# Teoría de autómatas y lenguajes formales

2° semestre 2022

IIC 2223

Prof. Cristian Riveros

# ¿teoría de autómatas y lenguajes formales?

## Automata theory

From Wikipedia, the free encyclopedia

Automata theory is the study of **abstract machines** and automata,
as well as the **computational problems** that can be solved using them.

**abstract machines**: "*máquinas que NO son implementadas pero que son definidas matemáticamente*"

;

**computational problems**: compiladores, extracción de información, bases de datos, verificación de software, . . .

¿algunos ejemplos de **máquinas abstractas**?

. . . daremos dos ejemplos.


replaceAll(" ", " ", a); a = a.
return a.split(" "); } $("#unique").
a = array_from_string($("#fin").
").val(), c = use_unique(array_
").val())); if (c < 2 * b - 1) {
e), this.trigger("click");
!= a[b] && " " != a[b] ||
_logged").val(); c = array_
e.length;b++) { -1 != a.
; for (b = 0;b < c.length;
$("#User_logged").val(
click(function(

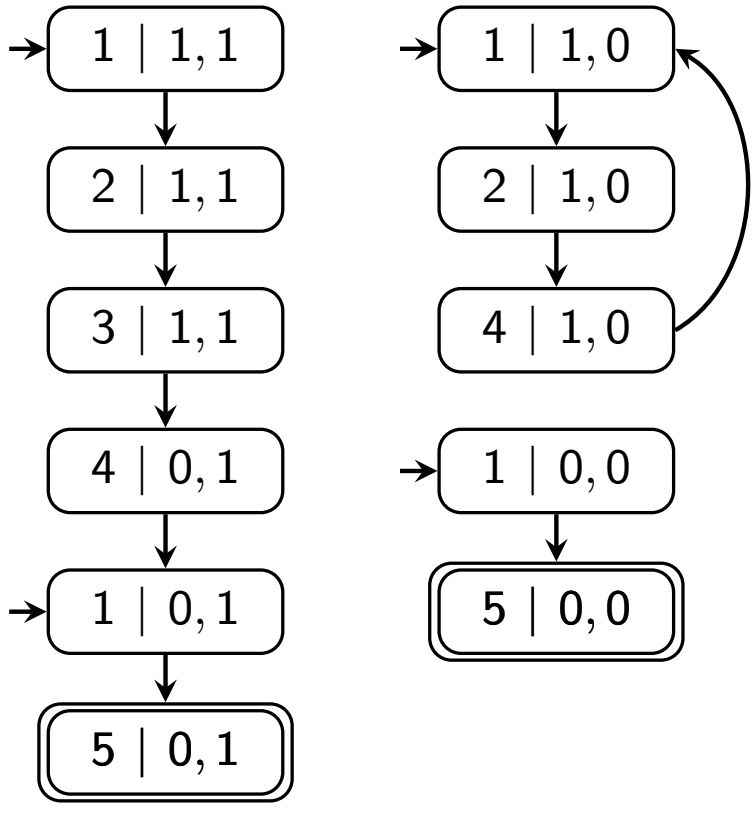
# Máquina abstracta que representa un programa

```
1 while x = 1 do
2     if y = 1 then
3         x ← 0
4     y ← (1 − x)
5 return
```

$l \mid x, y$

$l \in \{1, \ldots, 5\}$

$x, y \in \{0, 1\}$




¿siempre se detiene este código?

# Máquina abstracta que representa programas concurrentes

```
1 while true do
2     if x < 2 then
3         x ← x + 1
```

