

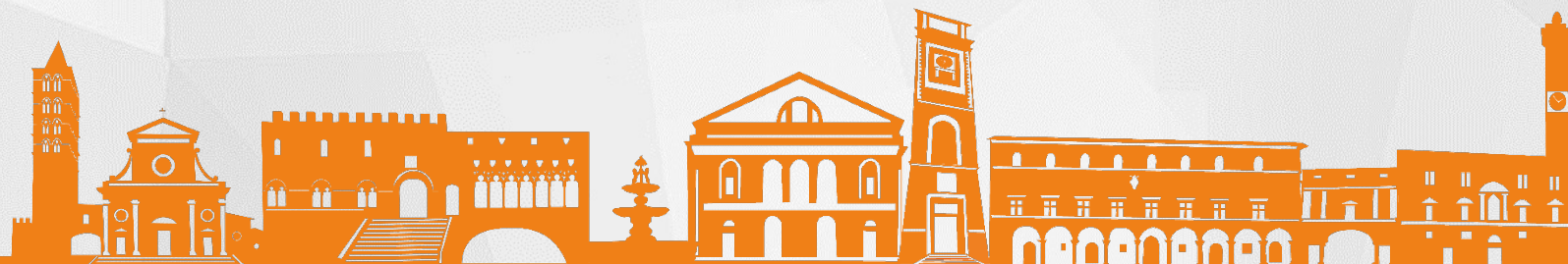
TEST AUTOMATICI SU AUTOMI A STATI FINITI E MACCHINE DI TURING CON CODERUNNER

Adolfo Casagrande
ISIS. A. Malignani

adolfo.casagrande@malignani.ud.it

Luciano Dereani
ISIS. A. Malignani

luciano.dereani@malignani.ud.it



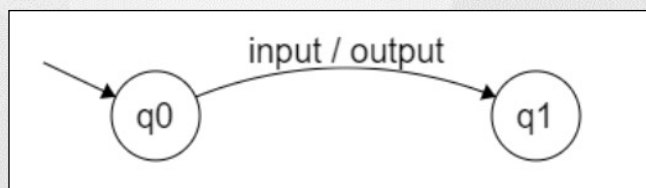
Introduzione

- Esigenza didattica classi 5[^] Liceo Scienze Applicate
- Argomenti: *automi a stati finiti* e *macchine di turing*
- **Editor** per lo svolgimento degli esercizi (grafico)
- Strumento di **correzione automatica**

