Programming Languages Foundations

Computer Science III

Author: Eng. Carlos Andrés Sierra, M.Sc.
cavirguezs@udistrital.edu.co

Lecturer
Computer Engineer
School of Engineering
Universidad Distrital Francisco José de Caldas

2024-III

Outline

1. Programming Languages
2. Finite State Machines
3. Generative Grammars

Outline

1 Programming Languages

2 Finite State Machines

3 Generative Grammars

UNIVERSIDAD DISTRITAL
FRANCISCO JOSÉ DE CALDAS

Babbage Machine

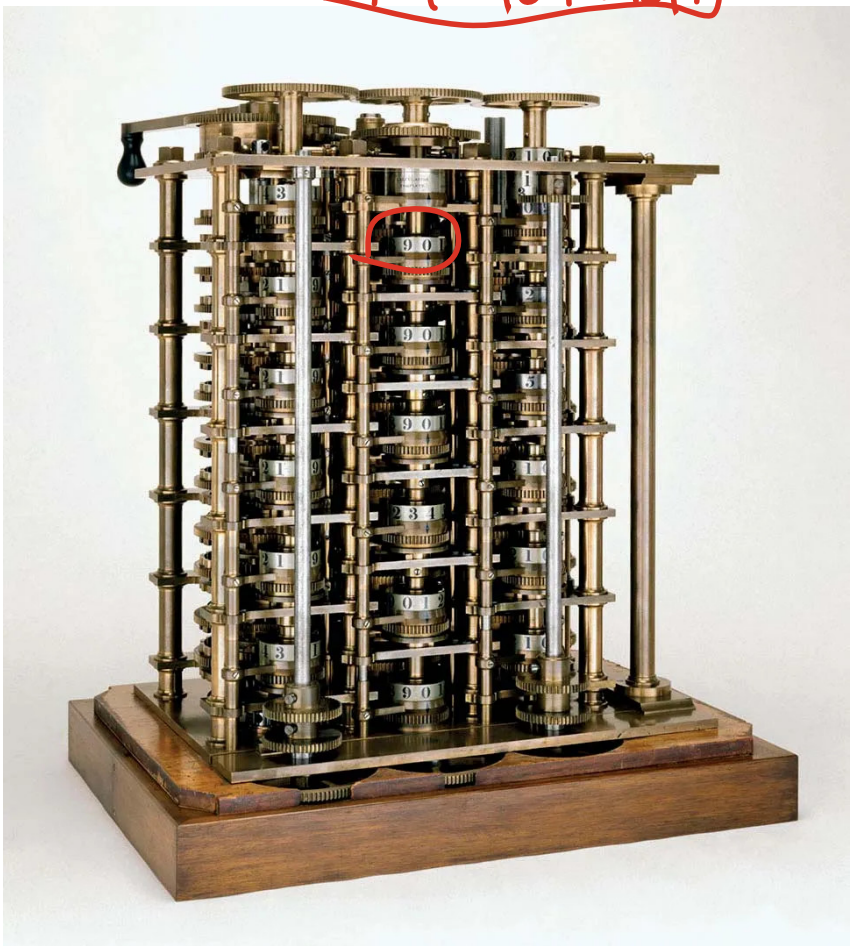


Figure: Analytical Machine

- **Charles Babbage** (1791 — 1871) was an English mathematician, philosopher, inventor, and mechanical engineer.
- He originated the **concept** of a digital programmable computer.
- Considered the "father of the computer". He creates the Analytical Engine.
- The **Analytical Engine** was a general-purpose mechanical computer.

Ada Lovelace

- **Ada Lovelace** (1815 — 1852) was an English mathematician and writer.
- She is known for her work on Charles Babbage's early mechanical general-purpose computer, the Analytical Engine.
- She was the **first** to recognize that the machine had applications beyond pure calculation, and to have published the first algorithm intended to be carried out by such a machine.

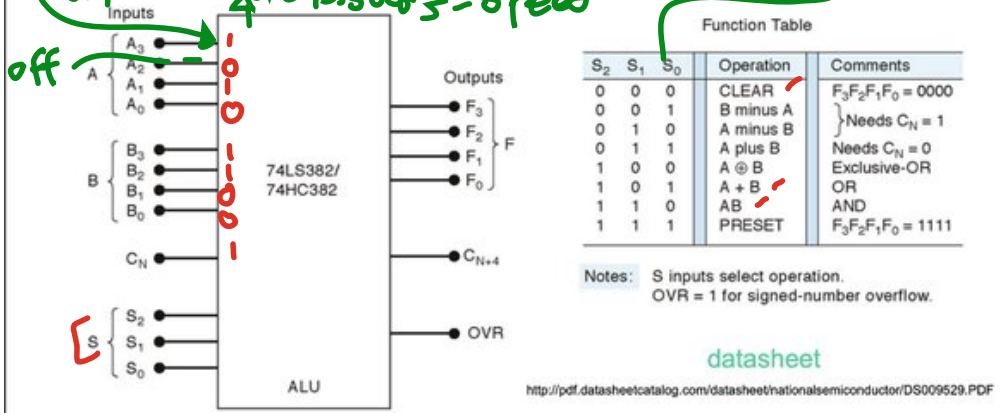
Figure: Ada Lovelace

Physical Binary Language

- **Binary** is a base-2 number system. It uses only two symbols: typically 0 (zero) and 1 (one).
- The **bit** is the basic unit of information in computing and digital communications.

Arithmetic Logic Unit

- Examine the functionality of this 74LS382 ALU chip



Function Table

S_2	S_1	S_0	Operation	Comments
0	0	0	CLEAR	$F_3F_2F_1F_0 = 0000$
0	0	1	B minus A	Needs $C_N = 1$
0	1	0	A minus B	Needs $C_N = 1$
0	1	1	A plus B	Needs $C_N = 0$
1	0	0	A ⊕ B	Exclusive-OR
1	0	1	A + B	OR
1	1	0	AB	AND
1	1	1	PRESET	$F_3F_2F_1F_0 = 1111$

Notes: S inputs select operation.
OVR = 1 for signed-number overflow.

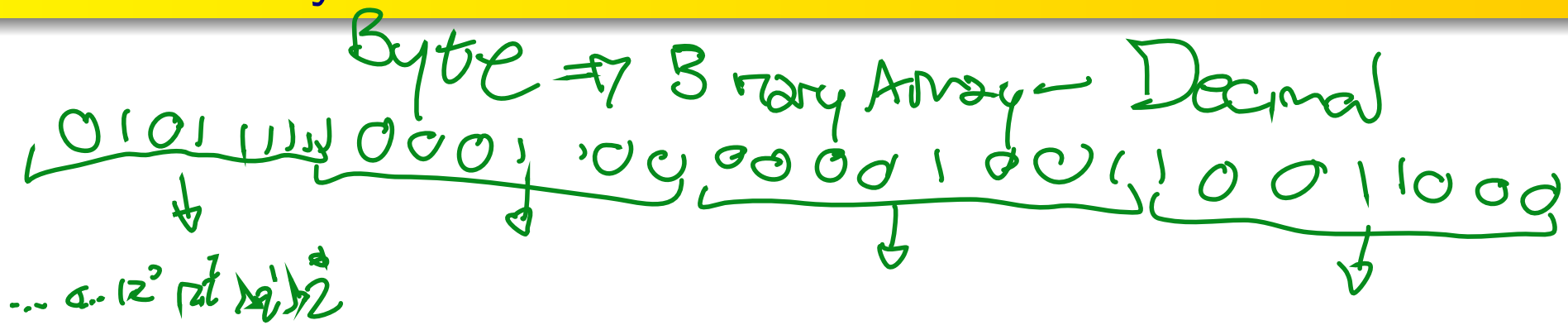
datasheet

http://pdf.datasheetcatalog.com/datasheet/nationalsemiconductor/DS009529.PDF

A = 4-bit input number
B = 4-bit input number
C_N = carry into LSB position
S = 3-bit operation select inputs

F = 4-bit output number
C_{N+4} = carry out of MSB position
OVR = overflow indicator

