

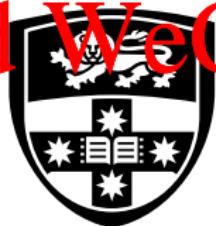
COMP2022: Formal Languages and Logic

2018 Semester 2, Week 4

Assignment Project Exam Help

<https://eduassistpro.github.io>

Add WeChat `edu_assist_pro`



THE UNIVERSITY OF
SYDNEY

COMMONWEALTH OF AUSTRALIA

Copyright Regulations 1969

Assignment Project Exam Help

WARNING

This
on be
Cop

<https://eduassistpro.github.io>

The material in this communication may be subject
under the Act. Any further copying or communicat
material by you may be subject of copyright protec

Add WeChat edu_assist_pro

Do not remove this notice.

OUTLINE

Assignment Project Exam Help

- ▶ Lambda Calculus
 - ▶ Concepts (Operators, FV, Reductions)

<https://eduassistpro.github.io>

- ▶ Automata Theory
 - ▶ Background concepts
 - ▶ DFA
 - ▶ Regular languages
 - ▶ NFA

Add WeChat edu_assist_pro

OPERATORS

Assignment Project Exam Help

► Application

► Notation: $(A \cdot B)$

<https://eduassistpro.github.io>

Add WeChat edu_assist_pro

OPERATORS

Assignment Project Exam Help

► Application

► Notation: $(A \cdot B)$

<https://eduassistpro.github.io>

► Abstraction

► $(\lambda x.M)$
► Variable x is abstracted in expression

► Right associative:

$$(\lambda abcde.M) = (\lambda a.(\lambda b.(\lambda c.(\lambda d.(\lambda e.(\lambda f.M))))))$$

FREE VARIABLES

Assignment Project Exam Help

Formally, the set of free variables of an expression is defined inductively like this:

Formally, the set of free variables of an expression is defined inductively like this:

<https://eduassistpro.github.io>

$$FV(MN) = FV(M) \cup FV(N) \quad M, N \in \text{ssions}$$

$$FV(\lambda x.M) = FV(M) - \{x\}$$

Add WeChat edu_assist_pr

Any variable in an expression that is not free, is bound.

REWRITING

Assignment Project Exam Help

- <https://eduassistpro.github.io>
 - e.g. Add WeChat edu_assist_pro
 - $(xyz\lambda x.(zxz))[x := A] \equiv (Ayz)$
 - $(xyz\lambda x.(zxz))[y := B] \equiv (xBz\lambda x.(zxz))$
 - $(xyz\lambda x.(zxz))[z := C] \equiv (xyC\lambda x.(Cxz))$

α -REDUCTION

Assignment Project Exam Help

$$\rightarrow \lambda x. M = \lambda y. (M[x := y])$$

- ▶ <https://eduassistpro.github.io>
 - ▶ **Add WeChat edu_assist_pro**
 - ▶ y must be a new variable
 - ▶ Do not choose a symbol that is already in the environment
 - ▶ It's usually easiest to start with the innermost λ

β -REDUCTION

Assignment Project Exam Help



<https://eduassistpro.github.io>

- ▶ Note: the free occurrences of x i
occurrences which are bound to the

Add WeChat edu_assist_pro

- ▶ Do α -reductions first if necessary

η -REDUCTION

Assignment Project Exam Help

- If x is *not* free in M (i.e. $x \not\in FV(M)$), then we can write:

<https://eduassistpro.github.io>

- Add WeChat edu_assist_pro
- Reduces an abstraction directly

NAMING

Assignment Project Exam Help

- ▶ If a variable is free, but there is also λ with the same label

▶

▶ <https://eduassistpro.github.io>

To fix them:

- ▶ Add WeChat edu_assist_pr
- ▶ Apply α -reduction repeatedly to change the labels
- ▶ Always rename to a label not already in use
- ▶ Work from the innermost λ to the outermost one

FIXED POINT COMBINATORS

Assignment Project Exam Help

A *fixed point combinator* is a combinator which has a fixed point.

We see <https://eduassistpro.github.io/>

Add WeChat edu_assist_pro
i.e. some input X exists which, when applied again,

FIXED POINT THEOREM

1. $\forall F \exists X FX = X$

- ▶ All functions have a fixed point
- ▶ Proven last week

2. <https://eduassistpro.github.io>

$$\forall F F(YF)$$

Add WeChat edu_assist_pro

- ▶ $Y = \lambda f. (\lambda x. f(xx))(\lambda x. f(xx))$
- ▶ i.e. for any function F , YF is a fixed point of F .
- ▶ Proven last week
- ▶ Very useful for recursion

RECURSION

Recursive functions often look like this:

Assignment Project Exam Help

$$F = \lambda xyz.(\text{condition})(\text{base case})(\text{recursive case calls } F)$$

If we ne

We ca

<https://eduassistpro.github.io>

We can define it like this instead:

Add WeChat edu_assist_pr

$$H = (\lambda fxyz.(\text{condition})(\text{base case}$$
$$F = Y H$$

$$H = \left(\lambda fxyz.(\text{condition})(\text{base case})(\text{recursive case calls } f) \right)$$

Assignment Project Exam Help

Eval

$F \ a \ b \ c$ <https://eduassistpro.github.io>

$$= H (Y H) a b c \qquad \qquad \qquad H)$$

Add WeChat edu_assist_pro

$$= \left(\lambda xyz.(\text{condition})(\text{base})(\text{call(s) to } (Y H)) \right) a b c$$

Notice that we managed to call $(Y H) = F$ from within F

- recursion!