Problema – Automi a stati finiti

Biglietti da distributore automatico

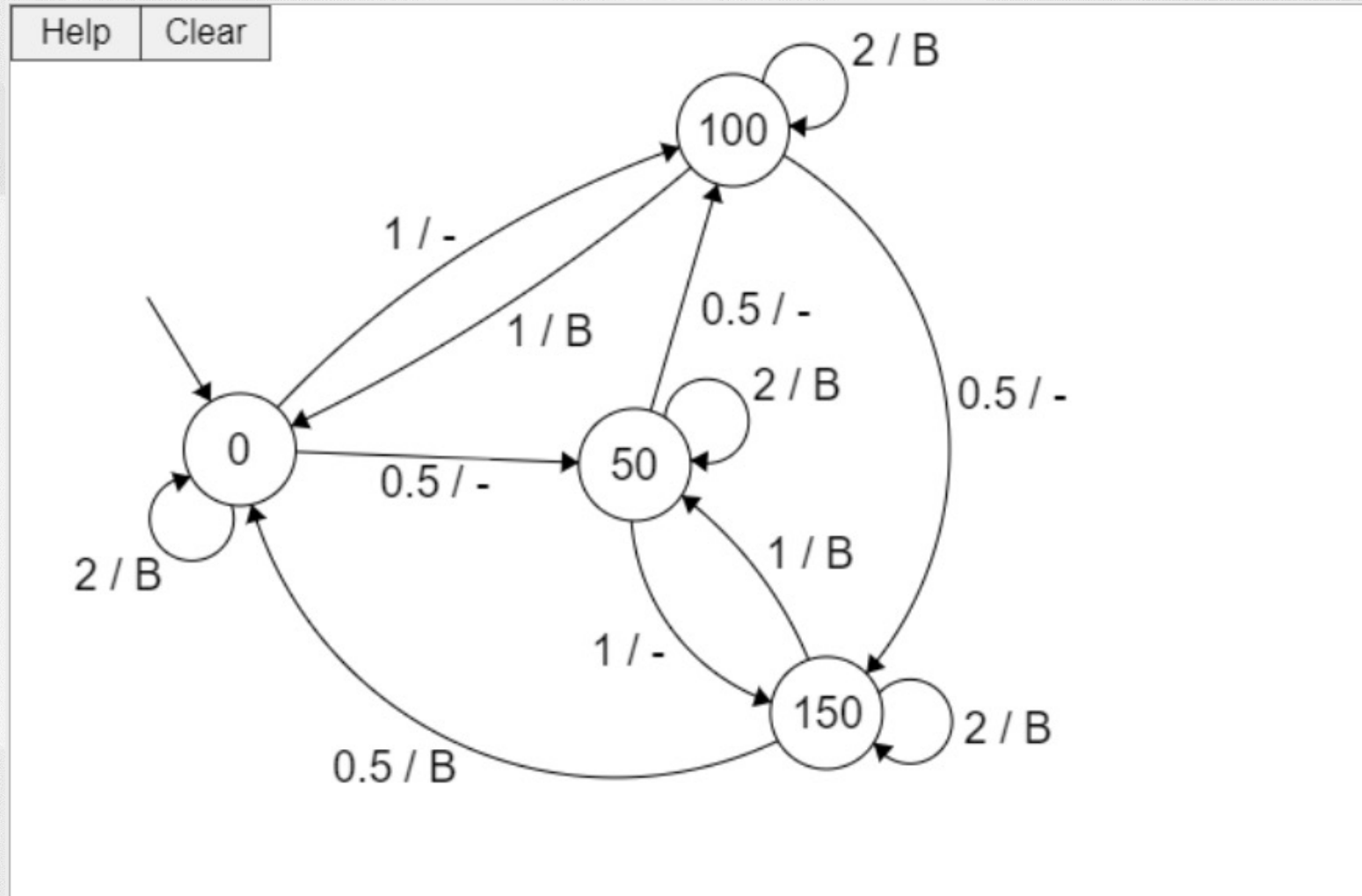
Si vuole progettare un automa a stati finiti che controlli il funzionamento di un distributore automatico di biglietti da 2€; il distributore accetta soltanto monete da 50 centesimi di euro, 1€ o 2€, inserite una alla volta, senza restituire alcun resto ma utilizzandolo per l'emissione del biglietto successivo



Automa di Mealy



Soluzione – Distributore biglietti



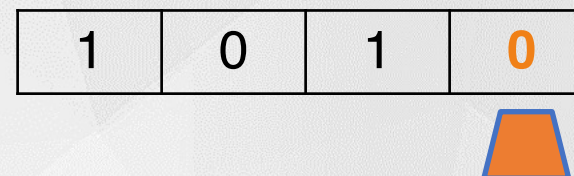
Problema – Macchina di Turing

Controllo Bit di parità

Si vuole progettare una MdT che, dato un nastro iniziale contenente una sequenza binaria, termini la sua esecuzione aggiungendo alla sequenza data il bit di parità, che sarà pari a 0 se il numero di 1 della sequenza è pari, 1 se è dispari.

`qp, I, U, qs, S`

Formato istruzioni MdT



Soluzione – Bit di parità

$Q_i = q_P$

$q_P, 0, 0, q_P, >$

$q_P, 1, 1, q_D, >$

$q_P, _, 0, q_f, -$

$q_D, 0, 0, q_D, >$

$q_D, 1, 1, q_P, >$

$q_D, _, 1, q_f, -$

Problema – Utilizzo strutture di controllo

Programmi ricorsivi

Realizzare un programma in python che calcoli il **fattoriale** di un numero utilizzando la **ricorsione** (e non l'iterazione).

$fatt(n) = 1 \quad se \ n \leq 0$

$fatt(n) = n * fatt(n-1) \quad se \ n > 0$

Problema – For o While

Programmi iterativi con for / while

Dato il seguente programma scritto utilizzando la struttura di controllo **for**, scrivere un programma equivalente che utilizzi il **while**...