Bits and Bytes



1+2+4+8+16+32+64+128 = 255

0+0+0+0+0+0+0+0 = 0

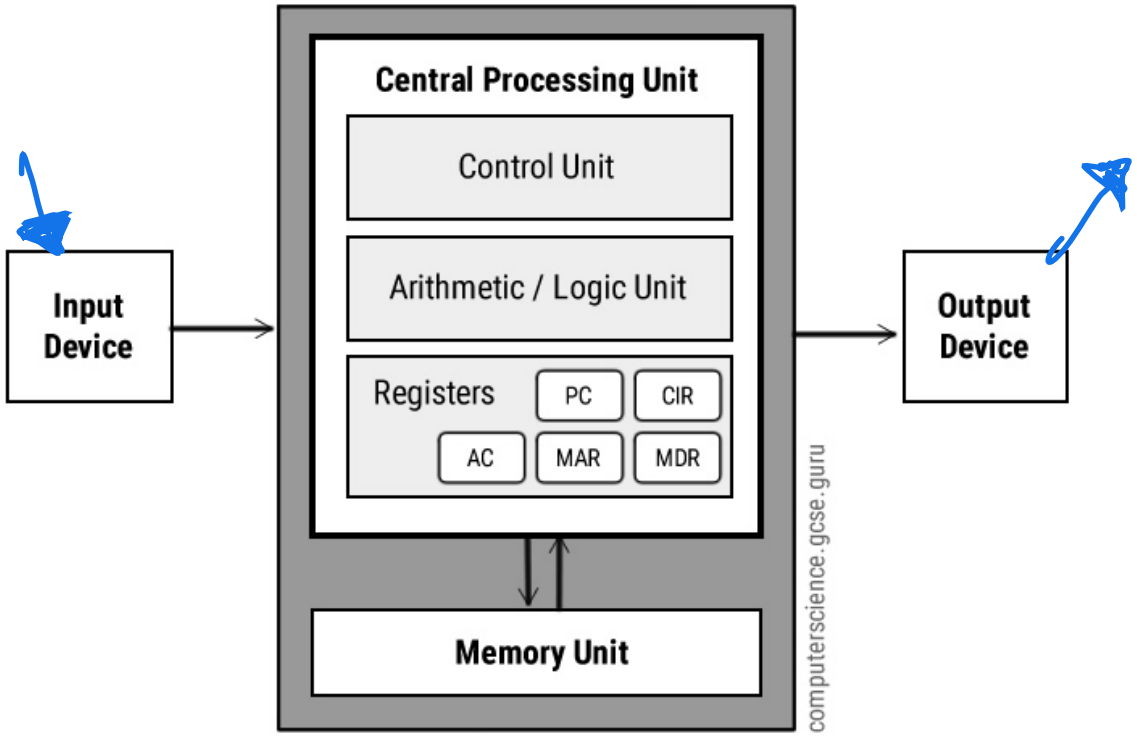
256 possibilities

ASCII => Unicode => Hexadecimal

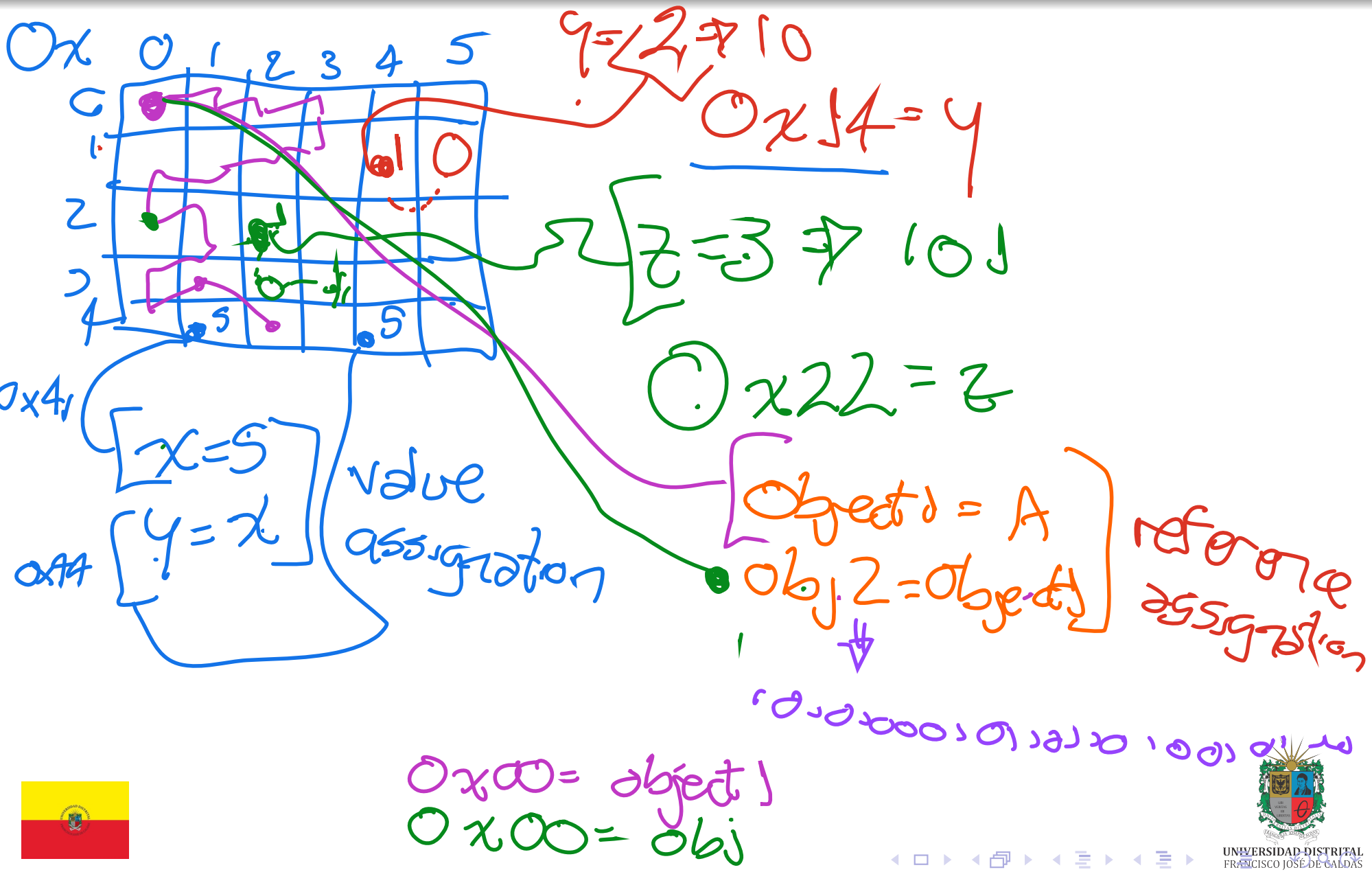
byte	dec	Symbol

Von Neumann Architecture

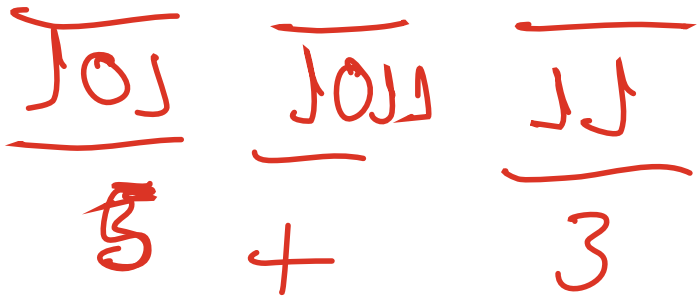
- The **Von Neumann Architecture** is a computer architecture based on the stored-program computer concept.
- The **design** is based on the concept of an instruction set.
- The **program** and **data** are both stored in the same memory unit.

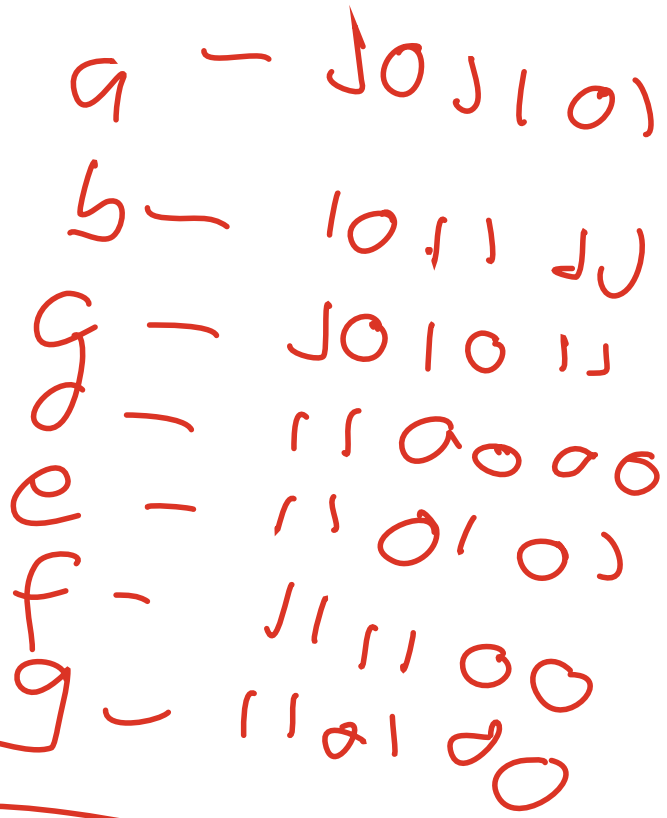


Memory and Bit Storage



Machine Programming Language



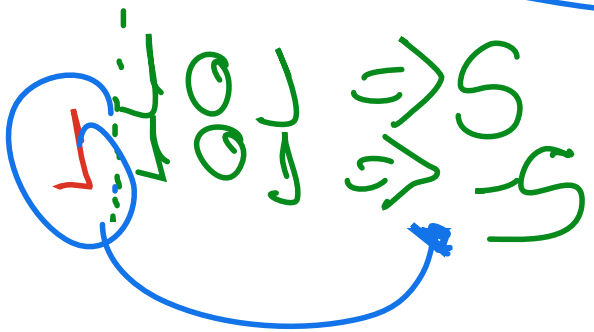


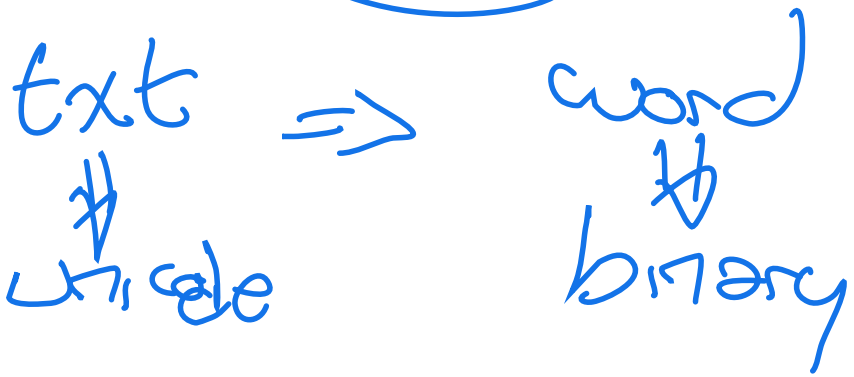
Physical Machine
≠
Digital Electronic

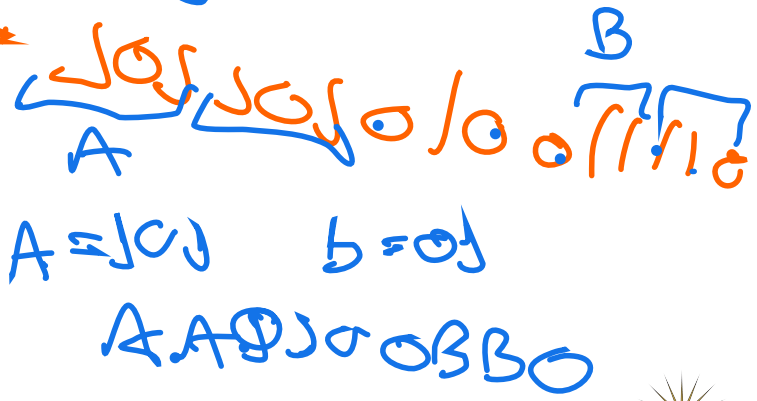
50 Mbps → 50.000.000 bytes/s
400.000.000 bits/s