REDUCING THE Y COMBINATOR

Assignment Project Exam Help

We don't need to reduce ($Y H$) directly



uce

► <https://eduassistpro.github.io>

Add WeChat edu_assist_pro

EXAMPLE: LAST WEEK'S TUTORIAL QUESTIONS

We wanted a function that would build a list $\{x_1 \dots x_n\}$ from an expression like $(F n x_1 \dots x_n)$

Assignment Project Exam Help

- ▶ Recurse n times

- ▶ <https://eduassistpro.github.io>

$H = \lambda f t n. (ISZERO \vartheta n) \ t \ (\lambda e f ($
 $F = Y H NIL$

- ▶ Prepend the next argument to the list with $(CONSet)$, and recursively call $f \ list \ (n - 1)$ with the remaining arguments

EXAMPLE: LAST WEEK'S TUTORIAL QUESTIONS

Assignment Project Exam Help

$$\stackrel{F \ 3 \ a \ b \ c}{=} \stackrel{Y \ H \ NIL \ 3 \ a \ b \ c}{=}$$

$$= H$$

$$= (\stackrel{(Y \ H) \ NIL \ 3}{\text{https://eduassistpro.github.io}})$$

$$= \dots$$

$$= \dots = FALSE \ NIL \ (\lambda e. (Y \ H)) \ (CO \ a \ b \ c)$$

$$= \dots = (\lambda e. (Y \ H) \ (CONS \ e \ NIL)) \ ($$

$$= \dots = (Y \ H) \ (CONS \ a \ NIL) \ (PRE \ a \ b \ c)$$

$$= \dots = (Y \ H) \ (CONS \ a \ NIL) \ 2 \ b \ c$$

$$= \dots = (Y \ H) \ (CONS \ b \ (CONS \ a \ NIL)) \ 1 \ c$$

$$= \dots = (Y \ H) \ (CONS \ c \ (CONS \ b \ (CONS \ a \ NIL))) \ 0$$

$$= \dots = (CONS \ c \ (CONS \ b \ (CONS \ a \ NIL)))$$

Add WeChat **edu_assist_pro**

EXAMPLE: LAST WEEK'S TUTORIAL QUESTIONS

Reversing a list:

$$H = \lambda fab. (ISNIL a) b \mid f (TAIL a) (CONS (HEAD a) b))$$

$$F = \lambda a. Y H a NIL$$

$$F \{$$

$$= (\lambda a$$

<https://eduassistpro.github.io>

$$= Y H$$

$$= H (Y H) \{c, b, a\} NIL$$

$$= (\lambda fab. (ISNIL a) b \mid f (TAIL a) (CONS (HEAD a) b)) NIL$$

$$= \dots = (Y H) (TAIL \{c, b, a\}) (CONS (HE$$

$$= \dots = (Y H) \{b, a\} (CONS c NIL)$$

$$= \dots = (Y H) \{b, a\} \{c\}$$

$$= \dots = (Y H) \{a\} \{b, c\}$$

$$= \dots = (Y H) NIL \{a, b, c\}$$

$$= \dots = \{a, b, c\}$$

Add WeChat edu_assist_pro

NOTATION

We've mostly been using the = sign everywhere. This isn't strictly correct!

Assignment Project Exam Help

We co

- ▶ <https://eduassistpro.github.io/>
- ▶
- ▶ $M \rightarrow_{\beta} N$: M β -reduces to N
- ▶ $M \vdash_{\beta} N$: M β -reduces to N (i.e.)
- ▶ $M =_{\beta} N$: M is β -convertable to N

Add WeChat edu_assist_pro

We've avoided using these so far for the sake of simplicity

β -NORMAL FORM

An expression is in β -normal form if no β -reductions are available

Assignment Project Exam Help

Not all expressions have normal forms



Nor
<https://eduassistpro.github.io>

found by following the leftmost reduction

- ▶ $((\lambda x.xx)(\lambda x.xx))((\lambda ab.b))$ does not have a normal form because the leftmost reduction loops infinitely
- ▶ $(\lambda ab.b)((\lambda x.xx)(\lambda x.xx))$ has a normal form, $(\lambda b.b)$, even though following the reduction on the right would not have found it

β -EQUIVALENCE

Two expressions are β -equivalent if they have a common reduct.

Assignment Project Exam Help

- ▶ them
- ▶ <https://eduassistpro.github.io>
- ▶ $(\lambda xy.x)ab$ and $(\lambda xy.x)ac$ both reduce to a
 - ▶ Therefore they are β -equivalent
 - ▶ ... even though we have no way to reduce better
- ▶ We've been using this property often, without stating it

β -EQUIVALENCE

Assignment Project Exam Help

YF

$\beta \ YF$

<https://eduassistpro.github.io>

Turi

$\Theta \equiv (\lambda xy.y(xxy))(\lambda xy.y(xxy))$, which h

$\Theta F \rightarrow^* F(\Theta F)$

Add WeChat edu_assist_pro

There are an infinite number of fixed point combinators in untyped lambda calculus!

CHURCH-ROSSER THEOREM

If $M \xrightarrow{\beta} N$ and $M \xrightarrow{\beta} N_2$, then there exists some N_3 such that
 $N_1 \xrightarrow{\beta} N_3$ and $N_2 \xrightarrow{\beta} N_3$

This is <https://eduassistpro.github.io>

Add WeChat edu_assist_pro

We do need to be a little bit careful sometimes, as some reductions may not lead to the normal form, although they remain β -convertible to it!)

COMPUTATIONAL POWER

Lambda calculus can compute all the computable (recursive) functions.

Assignment Project Exam Help

- ▶ Proof is beyond the scope of this course
- ▶

<https://eduassistpro.github.io>

statements)

▶ If a normal form exists, we can find it (Normal Form Theorem)
▶ If a normal form exists, it is unique

Essentially, given any problem you can express with set theory (i.e. maths), then *if* it's possible to solve (compute) it, lambda calculus can do so.

COMPUTATIONAL POWER

Assignment Project Exam Help

Later in the course we will look at Turing machines.

The
C
Turi

<https://eduassistpro.github.io>

Add WeChat edu_assist_pro

The proof is relatively simple. If we have time at the end of the course, we will show you.

WHY?

Assignment Project Exam Help

If we can do just as much using the imperative paradigm, then why use the functional paradigm?

≡ is at <https://eduassistpro.github.io/>

- They have equivalent computational power
- They are not the *same*
- Some things are *easier* with each approach

Add WeChat `edu_assist_pro`

POSITIVE TRAITS OF FUNCTIONAL PARADIGM

- # Assignment Project Exam Help
- ▶ Often easier to write concurrent code (Church-Rosser)
 - ▶ Often easier to prove correctness
 - ▶ Lazy evaluation
 - ▶ <https://eduassistpro.github.io>
 - 'crawl' over our data. Performing different operations now just becomes arguments to these methods, entirely new functions.
 - ▶ Inherent immutability
 - ▶ Our functions define relationships between the existing structure and the one we want
 - ▶ Notions like undo/redo become almost trivial

Add WeChat `edu_assist_pro`

HYBRID LANGUAGES

Assignment Project Exam Help

NOT an either/or choice.