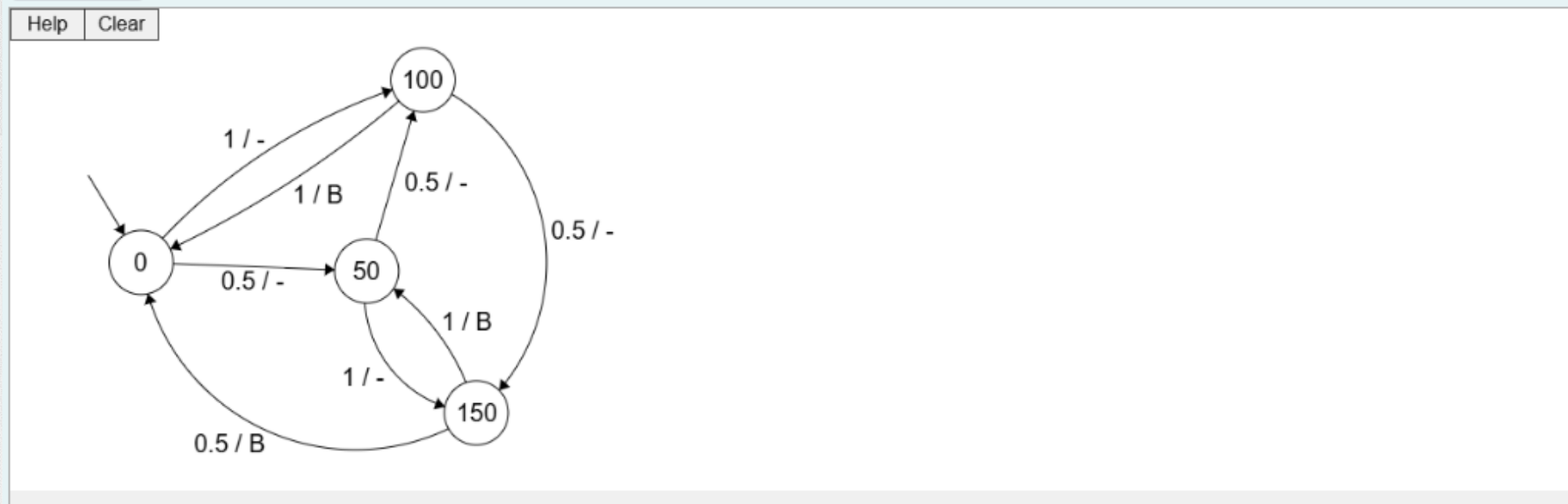
for  **while** (o viceversa)

Soluzione

- plugin Code-Runner di Moodle
- **personalizzazione tipologia domande**
- Ulteriori potenzialità: validazione automatica nell'uso di strutture di controllo in python



Esempio di validazione automatica - 1



Verifica risposta

	Test	Input	Expected	Got	
✓	1	0.5 0.5 1 0.5	['-', '-', 'B', '-']	['-', '-', 'B', '-']	✓
✗	2	2 1 0.5 2 1 0.5 0.5 0.5	['B', '-', '-', 'B', 'B', '-', '-', 'B']	[] STOP 0	✗
✗	3	1 1 0.5 1 0.5 0.5 0.5 2 2 1 0.5 2 1 0.5	['-', 'B', '-', '-', 'B', '-', '-', 'B', 'B', 'B', '-', 'B', '-', 'B']	['-', 'B', '-', '-', 'B', '-', '-', 'B', '-', 'B', '-', 'B', '-', 'B'] STOP 100	✗