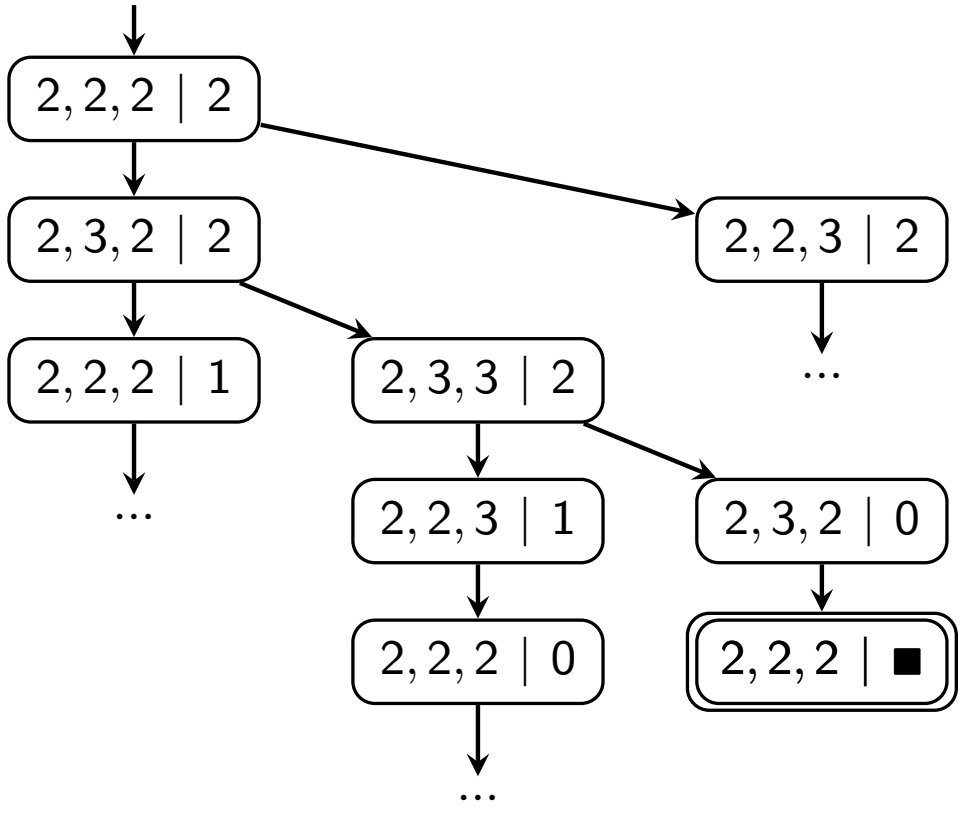
```
1 while true do
2     if x > 0 then
3         x ← x − 1
```

```
1 while true do
2     if x = 2 then
3         x ← 0
```

$\boxed{l_1, l_2, l_3 \mid x}$

$l_1, l_2, l_3 \in \{2, 3\}$

$x \in \{0, 1, 2, \blacksquare\}$




¿siempre se cumple que $0 \leq x \leq 2$?

# Model checking

**Model checking**: dado un modelo de un sistema, verificar automaticamente si el modelo cumple una especificación dada.

Dos premios Turing (novel en computación) en esta área:

- Amir Pnueli (1996).

  *"For seminal work introducing temporal logic into CS and for outstanding contributions to program and systems verification."*

- Edmund M. Clarke, E. Allen Emerson y Joseph Sifakis (2007).

  *"For their roles in developing model checking into a highly effective verification technology, widely adopted in the hardware and software industries."*

**Teoría de autómatas** constituye una parte fundamental de model checking y verificación formal de software.

# Secuencias de ADN y verificación de patrones

```
AGGATGACCCGAAATGCCCC
TCCAGCCAAAGGAGTCCGTT
ATGAGGGGATGGCAGCATGT
TGGTGGACAATTTCGAGGGA
GAGAACCGCTTAGCAGCGCT
TTTGACCGAAATAACCCATA
GCCTCGCAATAATAGTACGC
CGCAATGAAGCTTGTTTGAG
TCTTAACAGTATCTGGA...
```

Buscamos la subsecuencia "**ACAA**".