Bit Operations

- **Bitwise operations** are operations that directly manipulate bits.
- They are used in low-level programming for performing calculations, file processing, and data compression.





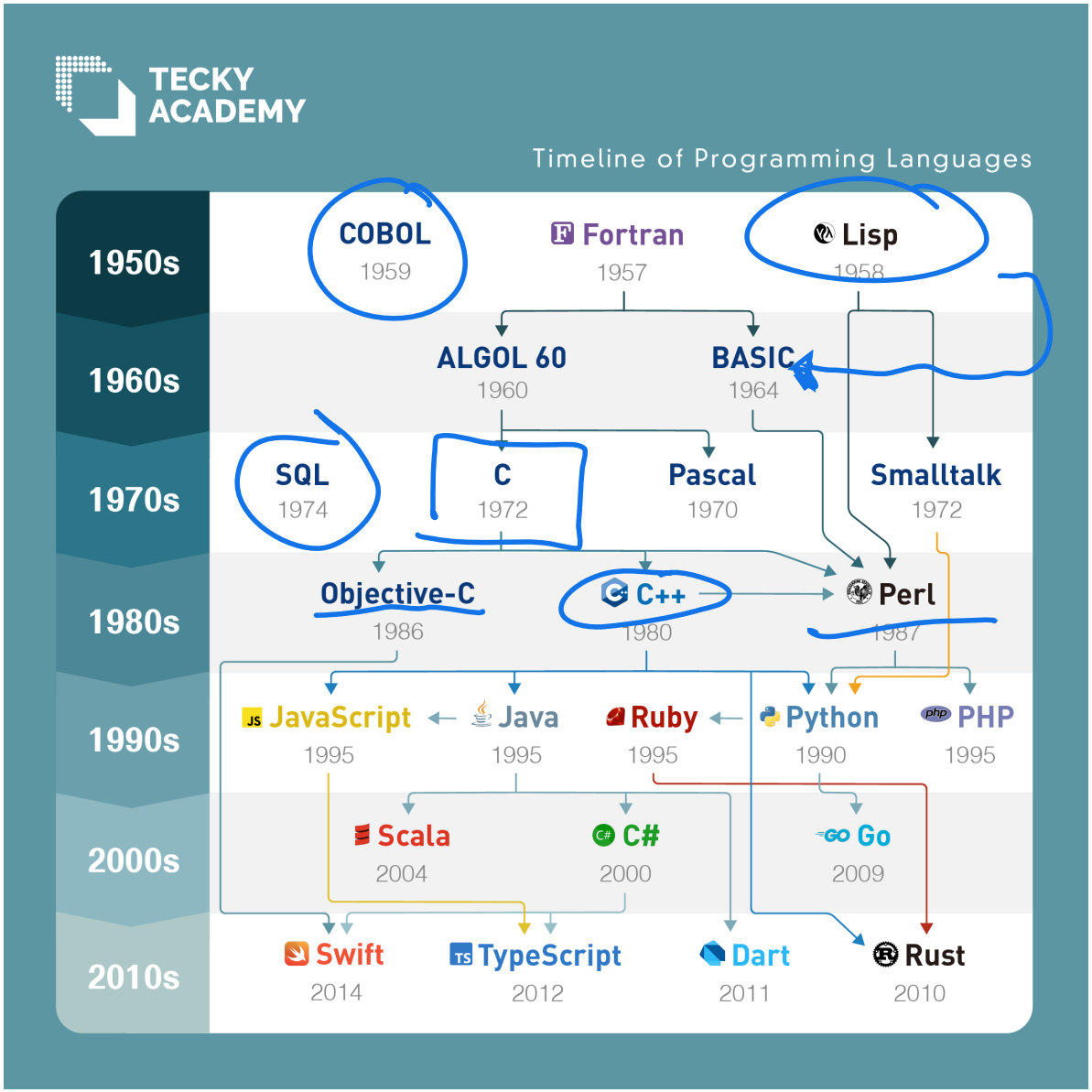


Assembly Programming Language

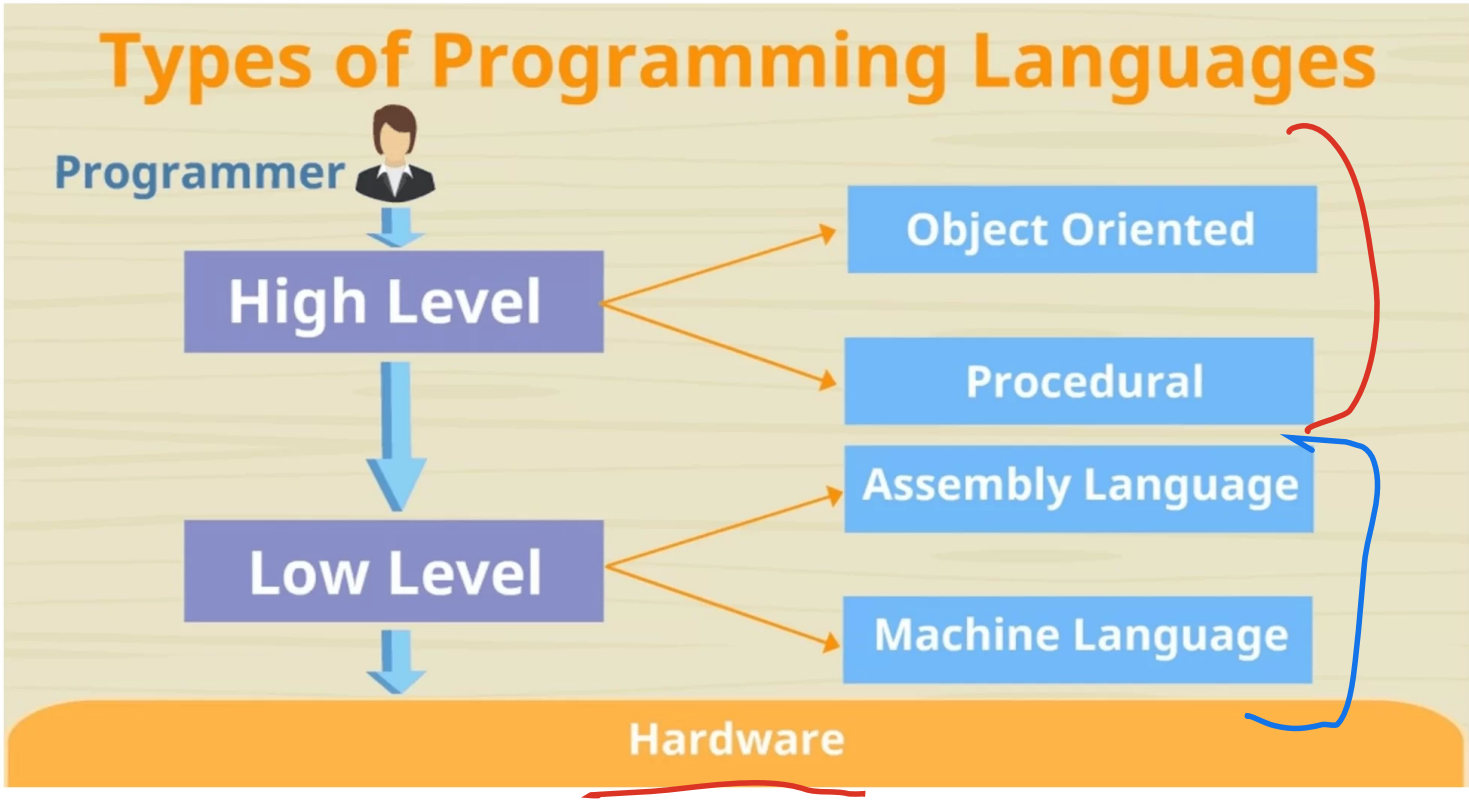
```
1  ; Example of a basic conditional structure in x86
   assembly
2
3  cmp eax, 10     ; Compare the value in eax with 10
4  je equal_label  ; Jump to equal_label if eax is equal to
   10
5
6  ; Code for not equal case
7  jmp end_label   ; Jump to end_label to avoid executing the
    equal case code
8
9  equal_label:
10 ; Code for equal case
11
12 end_label:
13 ; Continue execution
14
```