<https://eduassistpro.github.io>

Most languages blend imperative and functional

- ▶ e.g. Java, C++, Python,
- ▶ Add WeChat edu_assist_pro

OUTLINE

Assignment Project Exam Help

► Lambda Calculus

► Concepts (Operators, FV, Reductions)

<https://eduassistpro.github.io>

► Automata Theory

► Background concepts

► DFA

► Regular languages

► NFA

Add WeChat edu_assist_pro

ALAN TURING

Assignment Project Exam Help

► Founder of computer science, mathematician, philosopher, code breaker, visionary

uring
<https://eduassistpro.github.io>

- Church-Turing thesis
- Turing test
- Can machine think?
- The Imitation game
- Enigma code breaker

Add WeChat edu_assist_pro

NOAM CHOMSKY

- ▶ Linguist, philosopher, cognitive scientist, logician, historian, political critic and activist
- ▶ Chomsky Hierarchy: a containment hierarchy

<https://eduassistpro.github.io/>

Add WeChat edu_assist_pro



ALPHABET

Assignment Project Exam Help

For ex:

- ▶ <https://eduassistpro.github.io>
- ▶ All lower case letters: $\Sigma = \{a, b, \dots\}$
- ▶ Alphanumeric: $\Sigma = \{a, \dots, z, A, \dots, Z, 0, \dots, 9\}$
- ▶ ASCII, unicode
- ▶ Set of signals used by a protocol

Add WeChat `edu_assist_pro`

STRING

Assignment Project Exam Help

- ▶ A *string* is a finite sequence of symbols from Σ . For example
"01110", "alice", "Bnode.java"



▶ <https://eduassistpro.github.io>

- ▶ ε (epsilon) denotes the *empty string*

▶ $|\varepsilon| = 0$
if $a = abccade$ then $|a| = ?$

- ▶ xy = the concatenation of two strings

- ▶ x^n the string x repeated n times

Add WeChat edu_assist_pro

STRING

Assignment Project Exam Help

- ▶ A *string* is a finite sequence of symbols from Σ . For example
"01110", "alice", "Bnode.java"



▶ <https://eduassistpro.github.io>

- ▶ ε (epsilon) denotes the *empty string*

▶ $|\varepsilon| = 0$
if $a = abccade$ then $|a| = 7$

- ▶ xy = the concatenation of two strings

- ▶ x^n the string x repeated n times

Add WeChat edu_assist_pro

POWERS OF AN ALPHABET

Assignment Project Exam Help

Let

- ▶ <https://eduassistpro.github.io>
- ▶ $\Sigma^+ = \Sigma^1 \cup \Sigma^2 \cup \Sigma^3 \cup \dots$ (i.e. all stri

Add WeChat edu_assist_pro

LANGUAGES

Assignment Project Exam Help

- A language L over a alphabet Σ is a subset of Σ^* (i.e. $L \subseteq \Sigma^*$)
- Examples:

L

n allowed

<https://eduassistpro.github.io>

L

0s and 1s:

- $L = \{01, 10, 001, 0101, 0\}$
- \emptyset denotes the empty language
 - Beware: $\{\varepsilon\}$ is NOT \emptyset

THE MEMBERSHIP PROBLEM

Assignment Project Exam Help

Problem.

Given a string $w \in \Sigma^*$ and a language L over Σ , decide whether or not

<https://eduassistpro.github.io>

Example:

Let $w = 10011$

Does w belong to the language of strings with an even number of 0s and 1s?

Add WeChat edu_assist_pro

OUTLINE

Assignment Project Exam Help

- ▶ Lambda Calculus
 - ▶ Concepts (Operators, FV, Reductions)

<https://eduassistpro.github.io>

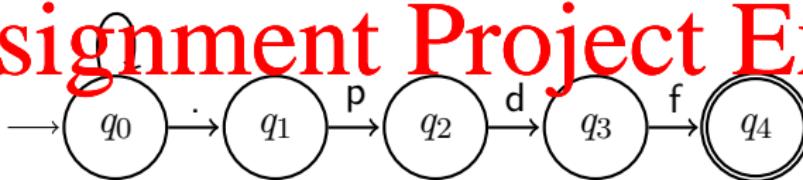
- ▶ Automata Theory
 - ▶ Background concepts
 - ▶ DFA
 - ▶ Regular languages
 - ▶ NFA

Add WeChat edu_assist_pro

INTRODUCTORY EXAMPLE

 $\Sigma \setminus \{.\}$

Assignment Project Exam Help



For
infor

<https://eduassistpro.github.io>

- ▶ Information is represented by the
- ▶ *Transition rules* (arrows) define state changes based on input
- ▶ *Start state* is denoted →
- ▶ *Accept state(s)* denoted by double circle
- ▶ The set of all possible input symbols is the *alphabet*, Σ

Add WeChat edu_assist_pro

INTRODUCTORY EXAMPLE

Assignment Project Exam Help



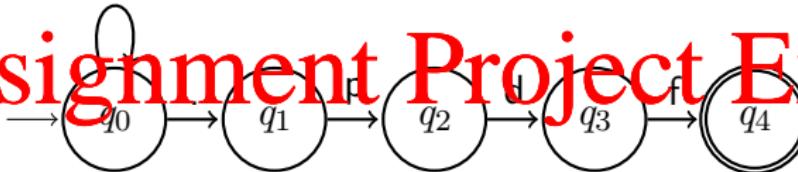
Give

<https://eduassistpro.github.io>

- ▶ Begin in the *start* state
- ▶ Follow one transition for each symbol in the input string
- ▶ After reading the entire input string:
 - ▶ The input is *accepted* if the automaton is in an *accept* state
 - ▶ The input is *rejected* if it is not