# Secuencias de ADN y verificación de patrones

```
AGGATGACCCGAAATGCCCC
TCCAGCCAAAGGAGTCCGTT
ATGAGGGGATGGCAGCATGT
TGGTGGACAATTTCGAGGA
GAGAACCGCTTAGCAGCGCT
TTTGACCGAAATAACCCATA
GCCTCGCAATAATAGTACGC
CGCAATGAAGCTTGTTTGAG
TCTTAACAGTATCTGGA...
```

Buscamos la subsecuencia "**ACAA**".

```
1 for i ← 1 to d.lenght do
2     if d[i] = 'A' then
3         if d[i + 1] = 'C' then
4             if d[i + 2] = 'A' then
5                 if d[i + 3] = 'A' then
6                     Output i
```

Para un patrón $p$

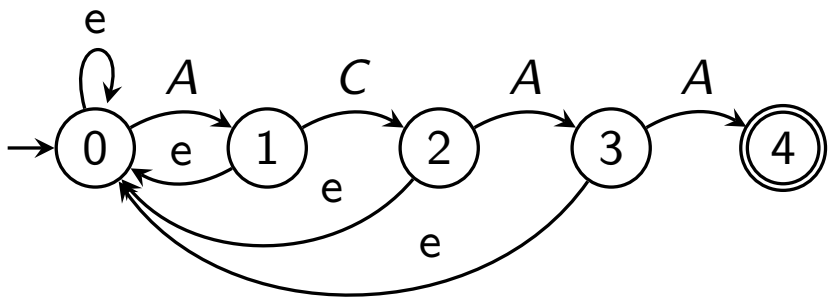
**tiempo**: $O(|p| \cdot |d|)$

¿podemos hacer una algoritmo más eficiente?

# Secuencias de ADN y verificación de patrones

```
AGGATGACCCGAAATGCCCC
TCCAGCCAAAGGAGTCCGTT
ATGAGGGGATGGCAGCATGT
TGGTGGACAATTTCGAGGA
GAGAACCGCTTAGCAGCGCT
TTTGACCGAAATAACCCATA
GCCTCGCAATAATAGTACGC
CGCAATGAAGCTTGTTTGAG
TCTTAACAGTATCTGGA...
```

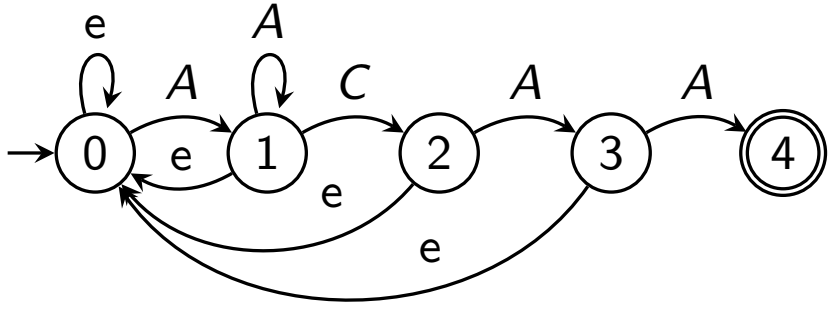
```
1  s = 0
2  for i ← 1 to d.lenght do
3      if s = 0 ∧ d[i] = 'A' then
4          s = 1
5      else if s = 1 ∧ d[i] = 'C' then
6          s = 2
7      else if s = 2 ∧ d[i] = 'A' then
8          s = 3
9      else if s = 3 ∧ d[i] = 'A' then
10         Output i − 3
11     else  s = 0
```



# Secuencias de ADN y verificación de patrones

```
AGGATGACCCGAAATGCCCC
TCCAGCCAAAGGAGTCCGTT
ATGAGGGGATGGCAGCATGT
TGGTGGACAATTTCGAGGA
GAGAACCGCTTAGCAGCGCT
TTTGACCGAAATAACCCATA
GCCTCGCAATAATAGTACGC
CGCAATGAAGCTTGTTTGAG
TCTTAACAGTATCTGGA...
```

```
1  s = 0
2  for i ← 1 to d.lenght do
3      if (s = 0 ∨ s = 1) ∧ d[i] = 'A' then
4          s = 1
5      else if s = 1 ∧ d[i] = 'C' then
6          s = 2
7      else if s = 2 ∧ d[i] = 'A' then
8          s = 3
9      else if s = 3 ∧ d[i] = 'A' then
10         Output i − 3
11     else  s = 0
```