Listing 1: Basic Conditional Structure in Assembly

History of Programming Languges



High-Level Programming Languages I



High-Level Programming Languages II

Purpose of High-Level Languages

Ease of Use

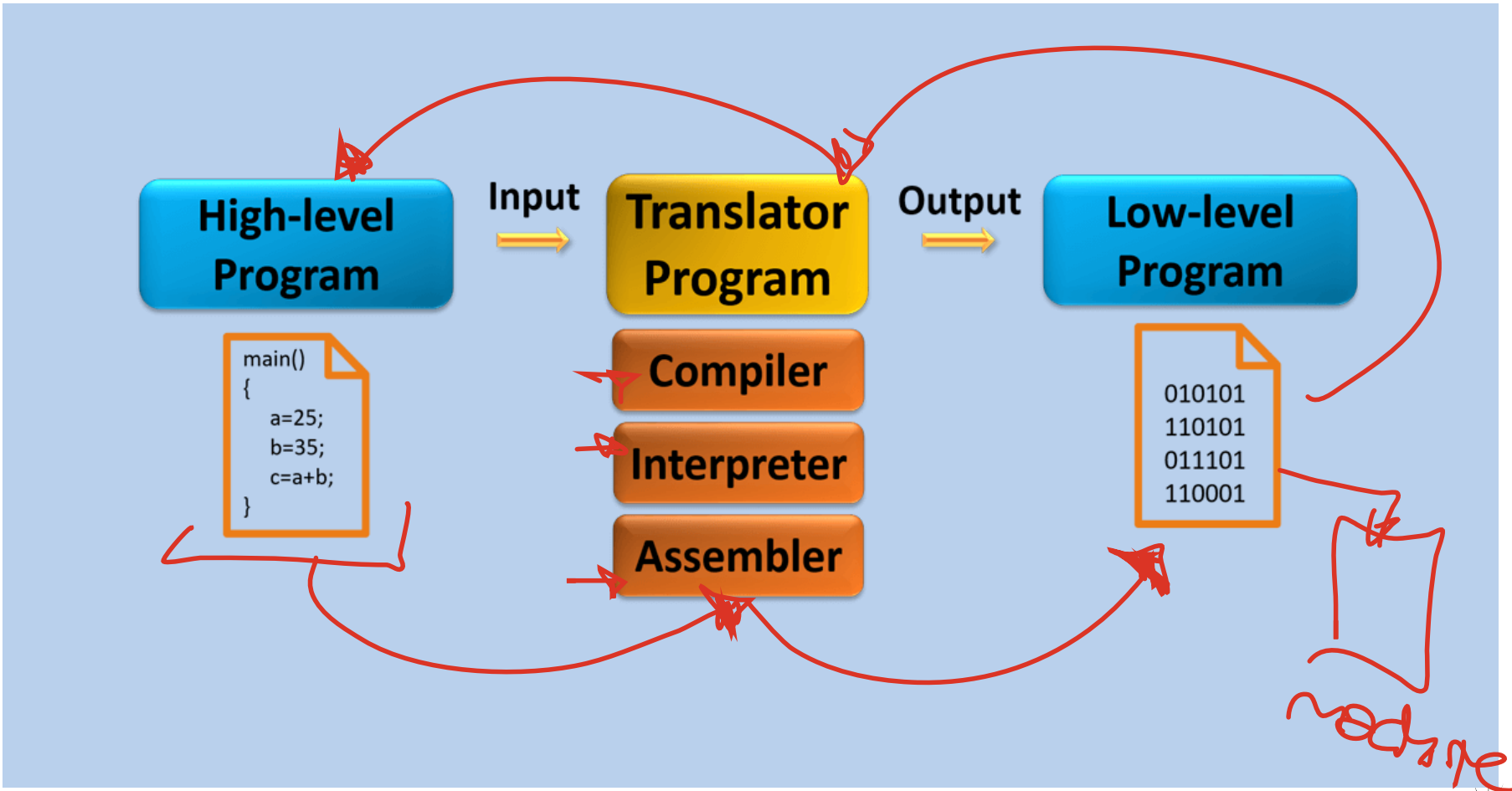
- Simplifying programming
- Minimizing learning curve
- Enhancing productivity
- Automated memory management
- Clear syntax
- Readability and maintainability

Portability Across Systems

- Cross-platform compatibility
- Utilization of compilers and interpreters
- Seamless execution on various platforms
- Reduction of platform-specific modifications
- Enhanced flexibility across environments