Add WeChat `edu_assist_pro`

INTRODUCTORY EXAMPLE

 $\Sigma \setminus \{.\}$ 

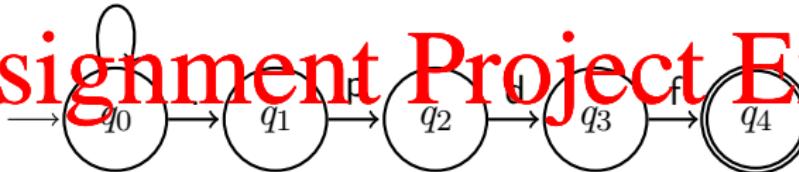
Exa

► <https://eduassistpro.github.io/>

- then q_2 for 'p', q_3 for 'd', q_4 for 'f'
- after all input is scanned, we ended in an accept state
therefore "example.pdf" is accepted

Add WeChat edu_assist_pro

INTRODUCTORY EXAMPLE

 $\Sigma \setminus \{.\}$ 

Exa

► <https://eduassistpro.github.io/>

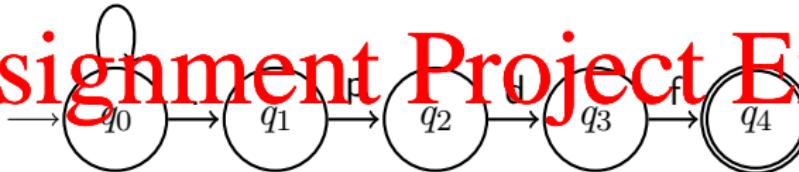
- then q_2 for 'p', q_3 for 'd', q_4 for 'f'
- after all input is scanned, we ended in an accept state
therefore "example.pdf" is accepted

How about:

- .pdf
- pdf
- example.pd
- example.pdf.pdf

Add WeChat edu_assist_pro

INTRODUCTORY EXAMPLE

 $\Sigma \setminus \{.\}$ 

Exa

► <https://eduassistpro.github.io/>

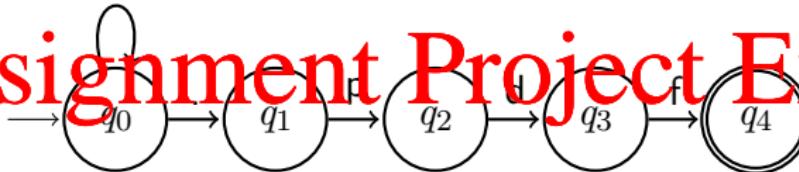
- then q_2 for 'p', q_3 for 'd', q_4 for 'f'
- after all input is scanned, we ended in an accept state
therefore "example.pdf" is accepted

How about:

- .pdf Yes
- pdf
- example.pd
- example.pdf.pdf

Add WeChat edu_assist_pro

INTRODUCTORY EXAMPLE

 $\Sigma \setminus \{.\}$ 

Exa

► <https://eduassistpro.github.io/>

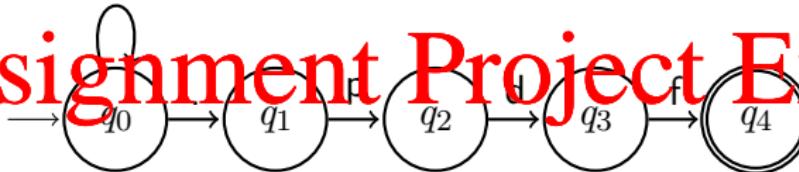
- then q_2 for 'p', q_3 for 'd', q_4 for 'f'
- after all input is scanned, we ended in an accept state
therefore "example.pdf" is accepted

How about:

- .pdf **Yes**
- pdf **No**
- example.pd
- example.pdf.pdf

Add WeChat edu_assist_pro

INTRODUCTORY EXAMPLE

 $\Sigma \setminus \{.\}$ 

Exa

► <https://eduassistpro.github.io/>

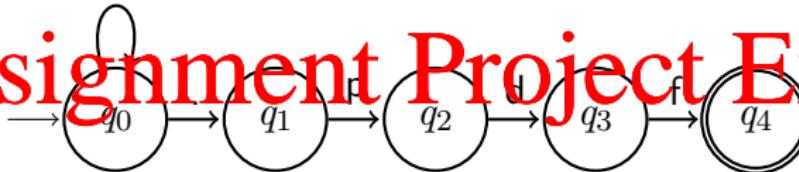
- then q_2 for 'p', q_3 for 'd', q_4 for 'f'
- after all input is scanned, we ended in an accept state
therefore "example.pdf" is accepted

How about:

- .pdf **Yes**
- pdf **No**
- example.pd **No**
- example.pdf.pdf

Add WeChat edu_assist_pro

INTRODUCTORY EXAMPLE

 $\Sigma \setminus \{.\}$ 

Exa

► <https://eduassistpro.github.io/>

- then q_2 for 'p', q_3 for 'd', q_4 for 'f'
- after all input is scanned, we ended in an accept state
therefore "example.pdf" is accepted

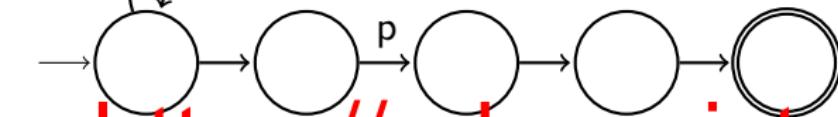
How about:

- .pdf **Yes**
- pdf **No**
- example.pdf **No**
- example.pdf.pdf **No(!)**

Add WeChat edu_assist_pro

INTRODUCTORY EXAMPLE

Assignment Project Exam Help



Recall <https://eduassistpro.github.io>

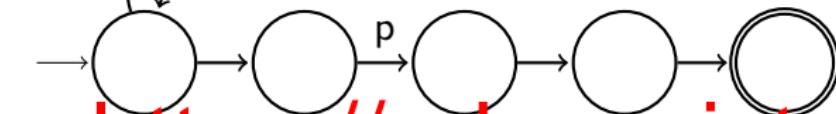


Add WeChat edu_assist_pro

What information is represented by state

INTRODUCTORY EXAMPLE

Assignment Project Exam Help



Recall <https://eduassistpro.github.io>



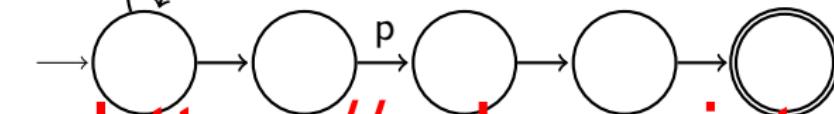
Add WeChat edu_assist_pro

What information is represented by state

- q_1 : We have scanned any number of letters, followed by ''