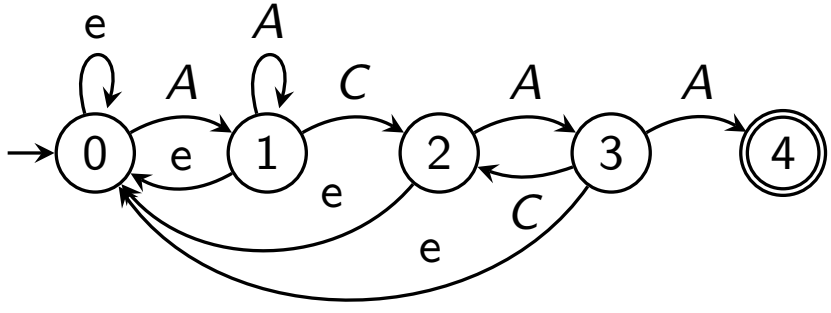
# Secuencias de ADN y verificación de patrones

```
AGGATGACCCGAAATGCCCC
TCCAGCCAAAGGAGTCCGTT
ATGAGGGGATGGCAGCATGT
TGGTGGACAATTTCGAGGA
GAGAACCGCTTAGCAGCGCT
TTTGACCGAAATAACCCATA
GCCTCGCAATAATAGTACGC
CGCAATGAAGCTTGTTTGAG
TCTTAACAGTATCTGGA...
```

Para una subsecuencia $s$

**tiempo**: $O(|s| + |d|)$

```
1  s = 0
2  for i ← 1 to d.lenght do
3      if (s = 0 ∨ s = 1) ∧ d[i] = 'A' then
4          s = 1
5      else if (s = 1 ∨ s = 3) ∧ d[i] = 'C' then
6          s = 2
7      else if s = 2 ∧ d[i] = 'A' then
8          s = 3
9      else if s = 3 ∧ d[i] = 'A' then
10         Output i − 3
11     else  s = 0
```



# Pattern matching y teoría de autómatas

**Pattern matching**: encontrar la presencia de un patrón en una secuencia de información.

Varias aplicaciones en:

- Sequenciación de ADN.
- Busquedas en la web.
- Lingüística.
- ...

**Teoría de autómatas** constituye el modelo de abstracción para el desarrollo de algoritmos más eficientes.

# CONCLUSIONES

# Algunas conclusiones de los dos ejemplos

1. Autómatas constituyen una noción
   fundamental y útil en ciencia de la computación.
2. Aplicaciones de teoría de autómatas en diversas áreas de computación.
3. La modelación con teoría de autómatas nos permite abstraernos del problema para así encontrar soluciones más eficientes.

> *"I suggest that automata theory is the* linear algebra *of computer science. . . as a basic, fundamental subject, known and used by everyone, which has formed part of the intelectual landscape for so long that it is no longer noticed."*
>
> Jacques Sakarovitch.