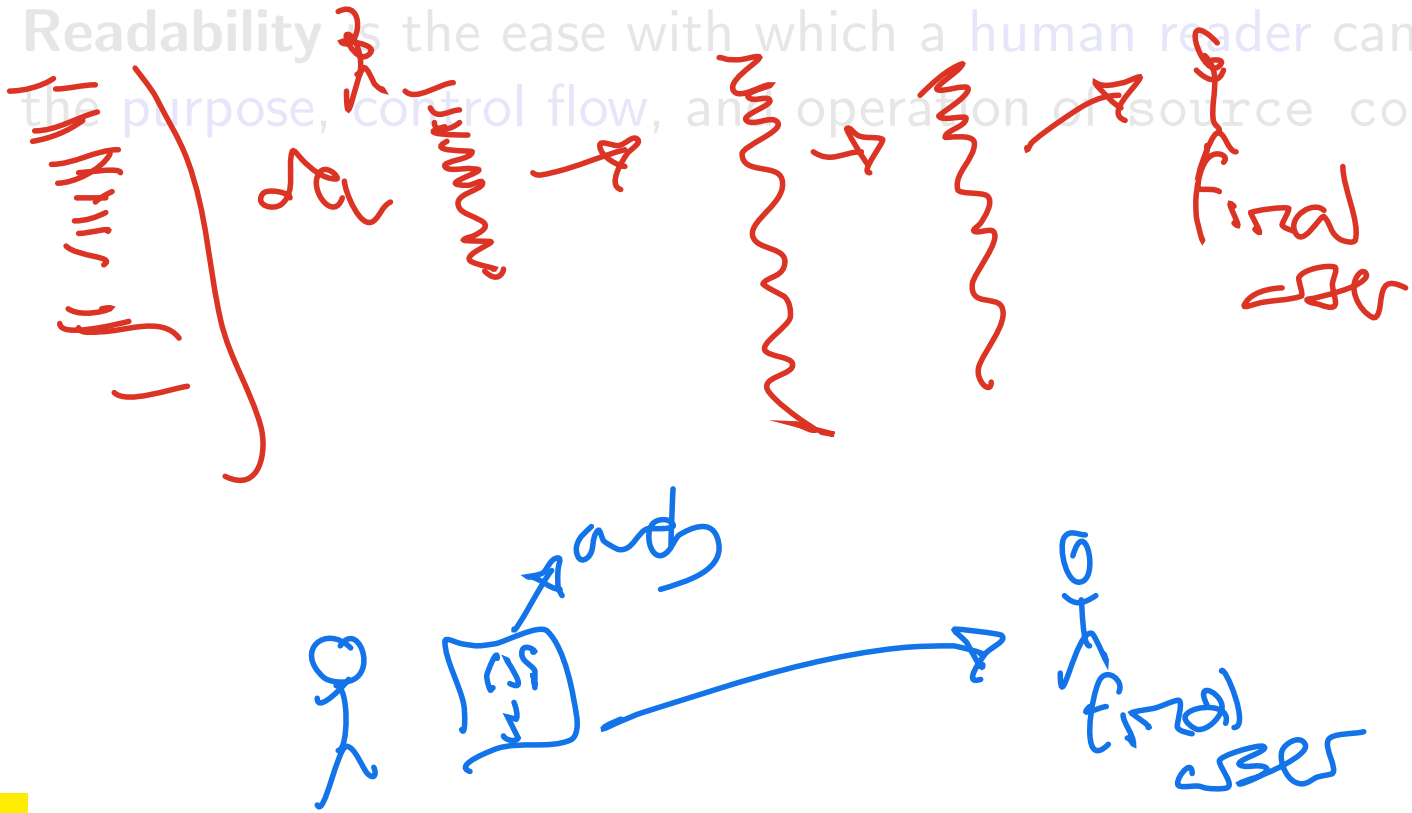
Techopedia

Translation Process



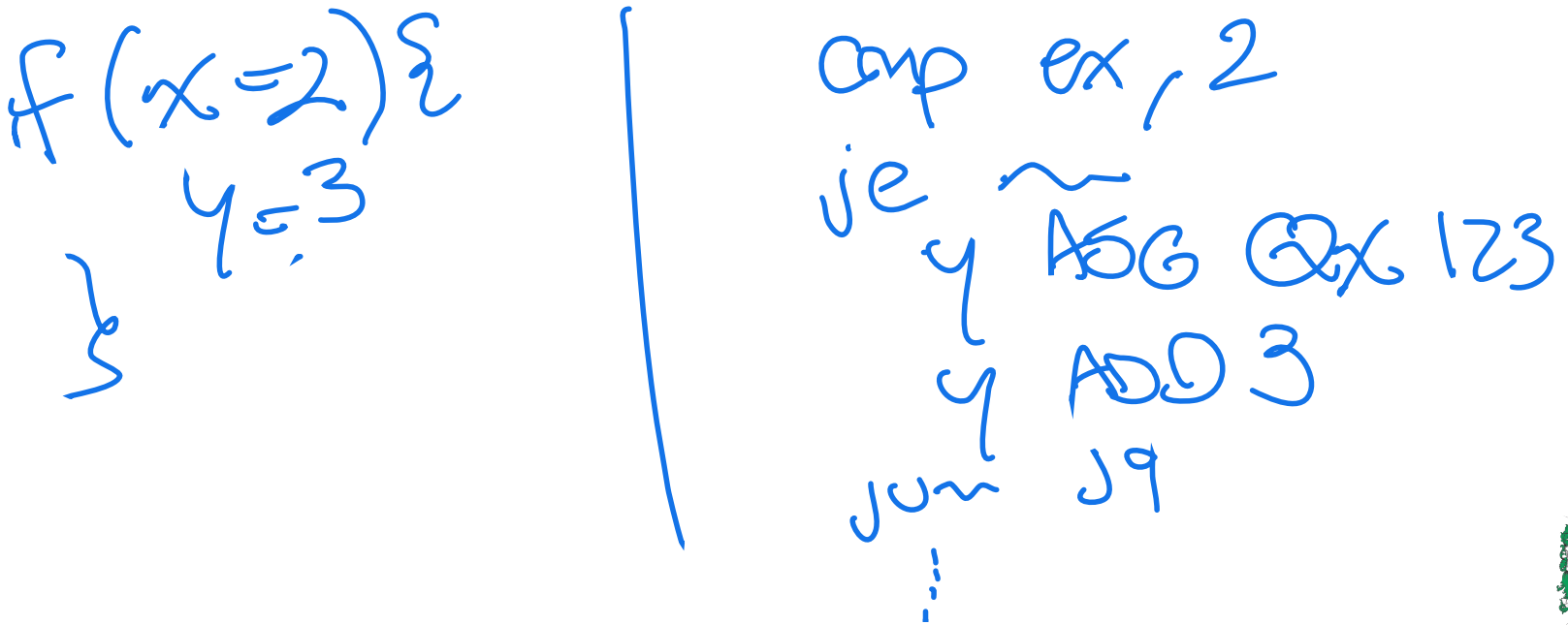
Efficiency and Readability

- **Efficiency** is the ability to avoid wasting materials, energy, efforts, money, and time in doing something or in producing a `desired result`.
- **Readability** is the ease with which a human reader can understand the purpose, control flow, and operation of source code.



Efficiency and Readability

- **Efficiency** is the ability to avoid wasting materials, energy, efforts, money, and time in doing something or in producing a `desired result`.
- **Readability** is the ease with which a human reader can understand the purpose, control flow, and operation of `source code`.



UNIVERSIDAD DISTRITAL
FRANCISCO JOSÉ DE CALDAS

Outline

1 Programming Languages

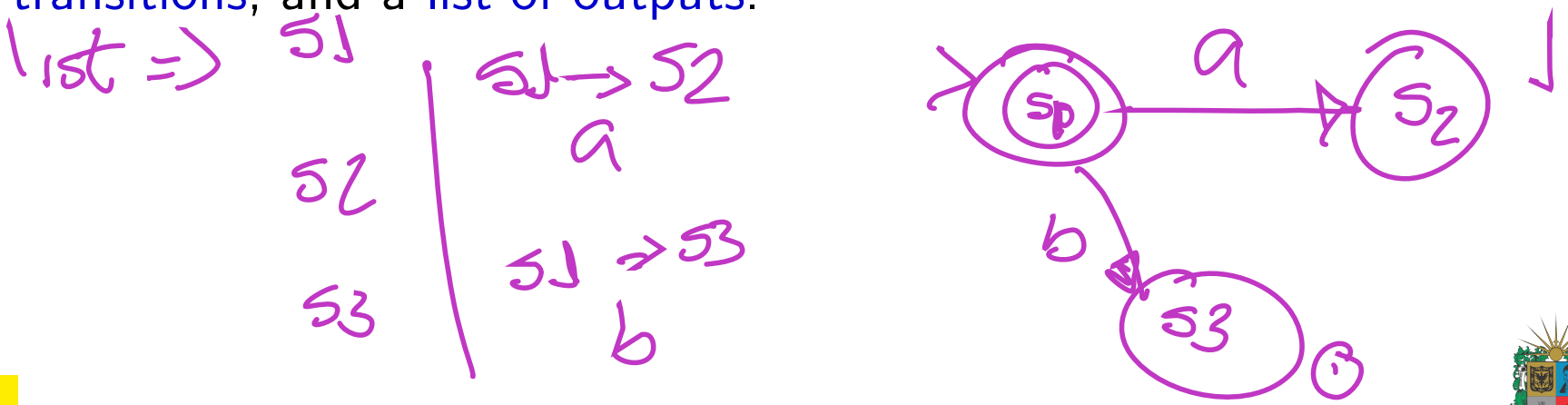
2 Finite State Machines

3 Generative Grammars

UNIVERSIDAD DISTRITAL
FRANCISCO JOSÉ DE CALDAS

Finite State Machines

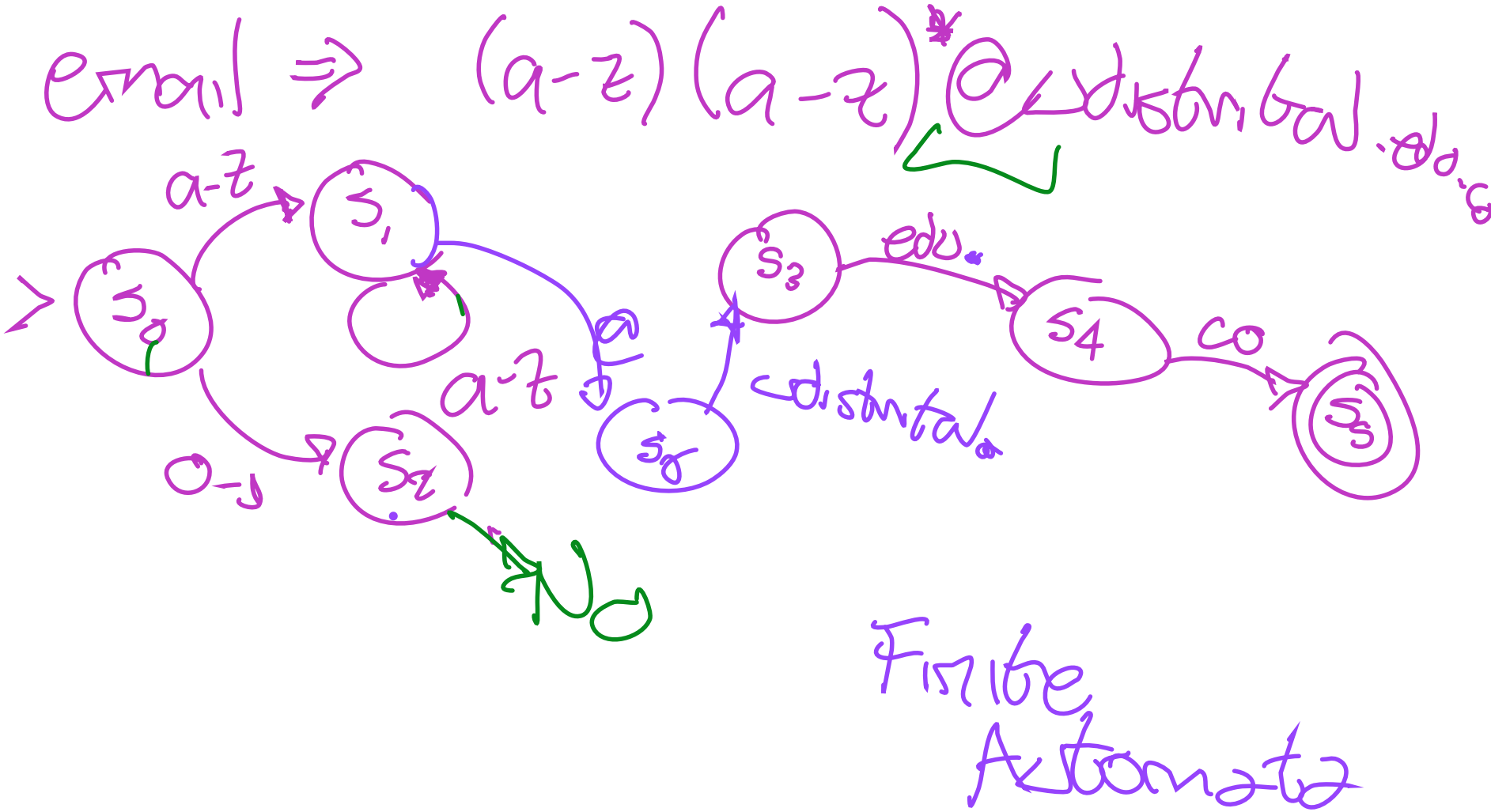
- A **finite-state machine** (FSM) is a mathematical model of computation.
- It is an abstract machine that can be in exactly one of a finite number of states at any given time.
- The FSM can change from one state to another in response to some external inputs.
- The FSM is defined by a list of states, a list of inputs, a list of transitions, and a list of outputs.