Esempio di validazione automatica - 2

$Q_i = qP$

$qP, 0, 0, qP, >$

$qP, 1, 1, qD, >$

$qP, _, 0, qf, -$

$qD, 0, 0, qD, >$

$qD, 1, 1, qP, >$

$qD, _, 0, qf, -$

Verifica risposta

	Input	Expected	Got
✓	1 0 0 1	Nastro di ingresso: ['1', '0', '0', '1'] Nastro di uscita: ['1', '0', '0', '1', '0']	Nastro di ingresso: ['1', '0', '0', '1'] Nastro di uscita: ['1', '0', '0', '1', '0']
✗	0 1 1 0 1	Nastro di ingresso: ['0', '1', '1', '0', '1'] Nastro di uscita: ['0', '1', '1', '0', '1', '1']	Nastro di ingresso: ['0', '1', '1', '0', '1'] Nastro di uscita: ['0', '1', '1', '0', '1', '0']
✗	1 0 0 1 0 1 1 1 0	Nastro di ingresso: ['1', '0', '0', '1', '0', '1', '1', '1', '0'] Nastro di uscita: ['1', '0', '0', '1', '0', '1', '1', '1', '0', '1']	Nastro di ingresso: ['1', '0', '0', '1', '0', '1', '1', '1', '0'] Nastro di uscita: ['1', '0', '0', '1', '0', '1', '1', '1', '0', '0']
✓	0	Nastro di ingresso: ['0'] Nastro di uscita: ['0', '0']	Nastro di ingresso: ['0'] Nastro di uscita: ['0', '0']

Soluzione – Fattoriale ricorsivo

Test	Result
print(fatt(0))	1
print(fatt(3))	6

Answer: (penalty regime: 0 %)

```

1 def fatt(n):
2     s = 1
3     for i in range(n, 0, -1):
4         s = s * i
5     return s
    
```



Verifica risposta

	Test	Expected	Got	
✗	print(fatt(0))	1	***Run error*** Non puoi usare un ciclo for!	✗

Answer: (penalty regime: 0 %)

```

1 def fatt(n):
2     """ fattoriale ricorsivo """
3     if (n == 0):
4         return 1
5
6     if (n > 0):
7         return n * fatt(n-1)
    
```



	Test	Expected	Got	
✓	print(fatt(0))	1	1	✓
✓	print(fatt(1))	1	1	✓
✓	print(fatt(3))	6	6	✓
✓	print(fatt(5))	120	120	✓
✓	print(fatt(8))	40320	40320	✓

Passed all tests! ✓

Come creare un prototipo – passo 1

Creare una nuova domanda
appartenente alla categoria
CR_CUSTOM_PROTOTYPES

Deposito delle domande

Filtra Tutto delle seguenti condizioni:

Filtra Categoria Scrivi o seleziona... CR_CUSTOM_PROTOTYPES (2) X

AND

Filtra Visualizza domande nascoste No

+ Aggiungi condizione

Crea una nuova domanda... Reimposta colonne Visualizzazione del testo della domanda nell'elenco di

<input type="checkbox"/>	Domanda	Azioni	Stato	Versione	Comme
	Nome della domanda / Codice identificativo				
<input type="checkbox"/>	PROTOTYPE_ASF automa	Modifica	Pronta	v3	0
<input type="checkbox"/>	PROTOTYPE_MDT automa	Modifica	Pronta	v4	0

Come creare un prototipo – passo 2

- Attivare il checkbox *Customisation* della sezione CodeRunner *question type*
- **Personalizzare il codice di controllo** impostando il campo Template nella sezione Customisation:

PYTHON

```
1 import json
2
3 class Automaton:
4     def __init__(self, grafo, splitChar):
5         # Inizializza gli stati e le transizioni
6         self.grafo = grafo # Dizionario contenente i nodi e le liste di adiacenza
7         self.current_state = list(grafo.keys())[0] # Inizia dallo stato 0 (nodo "A")
8         self.current_uscita = ""
9         self.splitChar = splitChar
10
11     def reset(self):
12         self.current_state = list(self.grafo.keys())[0]
13         self.current_uscita = ""
14
15     #Separa l'etichetta della transizione in input e output utilizzando '/' come separatore.
16     def get_ingresso(self, label):
17         parts = label.split(self.splitChar)
18         return parts[0].strip() if len(parts) > 0 else label.strip
19
20 ..
```

Come creare un prototipo – passo 3

Impostare i campi nella sezione Advanced Customisation:

- **Is prototype?** a Yes (user defined)
- Question type al **nome** che si vuole assegnare a questo tipo di domanda

Advanced customisation

Prototyping



Is prototype?