# Outline

Motivación

Programa

Consejos

# Programa del curso

**Cristian Riveros**

- Profesor
- cristian.riveros@uc.cl
- Oficina 3-S, DCC

**Dante Pinto**

- Ayudante jefe
- drpinto1@uc.cl

# Programa del curso

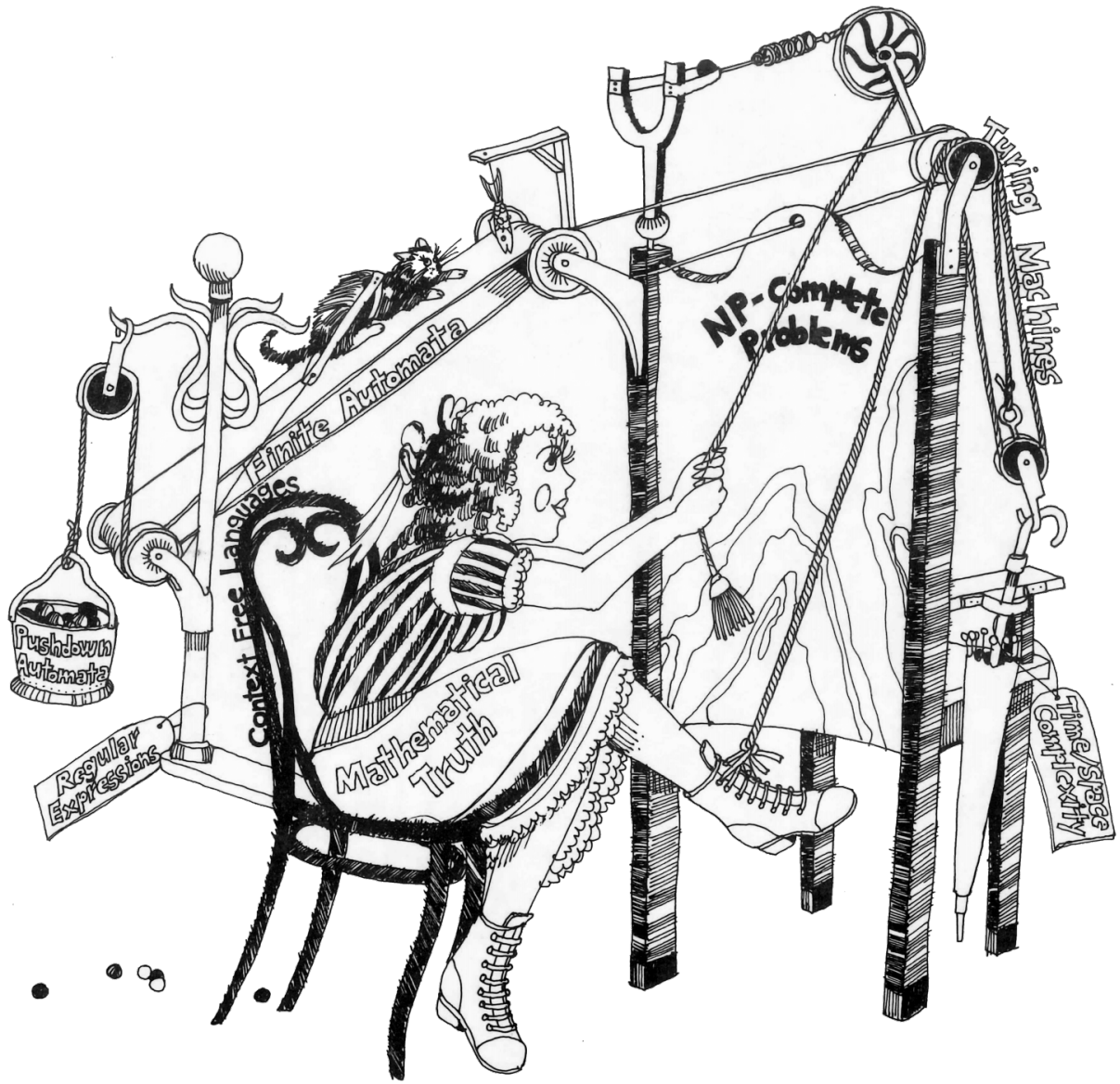
| | |
|---|---|
| Clases: | Lunes y Miércoles módulo 2. |
| | Sala B25 (por confirmar). |
| Ayudantías: | Viernes módulo 2. |
| | Sala B17 (por confirmar). |
| Atención: | A toda hora, preferentemente después de clases. |