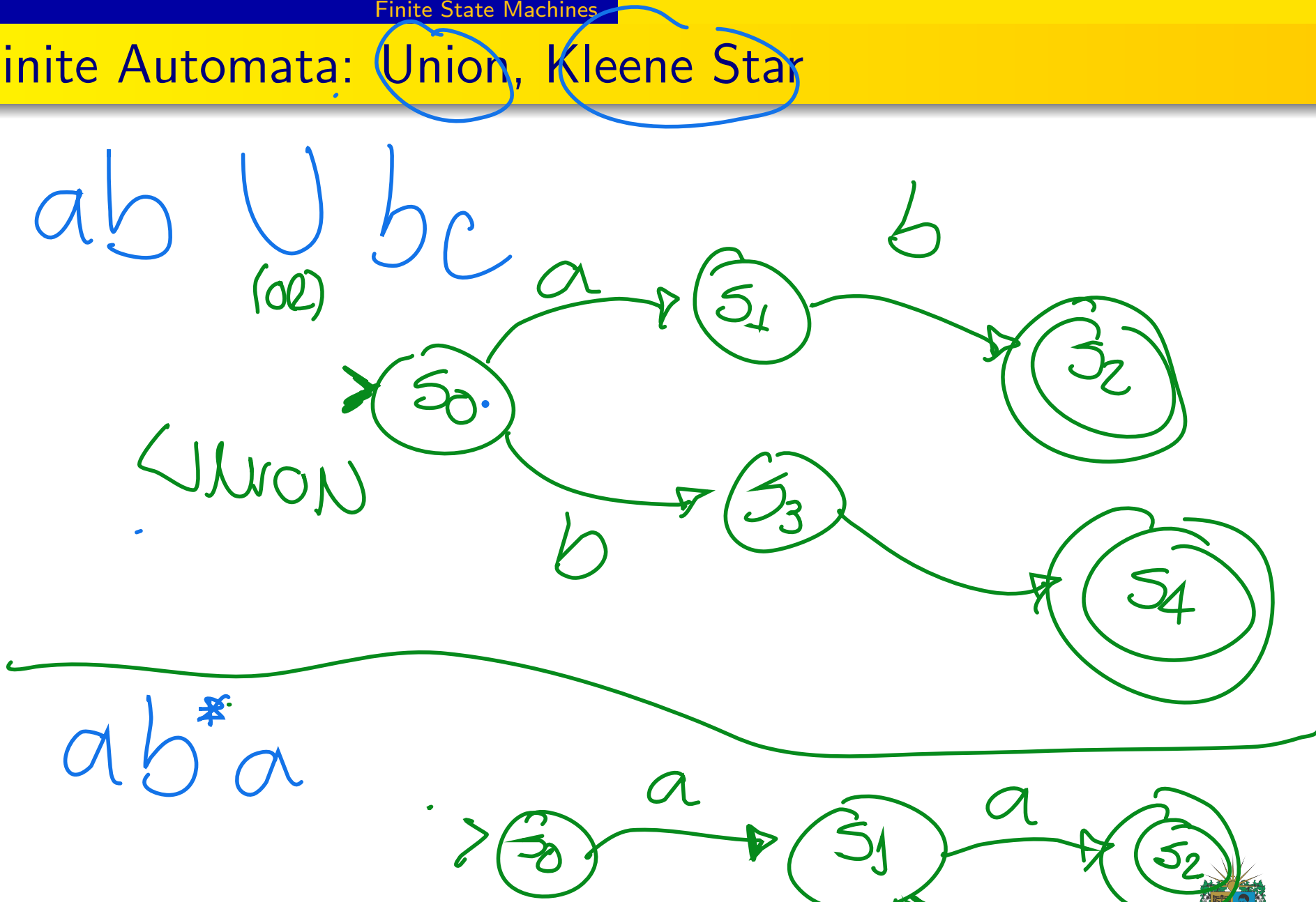
“ ”

.

Turing Master



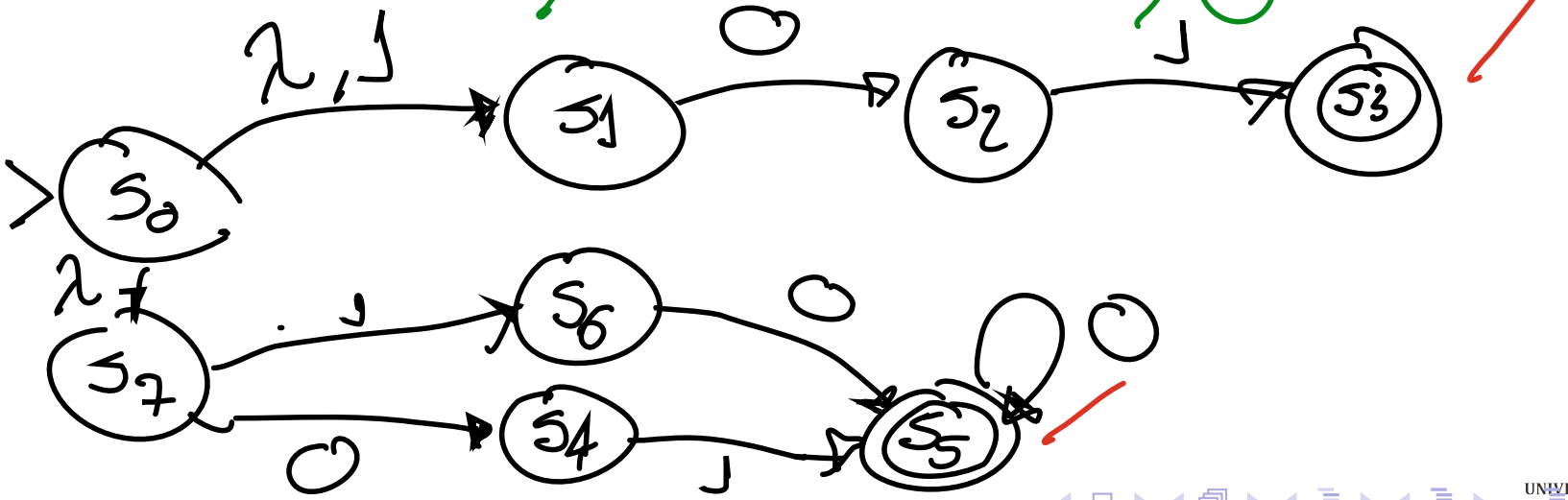
Finite Automata: Union, Kleene Star



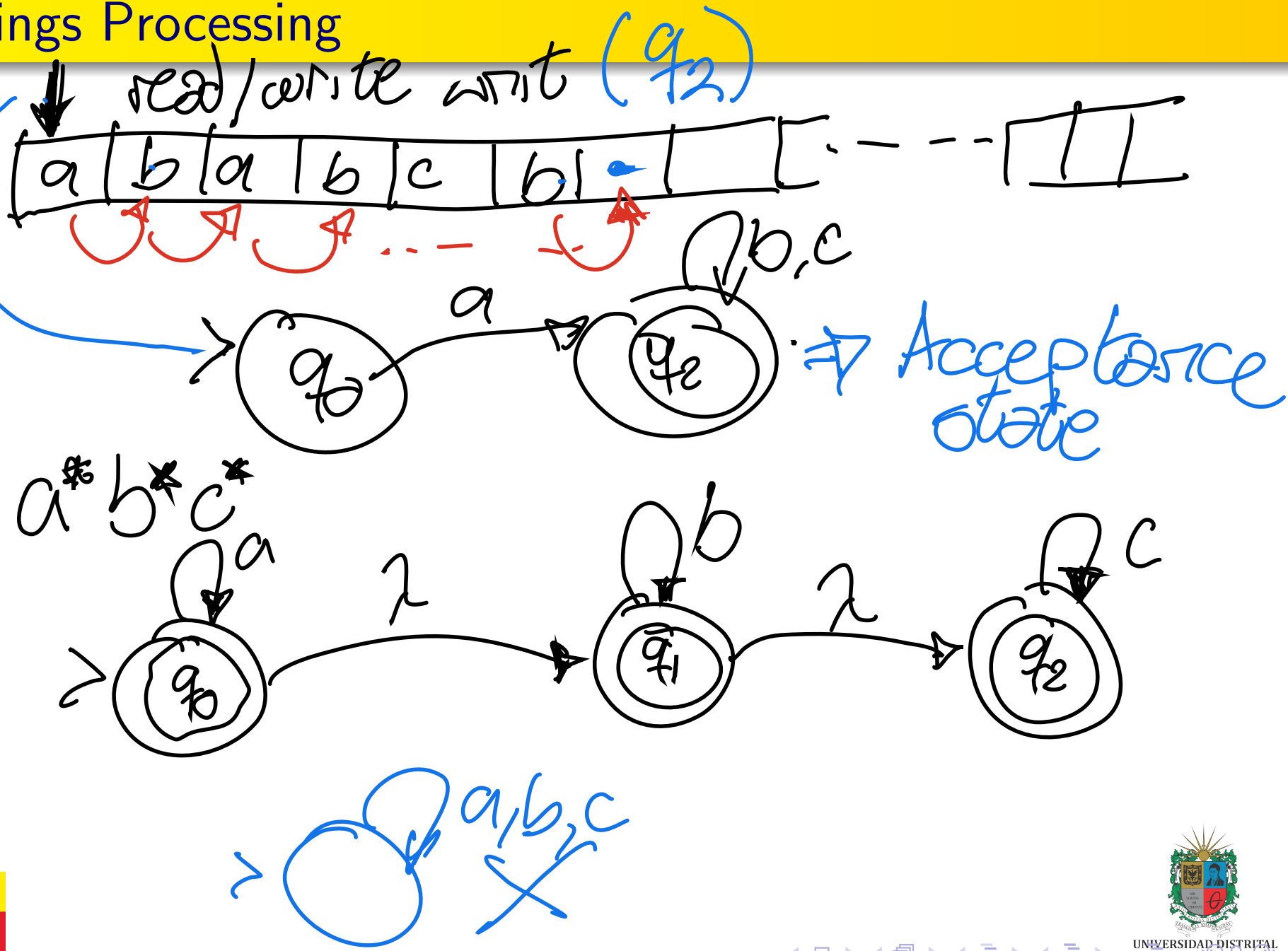
Regular Expressions

- $(bc \cup ab) \cdot c^* (ab)^*$

- $(\lambda 01 \cup 101) \cup (01 \cup 10) 0^*$



Strings Processing



Alan Turing

- **Alan Turing** (1912 — 1954) was an English mathematician, computer scientist, logician, cryptanalyst, philosopher, and theoretical biologist.
- He is widely considered to be the **father of theoretical computer science** and **artificial intelligence**.
- He was highly influential in the development of theoretical computer science, providing a formalization of the concepts of algorithm and computation with the Turing machine.

