

**ITIS-LS "Francesco Giordani" Caserta**

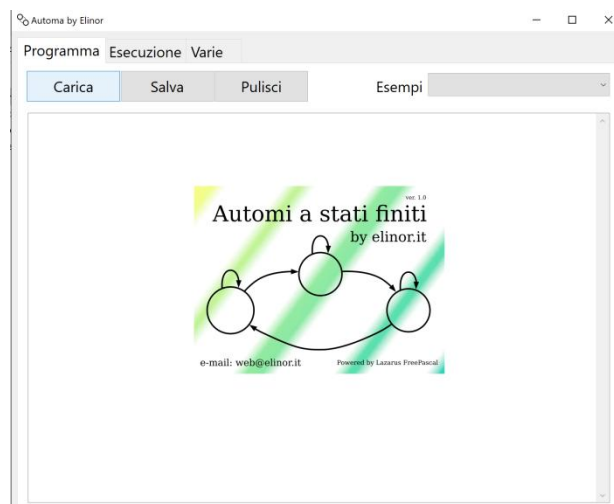
**prof. Ennio Ranucci**

**a.s. 2023-2024**

*Simulatore di automi a stati finiti (by ing. Stefano Bertoncello)*



<https://www.elinor.it/elinor.p?cid=PagStatic&pid=Automa&lid=IT>



Alla pagina web [elinor.it](http://elinor.it) trovate il Simulatore a stati finiti dell'ing. Stefano Bertoncello, scaricate il file: automa.exe

L'applicazione ha già due esempi: *Ascensore e Distributore*

Le **istruzioni** devono essere scritte nella forma:  
(<stato corrente>, <input>, <stato futuro>, <output>)

### Stato

Può essere costituito da uno o più dei seguenti caratteri:

*abcdefghijklmnopqrstuvwxy ABCDEFGHIJKLMNOPQRSTUVWXYZ 0123456789*

### Input e Output

Possono essere costituiti da uno o più dei seguenti caratteri:

*abcdefghijklmnopqrstuvwxy ABCDEFGHIJKLMNOPQRSTUVWXYZ 0123456789*

oppure uno solo dei seguenti caratteri *!#\$%&'\*+-. /:;<=>?@[\\]^\_{|}\_*

Nella parte alta si vedono le **transizioni di stato e gli output** corrispondenti.

Viene evidenziata l'ultima transizione eseguita.

Nella parte inferiore sono rappresentati i possibili **input**

Cliccando sull'input si invia all'automa l'evento corrispondente

Nella parte sinistra è riportato l'**elenco degli stati**, ed è evidenziato lo stato corrente.

Cliccando su uno stato è possibile modificare lo stato corrente.

E' possibile generare un file .dot per la **generazione del grafo** corrispondente alla macchina.

Aprite con "blocco note" il file.dot generato e copiate il contenuto.

Alla pagina web: <https://edotor.net/> trovate il visualizzatore dei file .dot

Incollate il vostro dot per visualizzare il grafo

```
1 digraph finite_state_machine
2 {
3 graph [fontname = "Verdana"];
4 node [fontname = "Verdana"];
5 edge [fontname = "Verdana"];
6 rankdir=LR;
7 size="8,8"
8 node [shape = doublecircle];
9 ROSSO;
10 node [shape = circle];
11 ROSSO -> VERDE [ label = "SegnaleCambioColore,ROSSO"];
12 VERDE -> GIALLO [ label = "SegnaleCambioColore,VERDE"];
13 GIALLO -> ROSSO [ label = "SegnaleCambioColore,GIALLO"];
14 }
15
```