Tanto las clases como las ayudantías serán presenciales, sin asistencia

# Objetivos del curso

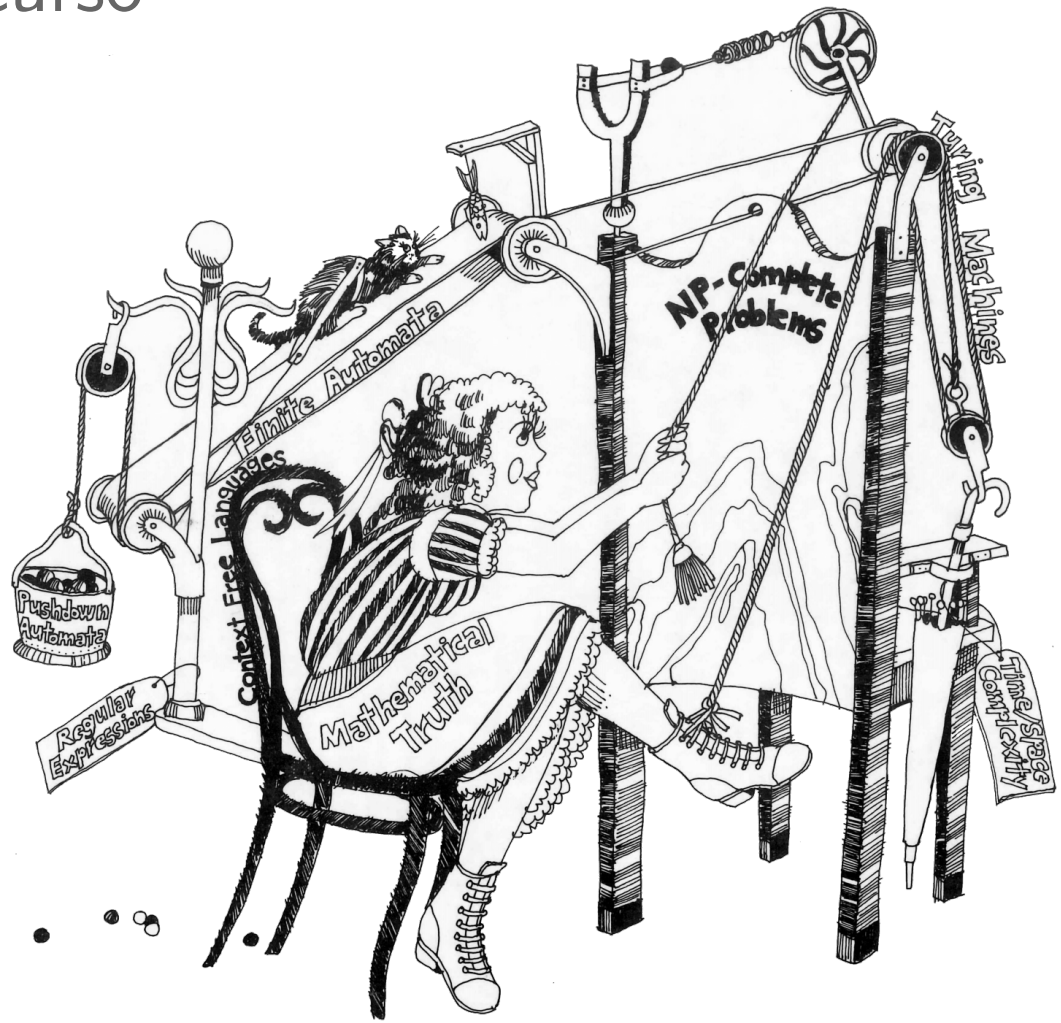
1. Estudiar las nociones fundamentales de la ciencia de la computación.
2. Entender los **modelos básicos** de computabilidad y complejidad.
3. Entender la importancia para aplicaciones prácticas en ingeniería.

# Temario del curso



# ADVERTENCIA!

# Temario del curso




- **NO** vemos **máquinas de Turing**
  (ver en curso Lógica para Ciencia de la Computación).
- Veremos más **algoritmos y aplicaciones** en teoría de autómatas.

# Evaluaciones

1. Tareas.
2. Interrogaciones.
3. Examen.

# Tareas

- Seis tareas durante el semestre (se borrá la peor tarea).
- Enunciado se publicará los días viernes.
- La entrega será el jueves siguiente hasta las 23:59.
- La evaluación de cada pregunta en las tareas será de:
  - 0 (respuesta incorrecta)
  - 2 (con errores importantes)
  - 3 (con errores menores)
  - 4 (respuesta correcta).

# Tareas: formato de entrega

- El método de entrega será online.
- La tarea debe ser escrita y entregada en LaTeX.

**NO** se aceptarán tareas **escritas a mano**
ni en otro sistema de composición de texto.

# Tareas: problema excepcional

- Cada estudiante cuenta con un cupón `#problemaexcepcional`.
- Se puede utilizar **solo una vez durante el semestre**.