INTRODUCTORY EXAMPLE

Assignment Project Exam Help



Recall <https://eduassistpro.github.io>



Add WeChat edu_assist_pro

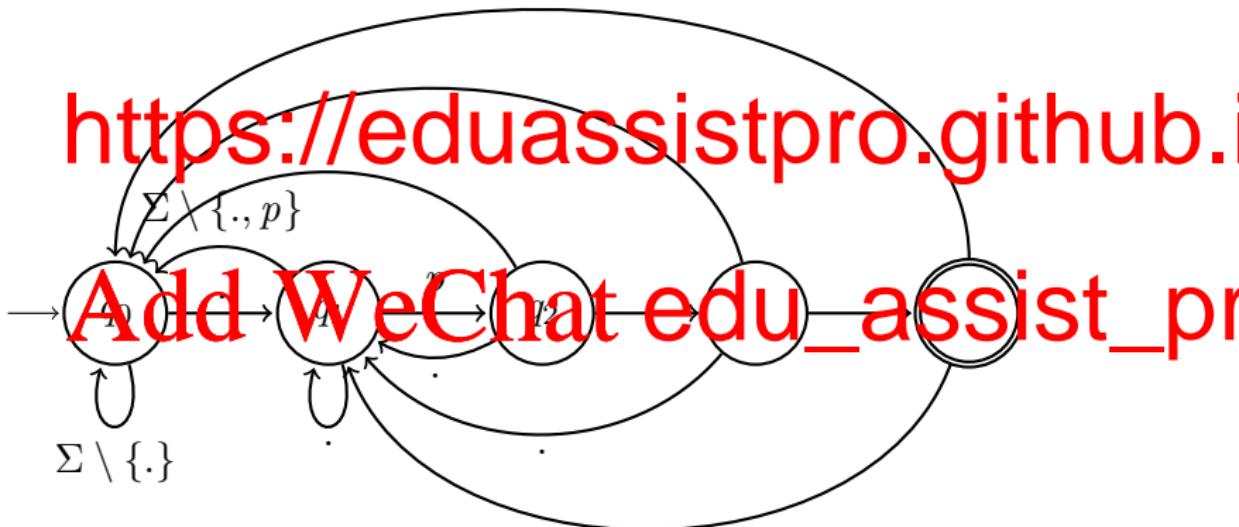
What information is represented by state

- ▶ q_1 : We have scanned any number of letters, followed by ‘.’
- ▶ q_2 : We have scanned any number of letters, followed by ‘.p’

OUR FIRST VALID DFA

A properly defined DFA will have a transition for every symbol, from every state:

Assignment Project Exam Help

 $\Sigma \setminus \{.\}$ 

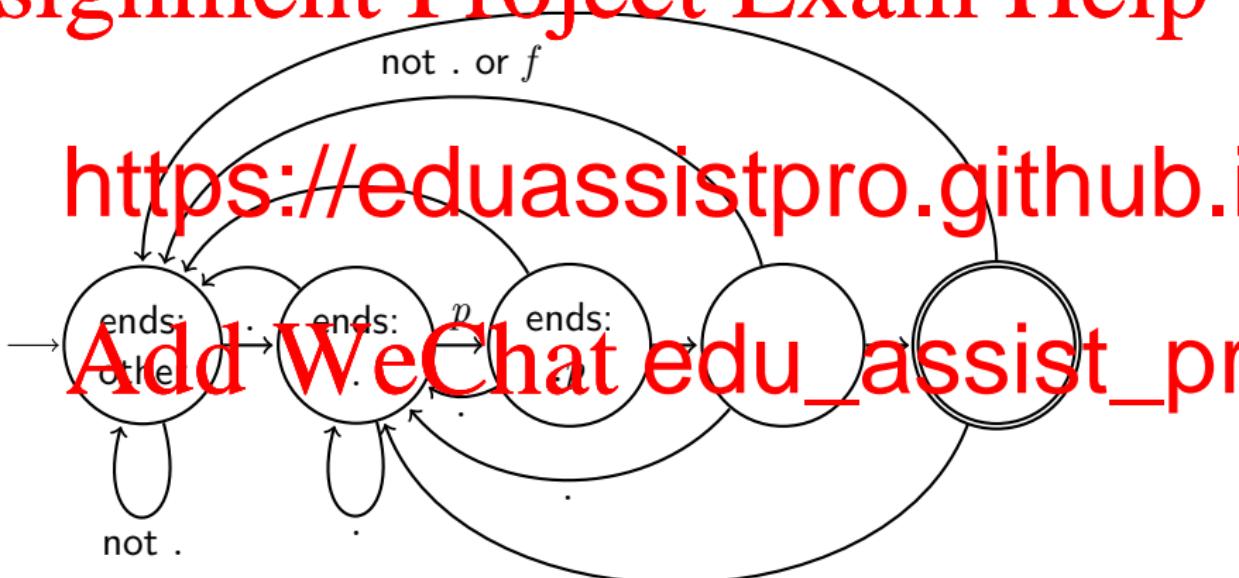
<https://eduassistpro.github.io>

Add WeChat edu_assist_pro

ALTERNATIVE NOTATION

You can be more descriptive if you like:

Assignment Project Exam Help



DETERMINISTIC FINITE AUTOMATA (DFA)

Assignment Project Exam Help

Informal definition:

- 1.
2. <https://eduassistpro.github.io>
- 3.
4. They have a behaviour given by t
 - Exact one for each alphabet symbol from Σ

5. They have accept state(s)

Add WeChat edu_assist_pro

DETERMINISTIC FINITE AUTOMATA (DFA)

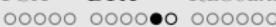
Assignment Project Exam Help

Formal definition:

A finit

1. <https://eduassistpro.github.io/>
- 2.
3. $\delta : Q \times \Sigma \rightarrow Q$ is the *transition function*
4. $q_0 \in Q$ is the *start state*, and
5. $F \subseteq Q$ is the set of *accept states*

Add WeChat edu_assist_pro



DRAW A DFA FROM THE DEFINITION

Let $M_1 = (Q, \Sigma, \delta, q_0, F)$ where:

1. $Q = \{q_0, q_1, q_2\}$

2. $\Sigma = \{0, 1\}$

3.