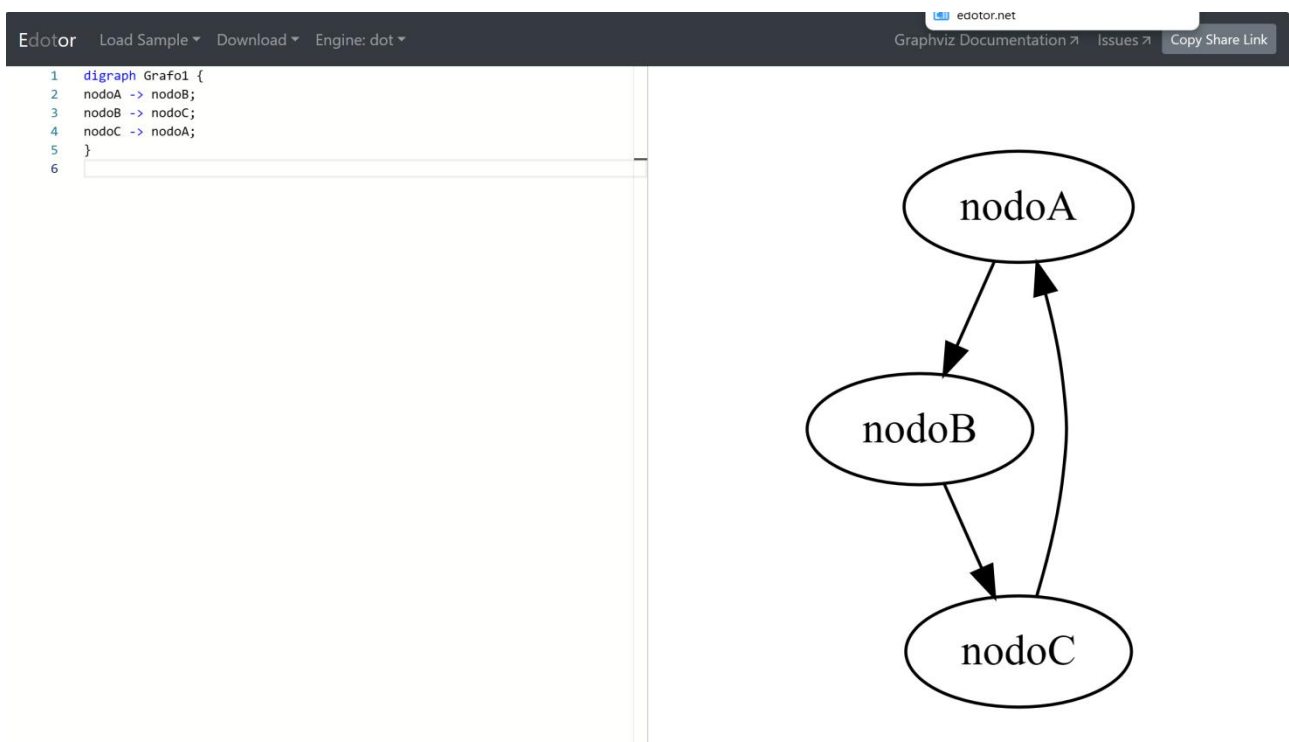
The diagram shows three states: ROSSO (left), VERDE (top), and GIALLO (right). ROSSO and GIALLO are double circles, while VERDE is a single circle. Transitions are: ROSSO to VERDE (labeled 'SegnaleCambioColore,ROSSO'), VERDE to GIALLO (labeled 'SegnaleCambioColore,VERDE'), and GIALLO to ROSSO (labeled 'SegnaleCambioColore,GIALLO').

**Il linguaggio DOT** è un linguaggio creato da AT&T che permette di descrivere dei grafi in formato testuale. Una volta descritto, posso rappresentare graficamente il grafo o la mappa concettuale in un'apposita applicazione di visualizzazione.

Per la visualizzazione di diagrammi di reti (o grafi) si utilizza principalmente l'applicazione **Graphviz**.

Semplice grafo orientato descritto con il linguaggio DOT

```
digraph Grafo1 {  
nodoA -> nodoB;  
nodoB -> nodoC;  
nodoC -> nodoA;  
}
```



Per creare lo stesso grafo “non orientato”:

```
graph Grafo2 {  
nodoA -- nodoB;  
nodoB -- nodoC;  
nodoC -- nodoA;  
}
```

ITIS-LS "Francesco Giordani" Caserta

Anno scolastico:

Classe sez. spec. Informatica e telecomunicazioni– articolazione Informatica

Data:

Numero progressivo dell'esercizio: es1

Versione: 1.0

Programmatore/i:

Sistema Operativo: Windows 10

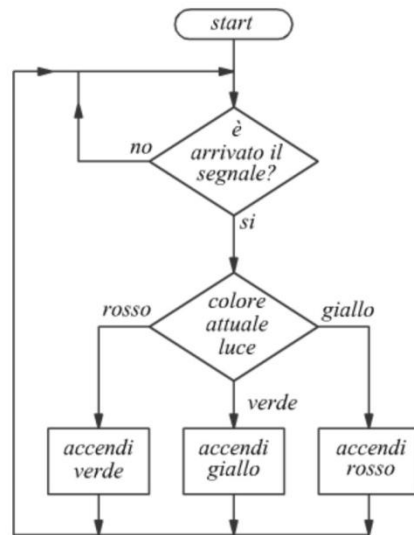
Applicazione: Automa.exe

Obiettivo didattico: saper programmare un automa a stati finiti in ambiente Elinor

