il simbolo con il fulmine "start autorouting"

FLOWGORITHM

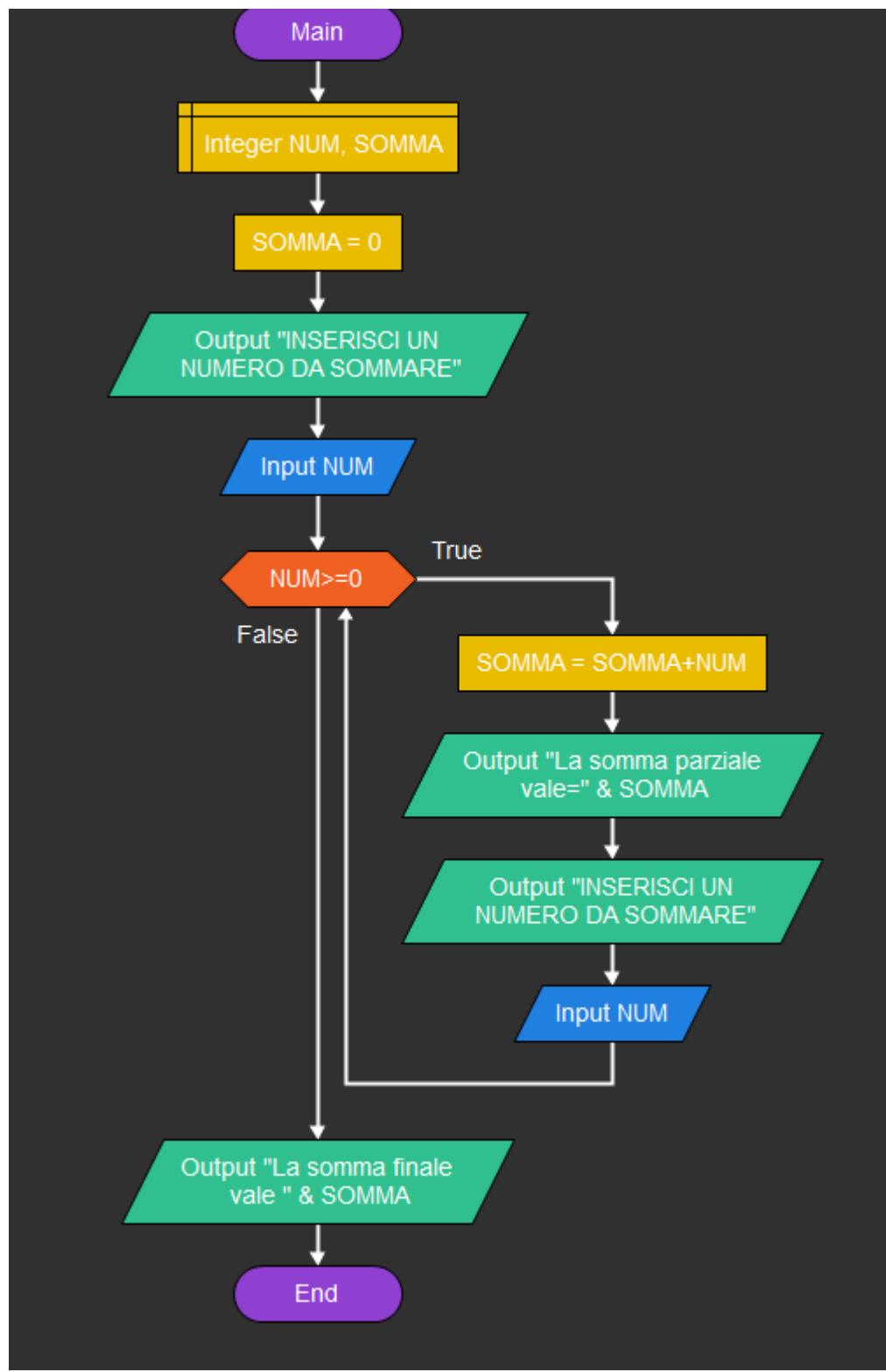
Scrivere un programma che legga due valori interi NUM1 e NUM2 e verifichi se l'utente indovina NUM1=5 && NUM2=7;



Scrivere un programma che legga due valori interi NUM1 e NUM2 e verifichi se l'utente indovina almeno uno di questi due numeri NUM1=5 || NUM2=7;

STRUTTURA ITERATIVA PRECONDIZIONALE

Il programma svolge la somma solo dopo aver verificato che i valori inseriti siano positivi o uguali a zero.



STRUTTURA ITERATIVA POSTCONDIZIONALE

Creare un programma che faccia la somma tra numeri inseriti da tastiera, quando questa è maggiore di 10 interrompe mostrando il valore della somma.