<https://eduassistpro.github.io/>

4. $q_0 \in Q$

5. $F = \{q_1\}$

Now we can draw M_1 :

	0	1

Add WeChat edu_assist_pro

DRAW A DFA FROM THE DEFINITION

Let $M_1 = (Q, \Sigma, \delta, q_0, F)$ where:

$$1. Q = \{q_0, q_1, q_2\}$$

$$2. \Sigma = \{0, 1\}$$

3.

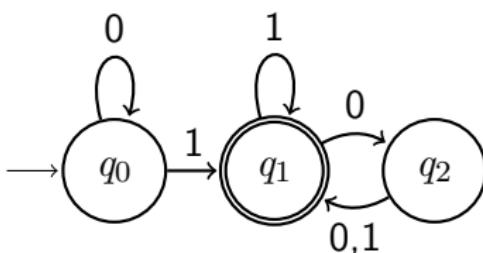
<https://eduassistpro.github.io/>

$$4. q_0 \in Q$$

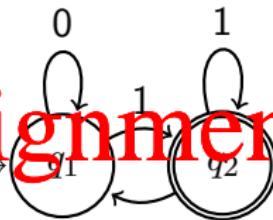
$$5. F = \{q_1\}$$

Now we can draw M_1 :

	0	1



DEFINE A DFA FROM A DIAGRAM



Assignment Project Exam Help

We can

<https://eduassistpro.github.io/>

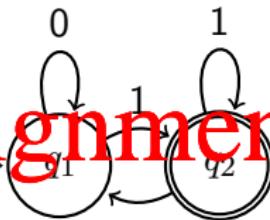
1. $\Sigma =$
2. $\Sigma =$

3. $\delta: Q \times \Sigma \rightarrow Q$ is given by

4. The start state is:
5. The set of accept states is: $F =$

Some strings where M_2 ends on an accept state are:

DEFINE A DFA FROM A DIAGRAM



Assignment Project Exam Help

We can

<https://eduassistpro.github.io/>

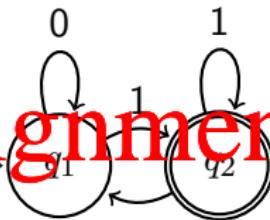
1. $\Sigma =$
2. $\Sigma =$

3. $\delta: Q \times \Sigma \rightarrow Q$ is given by

4. The start state is:
5. The set of accept states is: $F =$

Some strings where M_2 ends on an accept state are:

DEFINE A DFA FROM A DIAGRAM



Assignment Project Exam Help

We can

<https://eduassistpro.github.io/>

1. $\Sigma = \{0, 1\}$

2. $\Sigma = \{0, 1\}$

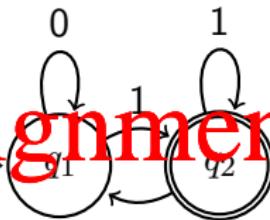
3. $\delta: Q \times \Sigma \rightarrow Q$ is given by

4. The start state is:

5. The set of accept states is: $F =$

Some strings where M_2 ends on an accept state are:

DEFINE A DFA FROM A DIAGRAM



Assignment Project Exam Help

We can

1. $\Sigma = \{0, 1\}$
2. $\Sigma = \{0, 1\}$

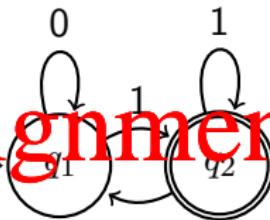
3. $\delta: Q \times \Sigma \rightarrow Q$ is given by

q		
q_2	1	2

4. The start state is:
5. The set of accept states is: $F =$

Some strings where M_2 ends on an accept state are:

DEFINE A DFA FROM A DIAGRAM



Assignment Project Exam Help

We can

1. $\Sigma = \{0, 1\}$
2. $\Sigma = \{0, 1\}$

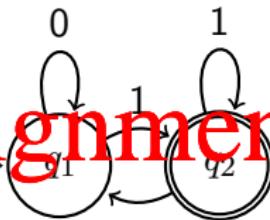
3. $\delta: Q \times \Sigma \rightarrow Q$ is given by

q		
q_2		
	1	2

4. The start state is: q_1
5. The set of accept states is: $F =$

Some strings where M_2 ends on an accept state are:

DEFINE A DFA FROM A DIAGRAM



Assignment Project Exam Help

We can

1. $\Sigma = \{0, 1\}$
2. $\Sigma = \{0, 1\}$

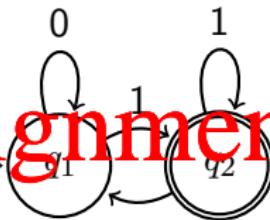
3. $\delta: Q \times \Sigma \rightarrow Q$ is given by

q		
q_2		
	1	2

4. The start state is: q_1
5. The set of accept states is: $F = \{q_2\}$

Some strings where M_2 ends on an accept state are:

DEFINE A DFA FROM A DIAGRAM



Assignment Project Exam Help

We can

- 1.
2. $\Sigma = \{0, 1\}$

3. $\delta: Q \times \Sigma \rightarrow Q$ is given by

q		
q_2		
	1	2

4. The start state is: q_1
5. The set of accept states is: $F = \{q_2\}$

Some strings where M_2 ends on an accept state are:
01, 0111, 0101

OUTLINE

Assignment Project Exam Help

- ▶ Lambda Calculus
 - ▶ Concepts (Operators, FV, Reductions)

<https://eduassistpro.github.io>

- ▶ Automata Theory
 - ▶ Background concepts
 - ▶ DFA
 - ▶ Regular languages
 - ▶ NFA

Add WeChat edu_assist_pro

FORMAL DEFINITION OF COMPUTATION

Assignment Project Exam Help

Let $M = (Q, \Sigma, \delta, q_0, F)$ be a finite automaton, and

Let

The
from

<https://eduassistpro.github.io>

1. $r_0 = q_0$
2. $\delta(r_i, w_{i+1}) = r_{i+1}$ for $i = 0, \dots,$
3. $r_n \in F$

Add WeChat edu_assist_pro

LANGUAGE OF AN AUTOMATON

Assignment Project Exam Help

► Automata of all kinds define languages

►

►

► <https://eduassistpro.github.io/>

► We often denote the language recognised by

► $L(M)$ is the set of all strings labelling paths from any state of M to any accept state in M

Add WeChat edu_assist_pro

REGULAR LANGUAGE

A language is *regular* if and only if there exists a finite automaton which recognises it.