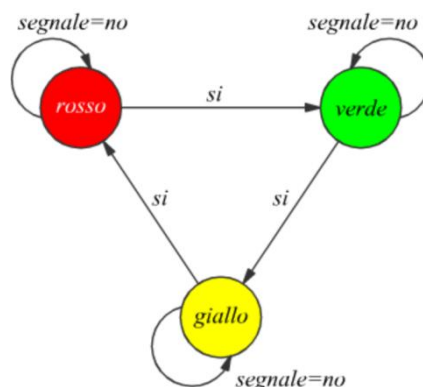
Obiettivo del programma:

Costruire un semaforo, cioè un dispositivo che accende una luce di colore diverso ogni volta che arriva un segnale di sincronismo. Il semaforo può accendere e spegnere una delle luci gialla rossa e verde.

Lo schema di flusso di un semaforo è indicato nel disegno seguente (by Edutecnica):



grafo:



Il *semaforo* quando è in funzione si può trovare in uno di 3 stati possibili (consideriamo non in funzione il semaforo lampeggiante):

1. Luce Rossa
2. Luce Gialla
3. Luce Verde

L'output del semaforo è l'accensione della luce corrispondente allo stato in cui si trova, ad esempio lo stato Luce Verde accenderà il disco verde , lo stato Luce Gialla il disco giallo e lo stato Luce Rossa il disco rosso.

Programma per il simulatore Elinor:

(ROSSO, SegnaleCambioColore, VERDE,ROSSO)

(VERDE, SegnaleCambioColore, GIALLO,VERDE)

(GIALLO, SegnaleCambioColore,ROSSO,GIALLO)

Stato corrente	Input	Stato futuro	Output
ROSSO	SegnaleCambioColore	VERDE	ROSSO
VERDE	SegnaleCambioColore	GIALLO	VERDE
GIALLO	SegnaleCambioColore	ROSSO	GIALLO
ROSSO	SegnaleCambioColore	VERDE	ROSSO

Stato corrente

ROSSO  
VERDE  
GIALLO

SegnaleCambioColore

ITIS-LS "Francesco Giordani" Caserta

Anno scolastico:

Classe sez. spec. Informatica e telecomunicazioni– articolazione Informatica

Data:

Numero progressivo dell'esercizio: es2

Versione: 1.0

Programmatore/i:

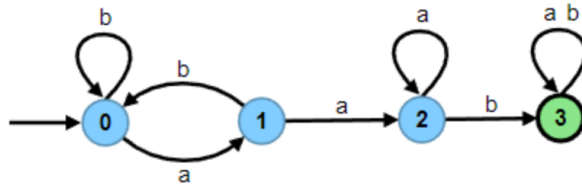
Sistema Operativo: Windows 10

Applicazione: Automa.exe

Obiettivo didattico: saper programmare un automa a stati finiti in ambiente Elinor

Obiettivo del programma:

Costruire un automa che riconosce le stringhe che contengono la sottostringa "aab"  
(esempio <https://www.andreaminini.org/sistemi/automi/esempi-di-automi-a-stati-finiti>).



WWW.ANDREAMININI.ORG

(0, a, 1,stringa-aab-assente)

(0, b, 0,stringa-aab-assente)

(1, a, 2,stringa-aab-assente)

(1, a, 2,stringa-aab-assente)

(1, b, 0,stringa-aab-assente)

(2, a, 2,stringa-aab-assente)

(2, b, 3,string-aab-trovata)

(3, a, 3,stringa-aab-trovata)

(3, b, 3,stringa-aab-trovata)

```
1 digraph finite_state_machine
2 {
3 graph [fontname = "Verdana"];
4 node [fontname = "Verdana"];
5 edge [fontname = "Verdana"];
6 rankdir=LR;
7 size="8,8"
8 node [shape = doublecircle];
9 0;
10 node [shape = circle];
11 0 -> 1 [ label = "a,assente"];
12 0 -> 0 [ label = "b,assente"];
13 1 -> 2 [ label = "a,assente"];
14 1 -> 2 [ label = "a,assente"];
15 1 -> 0 [ label = "b,assente"];
16 2 -> 2 [ label = "a,assente"];
17 2 -> 3 [ label = "b,trovata"];
18 3 -> 3 [ label = "a,trovata"];
19 3 -> 3 [ label = "b,trovata"];
20 }
21 |
```