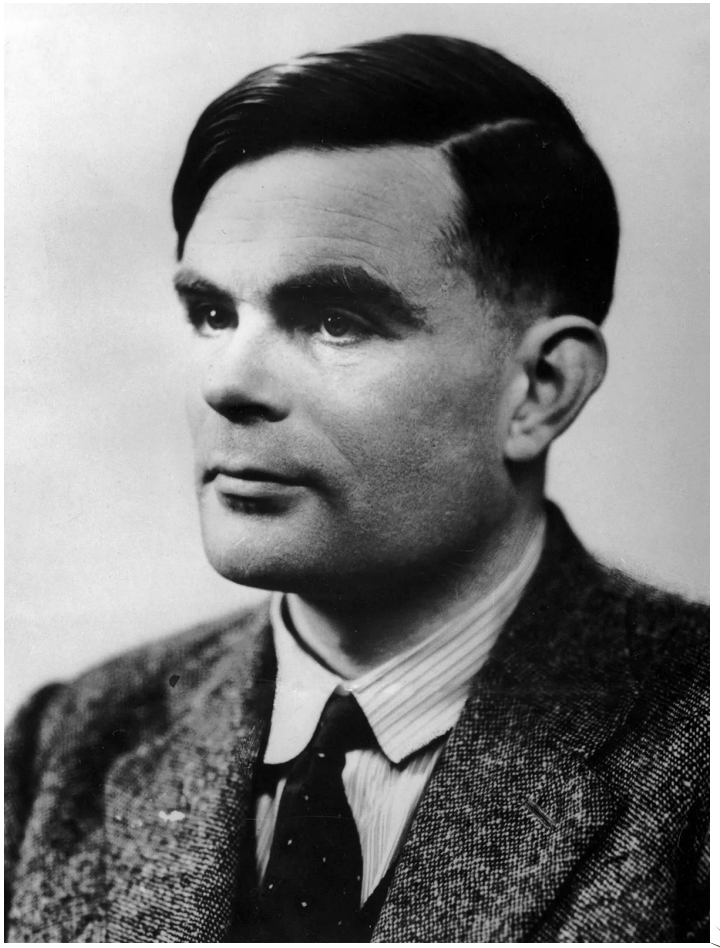
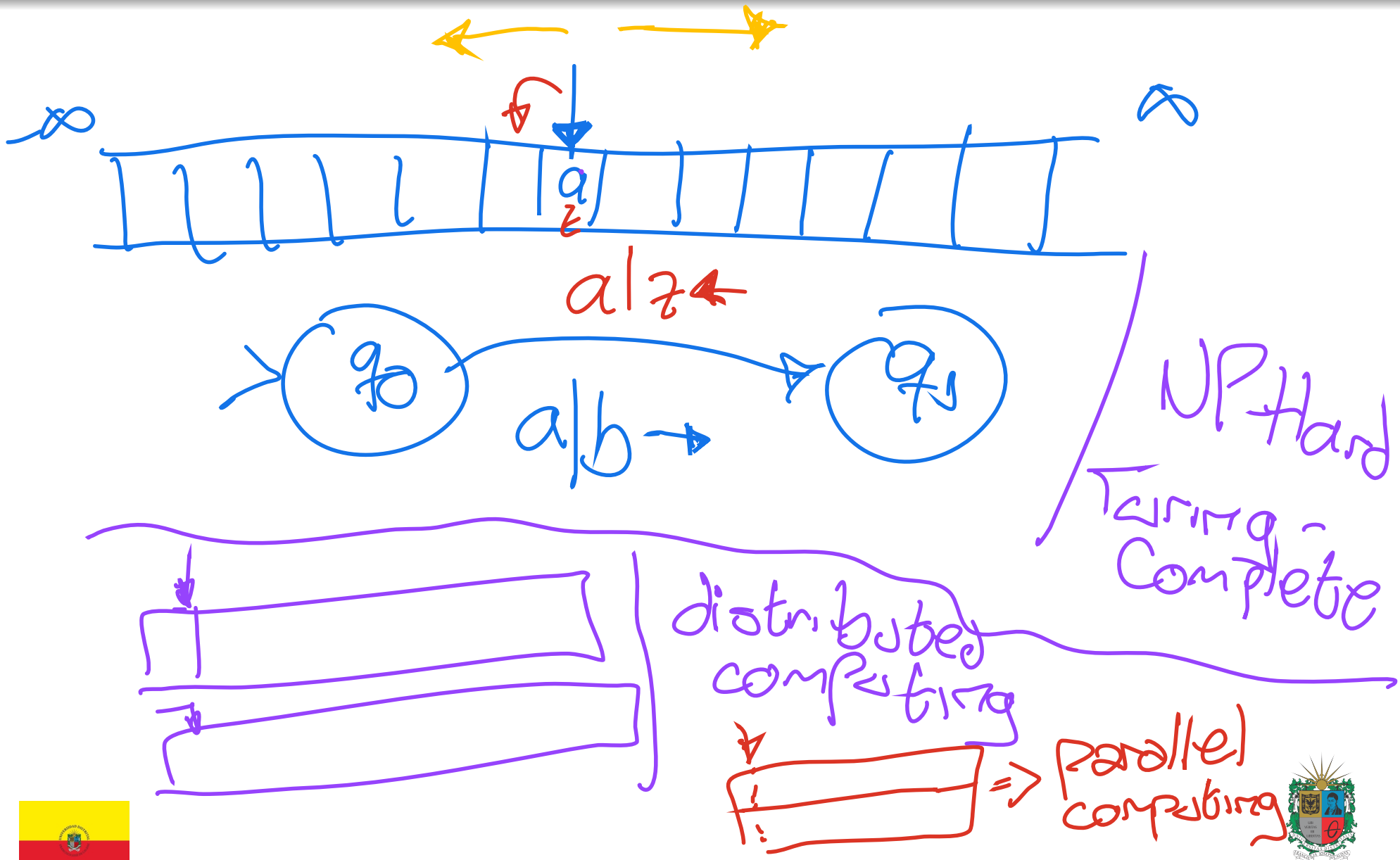
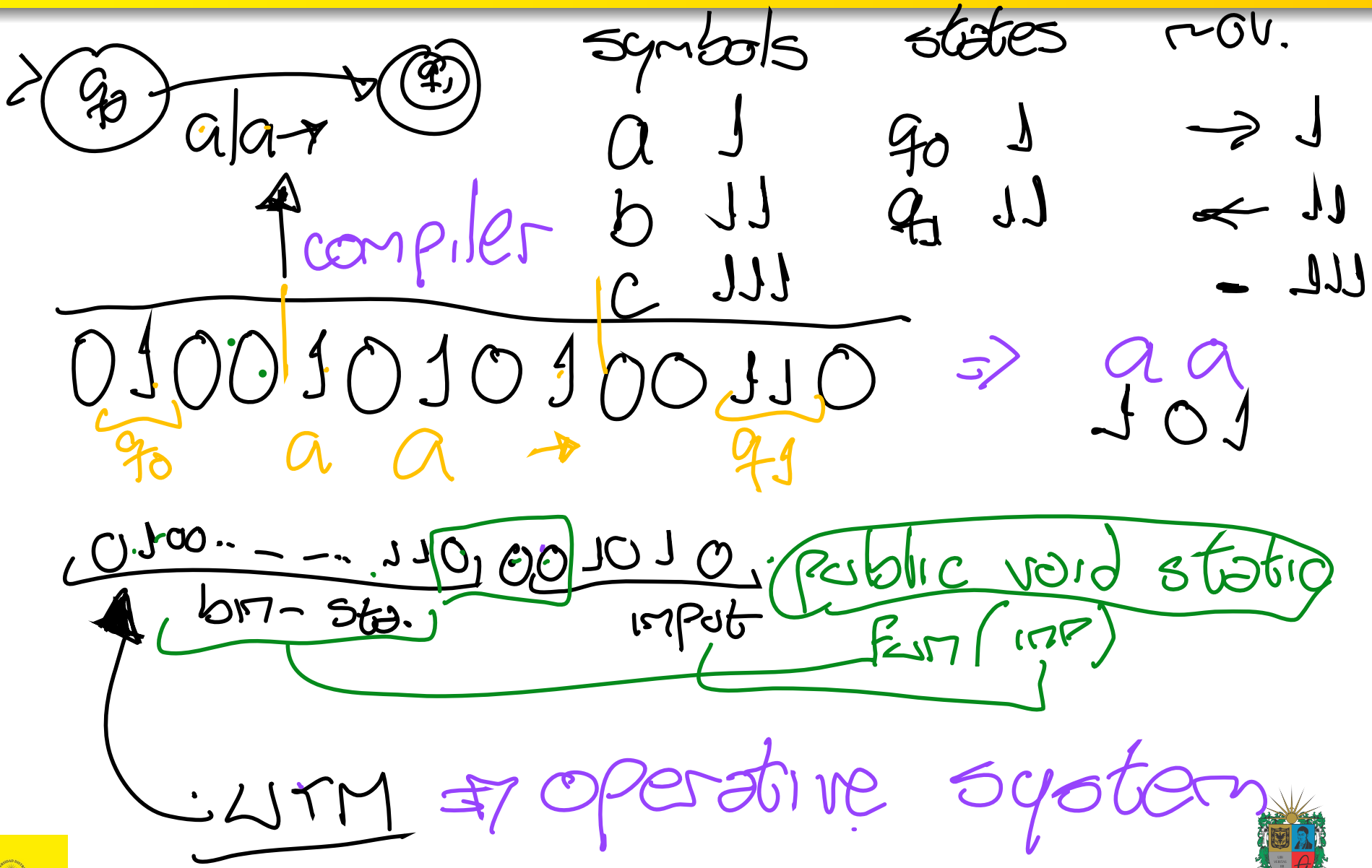