Assignment Project Exam Help

Exercise:

Prov

regul

} is a

<https://eduassistpro.github.io>

Add WeChat edu_assist_pro

REGULAR LANGUAGE

A language is *regular* if and only if there exists a finite automaton which recognises it.

Assignment Project Exam Help

Exercise:

Prov

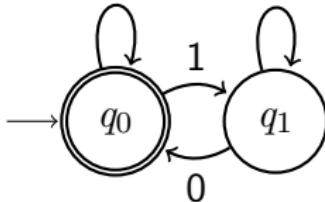
regul

<https://eduassistpro.github.io>

Solu

The following DFA recognises A , theref

Add WeChat edu_assist_pro



EXAMPLES OF REGULAR LANGUAGES

What language is accepted by M_1 ?

Assignment Project Exam Help



<https://eduassistpro.github.io>

Add WeChat edu_assist_pro

EXAMPLES OF REGULAR LANGUAGES

What language is accepted by M_1 ?

Assignment Project Exam Help



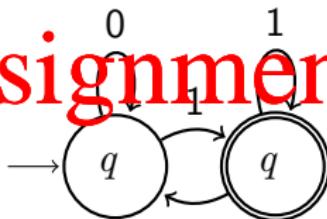
<https://eduassistpro.github.io>

Add WeChat `edu_assist_pro`

$L(M_1) = \{w \mid w \text{ contains at least } \}$
even number of 0s follow } }

EXAMPLES OF REGULAR LANGUAGES

Assignment Project Exam Help

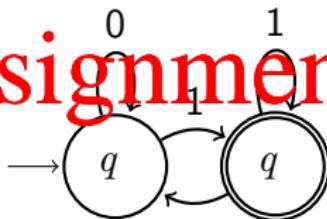


What <https://eduassistpro.github.io>

Add WeChat edu_assist_pro

What is the language recognised by the automaton obtained by inverting the accept and start states in M_2 ?

EXAMPLES OF REGULAR LANGUAGES



Assignment Project Exam Help

What is the language recognized by the automaton obtained by inverting the accept and start states in M_2 ? <https://eduassistpro.github.io>

$L(M_2) = \{w \mid w \text{ ends}$
Add WeChat edu_assist_pro

What is the language recognised by the automaton obtained by inverting the accept and start states in M_2 ?

DESIGNING FINITE AUTOMATA

“Reader as Automaton” method: The *states* encode what you need to *remember* about the string as you are reading it.

Example: Let $\Sigma = \{0,1\}$. Devise an automaton which accepts the language consisting of strings with an odd number of 1s.

We ne

- <https://eduassistpro.github.io/>

So we will need exactly two states. Then it just remains to define the transitions and define the start and accept states.

Add WeChat edu_assist_pro

DESIGNING FINITE AUTOMATA

“Reader as Automaton” method: The *states* encode what you need to *remember* about the string as you are reading it.

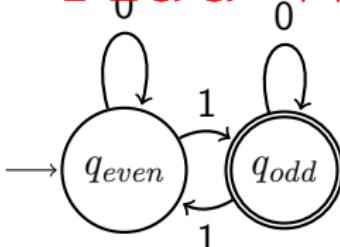
Example: Let $\Sigma = \{0,1\}$. Devise an automaton which accepts the language consisting of strings with an odd number of 1s.

We ne

► <https://eduassistpro.github.io/>

So we will need exactly two states. Then it just remains to define the transitions and define the start and accept states.

Add WeChat edu_assist_pro



EXAMPLE 2

Devise an automaton which accepts the language of strings which begin and end with 1, over $\{0, 1\}$.

We need to remember:

- Whether or not the string began with 1
-

<https://eduassistpro.github.io>

Add WeChat edu_assist_pro

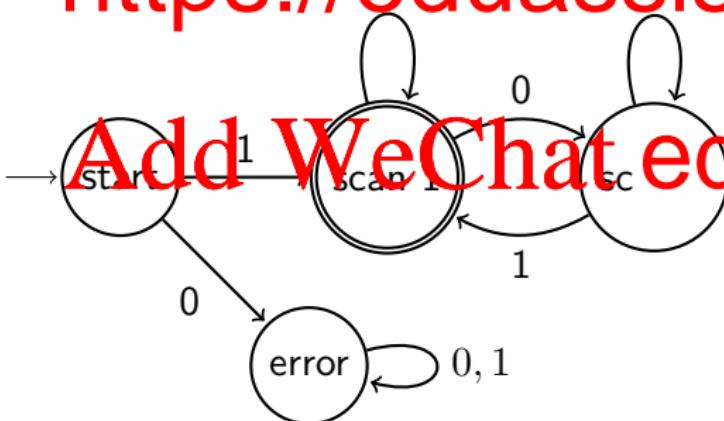
EXAMPLE 2

Devise an automaton which accepts the language of strings which begin and end with 1, over $\{0, 1\}$.

Assignment Project Exam Help

- Whether or not the string began with 1
-

<https://eduassistpro.github.io>



EXAMPLE 3

Devise an automaton which accepts the language of binary strings
which do not contain -1

Assignment Project Exam Help

What do we need to remember?

<https://eduassistpro.github.io>

Add WeChat edu_assist_pro

EXAMPLE 3

Devise an automaton which accepts the language of binary strings
which do not contain -1

Assignment Project Exam Help

What do we need to remember?



<https://eduassistpro.github.io>

Add WeChat edu_assist_pro

EXAMPLE 3

Devise an automaton which accepts the language of binary strings
which do not contain -1

Assignment Project Exam Help

What do we need to remember?



<https://eduassistpro.github.io>

Add WeChat edu_assist_pro

EXAMPLE 3

Devise an automaton which accepts the language of binary strings
which do not contain -1

Assignment Project Exam Help

What do we need to remember?