Yes (user defined) ▾

Question type

directed_graph_asf

Sandbox



TimeLimit (secs)

MemLimit (MB)

Parameters

Languages



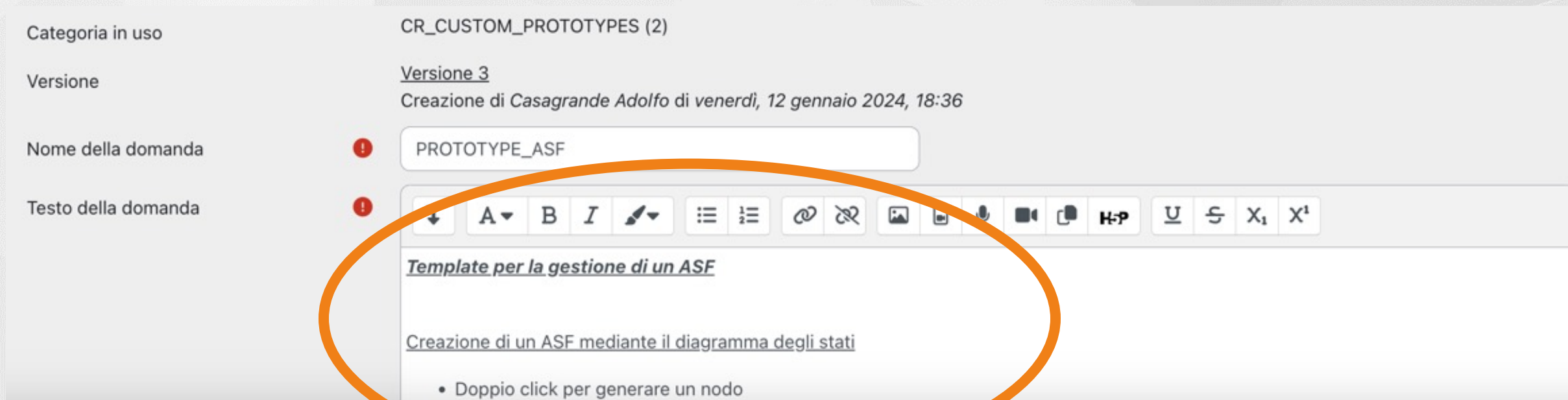
Sandbox language

python3

Ace language

Come creare un prototipo – passo 4

Impostare nella sezione Generale la **descrizione** del prototipo che funge da aiuto in fase di creazione del test



The screenshot shows the Moodle question creation interface. The 'Testo della domanda' field is highlighted with an orange oval. It contains the following text:

PROTOTYPE_ASF

Template per la gestione di un ASF

Creazione di un ASF mediante il diagramma degli stati

- Doppio click per generare un nodo

Creazione del TEST

- Selezione del prototipo
- Testo del problema
- Inserimento della soluzione
- Specifica dei casi di test

A screenshot of the Moodle question type selection interface. On the left, a list of question types is shown with a question mark icon next to each: Question type, Customisation, Answer box, Submit buttons, Stop button, Feedback, Marking, and Template params. On the right, a dropdown menu is open, displaying a list of question type options: Undefined (checked), MdT, c_function, c_program, cpp_function, cpp_program, directed_graph, directed_graph_asf (highlighted in blue), java_class, java_method, java_program, multilanguage, nodejs, octave_function, and pascal_function. A downward arrow is visible at the bottom of the dropdown menu.

Question type ?

Question type ?

Esempio casi di test – automa a stati finiti

▼ Test cases

Test case 1



1

Standard Input



0.5 1 1 0.5

Expected output



['-', '-', 'B', '-']


Extra template data



Creazione domande – Funzioni ricorsive

- Specifica dei vincoli (*global extra*)

▼ **Global extra**

Global extra		<pre>banned = ["For", "While"]</pre>
--------------	---	--------------------------------------

Requisiti

- Plug in CodeRunner [<https://coderunner.org.nz/>]
- Prototipi: *directed_graph_asf*, *MdT*, *python3 (con vincoli)*
[Download e import in moodle]
- Competenze di programmazione
[necessarie SOLO se si vuole personalizzare la tipologia dei test]



Conclusioni

La personalizzazione di CodeRunner permette di automatizzare l'apprendimento e la valutazione di argomenti complessi come:

- Automi a stati finiti
- Macchine di Turing

Vantaggi:

- Per docenti: **maggiore efficacia** nell'insegnamento grazie all'uso di test automatici
- Per studenti: strumento interattivo con **feedback immediato** e dettagliato

Grazie per l'attenzione