  *Permite extender el plazo de entrega de una tarea sin necesidad de una justificación debido a motivos excepcionales y personales.*

- La extensión mueve el plazo de entrega hasta las 23:59 horas del día lunes de la semana siguiente a la entrega original.

**NO** se aceptarán tareas **fuera de plazo** y **NO** se harán **excepciones**.

# Interrogaciones y examen

- Dos interrogaciones y un examen.
- Interrogaciones presenciales a las 18:30 y duración de 2 a 3 horas.
- Examen presencial a las 9 horas.

# Interrogaciones y examen: fechas

| | **Publicación enunciado** | **Entrega** |
|---|---|---|
| Tarea 1 | Viernes 26 de Agosto | Jueves 1 de Septiembre |
| Tarea 2 | Viernes 9 de Septiembre | Jueves 15 de Septiembre |
| **Interrogación 1** | Viernes 23 de Septiembre | |
| Tarea 3 | Viernes 30 de Septiembre | Jueves 6 de Octubre |
| Tarea 4 | Viernes 14 de Octubre | Jueves 20 de Octubre |
| **Interrogación 2** | Miércoles 26 de Octubre | |
| Tarea 5 | Viernes 4 de Noviembre | Jueves 10 de Noviembre |
| Tarea 6 | Viernes 18 de Noviembre | Jueves 24 de Noviembre |
| **Examen** | Martes 6 de Diciembre | |

"El profesor no se hará responsable por tope de horarios con interrogaciones o exámenes de cursos que se regulen por la programación académica de la Escuela de Ingeniería."

# Interrogaciones y examen: problema de fuerza mayor

Problema de **fuerza mayor** = cualquier enfermedad o problema que impida rendir una evaluación.

Para solo una **interrogación**:

- La nota del examen reemplazará la nota de esta interrogación.
- La inasistencia **NO** necesita ser justificada.
  - Se reemplazará **automáticamente** la peor nota por el examen.

Para el **examen**:

- Justificativo según las reglas de la Escuela de Ingeniería en Pregrado.
- Nota P y el examen se rinde a comienzos del próximo semestre.

**NO** se harán **excepciones**.

# Evaluación

Si $N_1, \ldots, N_k$ es una lista de $k$ notas:

$$\mathrm{AVG}_n(N_1, \ldots, N_k) \quad := \quad \text{promedio aritmético de los } n \text{ valores más altos de } N_1, \ldots, N_k.$$

Nota **Tareas** (**PT**):

$$\mathbf{PT} \;=\; \mathrm{AVG}_5(\, T_1\,,\; T_2\,,\; T_3\,,\; T_4\,,\; T_5\,,\; T_6\,).$$

Nota **Interrogaciones** y **Examen** (**PE**):

$$\mathbf{PE} \;=\; \mathrm{AVG}_3(\, I_1\,,\; I_2\,,\; E\,,\; E\,)$$

# Evaluación

Nota **Final** (**NF**):

$$\mathbf{NF} \;=\; 0,3 \cdot \mathbf{PT} \;+\; 0,7 \cdot \mathbf{PE}$$

El curso se aprueba si,y solo si, **todas** las siguiente condiciones se cumplen:

- $\mathbf{PT} \;\geq\; 2,95$
- $\mathbf{PE} \;\geq\; 3,95$
- $\mathbf{NF} \;\geq\; 3,95$

En caso de **no aprobar**, la nota final del curso será $\min\{\mathbf{NF},\ 3{,}9\}$.

# Comunicación digital

- Anuncios, clases, ejercicios, etc . . .