- ▶
- ▶ <https://eduassistpro.github.io>
- ▶

Add WeChat edu_assist_pro

EXAMPLE 3

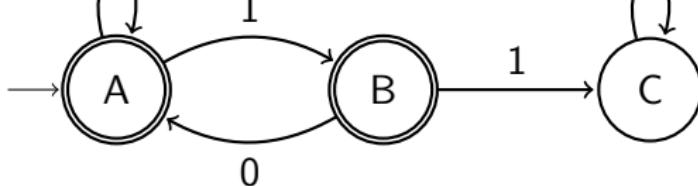
Devise an automaton which accepts the language of binary strings
which do not contain '11'

Assignment Project Exam Help

What do we need to remember?

- ▶
- ▶ <https://eduassistpro.github.io>
- ▶

Add WeChat `edu_assist_pro`



STRING IN A LANGUAGE

Let $L = \{w \mid w \in \{0, 1\}^* \text{ and } w \text{ does not contain consecutive 1s}\}$

The following DFA accepts the language L

Assignment Project Exam Help



<https://eduassistpro.github.io>

Is the string 101 in L ?

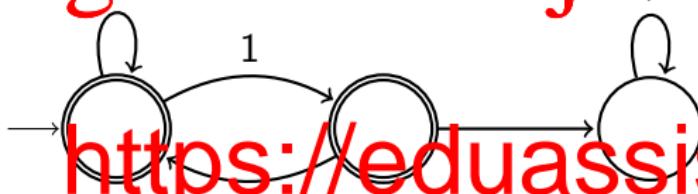
Add WeChat edu_assist_pro

STRING IN A LANGUAGE

Let $L = \{w \mid w \in \{0, 1\}^* \text{ and } w \text{ does not contain consecutive 1s}\}$

The following DFA accepts the language L

Assignment Project Exam Help



<https://eduassistpro.github.io>

Is the string 101 in L ?

- Start at A

STRING IN A LANGUAGE

Let $L = \{w \mid w \in \{0, 1\}^* \text{ and } w \text{ does not contain consecutive 1s}\}$

The following DFA accepts the language L

Assignment Project Exam Help



<https://eduassistpro.github.io>

Is the string 101 in L ?

- ▶ Start at A
- ▶ Follow transition 1 to reach B

STRING IN A LANGUAGE

Let $L = \{w \mid w \in \{0, 1\}^* \text{ and } w \text{ does not contain consecutive 1s}\}$

The following DFA accepts the language L

Assignment Project Exam Help



<https://eduassistpro.github.io>

Is the string 101 in L ?

- ▶ Start at A
- ▶ Follow transition 1 to reach B
- ▶ Follow transition 0 to reach A

STRING IN A LANGUAGE

Let $L = \{w \mid w \in \{0, 1\}^* \text{ and } w \text{ does not contain consecutive 1s}\}$

The following DFA accepts the language L

Assignment Project Exam Help



<https://eduassistpro.github.io>

Is the string 101 in L ?

- ▶ Start at A
- ▶ Follow transition 1 to reach B
- ▶ Follow transition 0 to reach A
- ▶ Follow transition 1 to reach B

STRING IN A LANGUAGE

Let $L = \{w \mid w \in \{0, 1\}^* \text{ and } w \text{ does not contain consecutive 1s}\}$

The following DFA accepts the language L

Assignment Project Exam Help



<https://eduassistpro.github.io>

Is the string 101 in L ?

- ▶ Start at A
- ▶ Follow transition 1 to reach B
- ▶ Follow transition 0 to reach A
- ▶ Follow transition 1 to reach B
- ▶ The result is an accepting state, so 101 is in the language

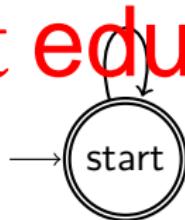
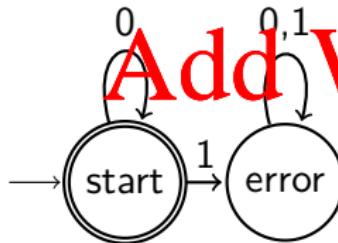
ERROR STATE(s)

An *error state* is any state from which it is impossible to reach an accept state. Sometimes we omit these from the diagram. All missing transitions point to an unseen error state.

You

Thes

<https://eduassistpro.github.io>



(error states not shown)

???

Assignment Project Exam Help

With
you a

<https://eduassistpro.github.io>

Add WeChat edu_assist_pro

OUTLINE

Assignment Project Exam Help

► Lambda Calculus

► Concepts (Operators, FV, Reductions)

<https://eduassistpro.github.io>

► Automata Theory

► Background concepts

► DFA

► Regular languages

► NFA

Add WeChat edu_assist_pro

NON DETERMINISTIC FINITE AUTOMATA (NFA)

DFA Assignment Project Exam Help

- ▶ has exactly one transition per input from each state
 - ▶

<https://eduassistpro.github.io>

NFA

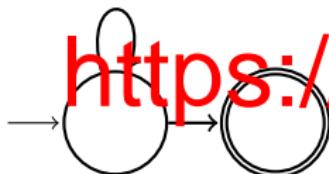
- ▶ can have any number of transitions per input f
 - ▶ so some steps of the computation might be
 - ▶ can also have ε -transitions, i.e. transi
automaton can follow without scanning any input

Add WeChat `edu_assist_pr`

NON DETERMINISTIC FINITE AUTOMATA (NFA)

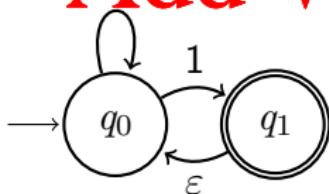
Two examples of NFA which accept the language:

$L(M) = \{wvw \in \{0, 1\}^* \text{ and } w \text{ ends in } 1\}$



<https://eduassistpro.github.io>

Add WeChat edu_assist_pro



REVIEW

Assignment Project Exam Help

► Lambda Calculus

► Revision

► Computational power



<https://eduassistpro.github.io/>

► Using a DFA to do pattern matching

► Language of a DFA

► Building a DFA

Add WeChat edu_assist_pro

► Regular languages

► Definition

► How to prove that a language is regular

► Introduction to NFA