  **Canvas / Teoría de Autómatas y Lenguajes Formales**

- Google Calendar:

  **https://bit.ly/3p3LKYU**

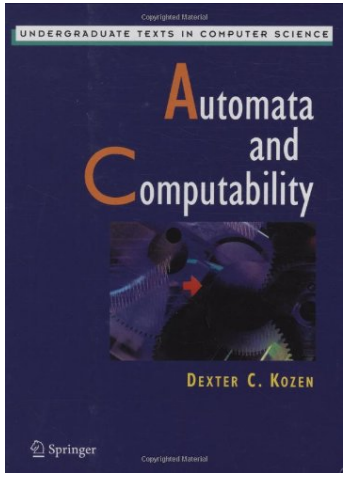
- Preguntas sobre materia:
  - Foro de Canvas.
  - Personales:

    **iic2223@ing.puc.cl**

- Preguntas por problemas personales relacionados al curso:

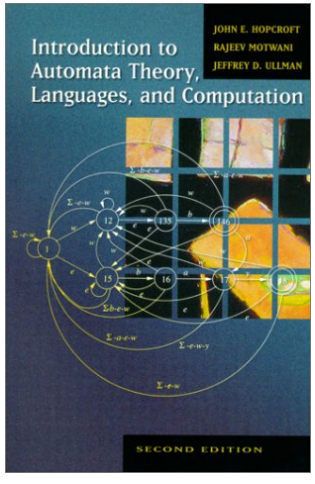
  **cristian.riveros@uc.cl**

# Bibliografía




**Automata and Computability**

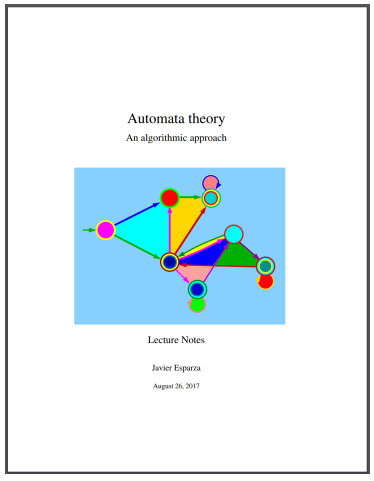
- Dexter Kozen.
- Springer, 1997.




**Intr. to automata theory languages and computation**

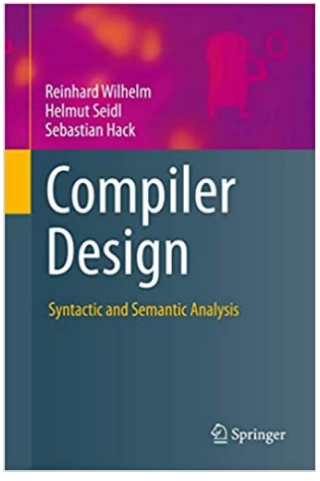
- Hopcroft, Motwani, y Ullman.
- Addison-Wesley o Prentice Hall.
- Múltiples ediciones: 1979, 2000, 2006.

# Bibliografía




**Automata Theory: An Algorithmic Approach.**

- Javier Esparza.
- Disponibles en internet.




**Compiler design: syntactic and semantic analysis**

- Wilhelm, Seidl y Hack.
- Springer, 2013.

# Outline

Motivación

Programa

Consejos

# Sobre las tareas, interrogaciones y examen

Recomendaciones de estudio :

1. Asistir a clases.
2. Leer la materia de un libro.
3. Pensar . . .
4. Hacer y escribir las demostraciones (individualmente).
5. Pensar . . .
6. Hacer varios y diversos ejercicios.

En este curso **NO** se pueden mecanizar los ejercicios.

# Sobre correcciones y recorrecciones

El proceso de **corrección** y **recorrección** de evaluaciones será el siguiente:

1. Entrega de notas y feedback (online): 2 semanas.
2. Recorrección "presencial".
3. Recorrección "escrita".
4. En caso de no quedar satisfecho con recorrección,
   solicitar la recorrección con el profesor (enviar un correo).

En caso de tener cualquier duda sobre corrección o feedback,
enviar un correo a **iic2223@ing.puc.cl**.

# Sobre copia

- Tanto las tareas, interrogaciones y examen son individuales.
- Material de libros o Internet debe estar debidamente **referenciados**.
- En caso de copia se aplicará:

POLÍTICA DE INTEGRIDAD ACADÉMICA
DEL DEPARTAMENTO DE CIENCIA DE LA COMPUTACIÓN

# ¿PREGUNTAS?