Figure: Alan Turing

Turing Machine



Universal Turing Machine



Outline

1 Programming Languages

2 Finite State Machines

3 Generative Grammars

UNIVERSIDAD DISTRITAL
FRANCISCO JOSÉ DE CALDAS

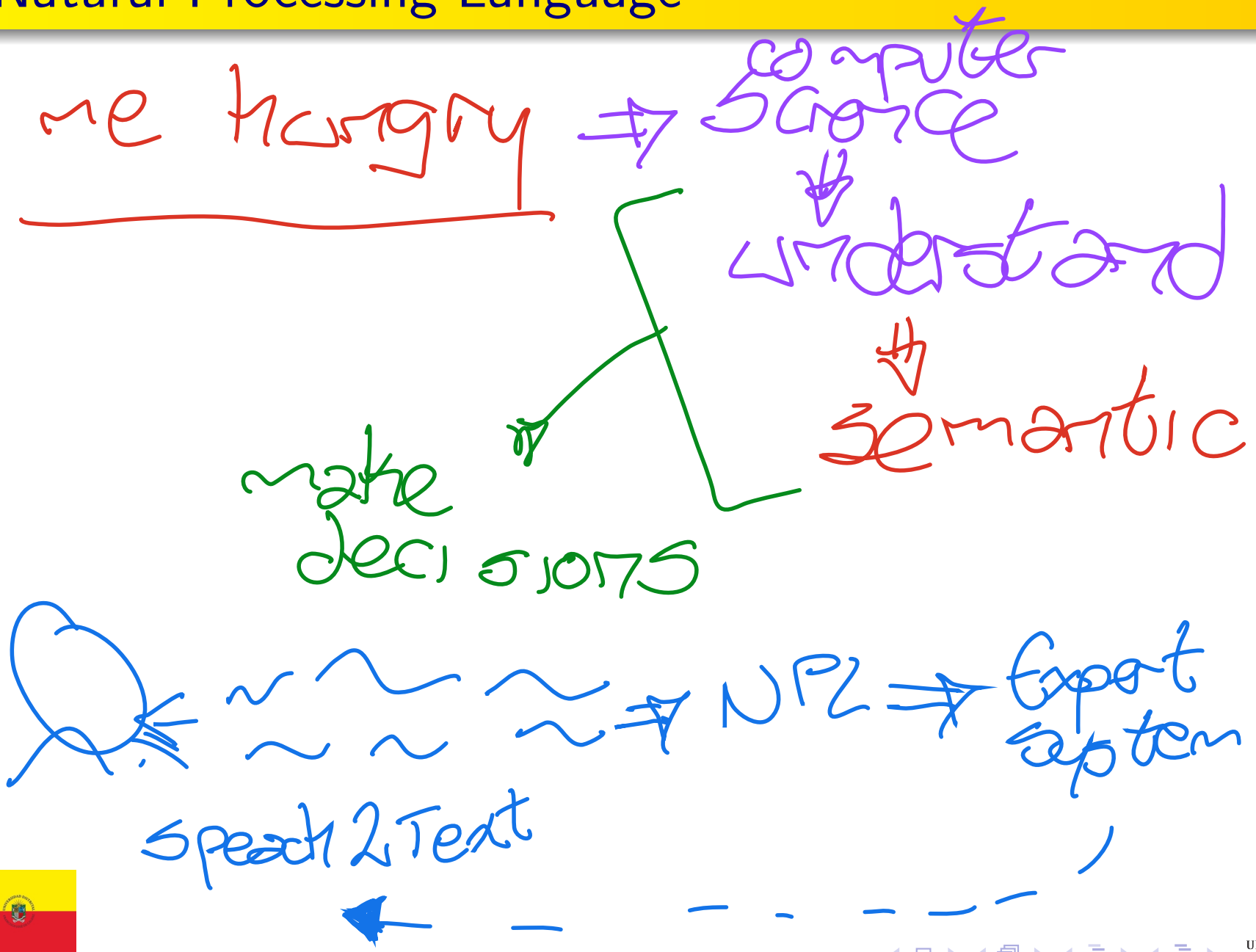
Noam Chomsky

- **Noam Chomsky** (1928 —) is an American linguist, philosopher, cognitive scientist, historian, social critic, and political activist.
- He is considered the **father of modern linguistics**.
- He introduced the **Chomsky hierarchy**, a classification of formal languages.

60's

Figure: Noam Chomsky

Natural Processing Language



Formal Languages

Grammar => rules

<sentence> : <subject> <verb> <complement>

<subject> : me | you | he | she | ...

<verb> : eat | sleep | play | ...

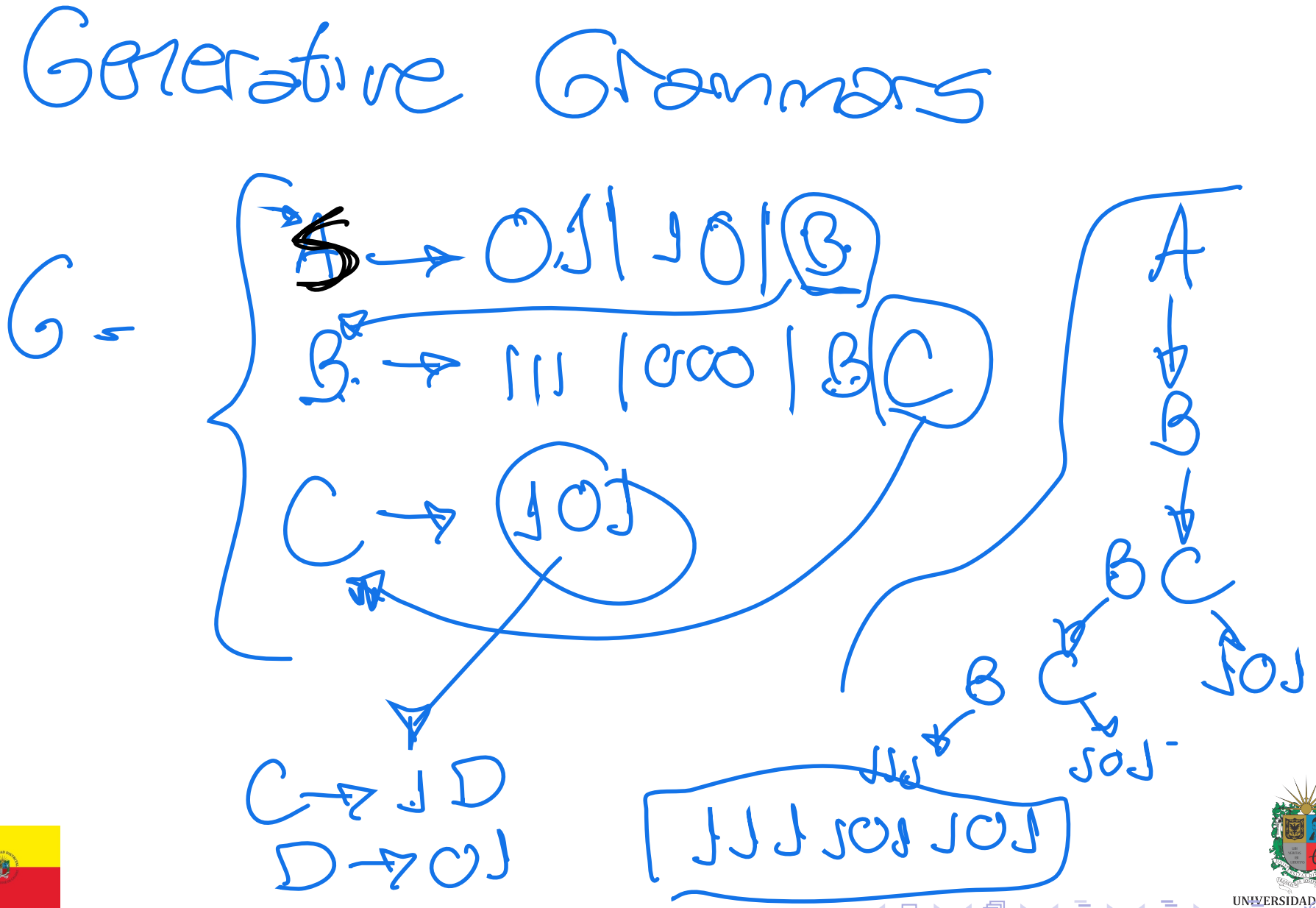
<comp> : <adj> <adv>

<adj> : ...

<adv> : ...

UNIVERSIDAD DISTRITAL
FRANCISCO JOSÉ DE CALDAS

Grammars Foundations

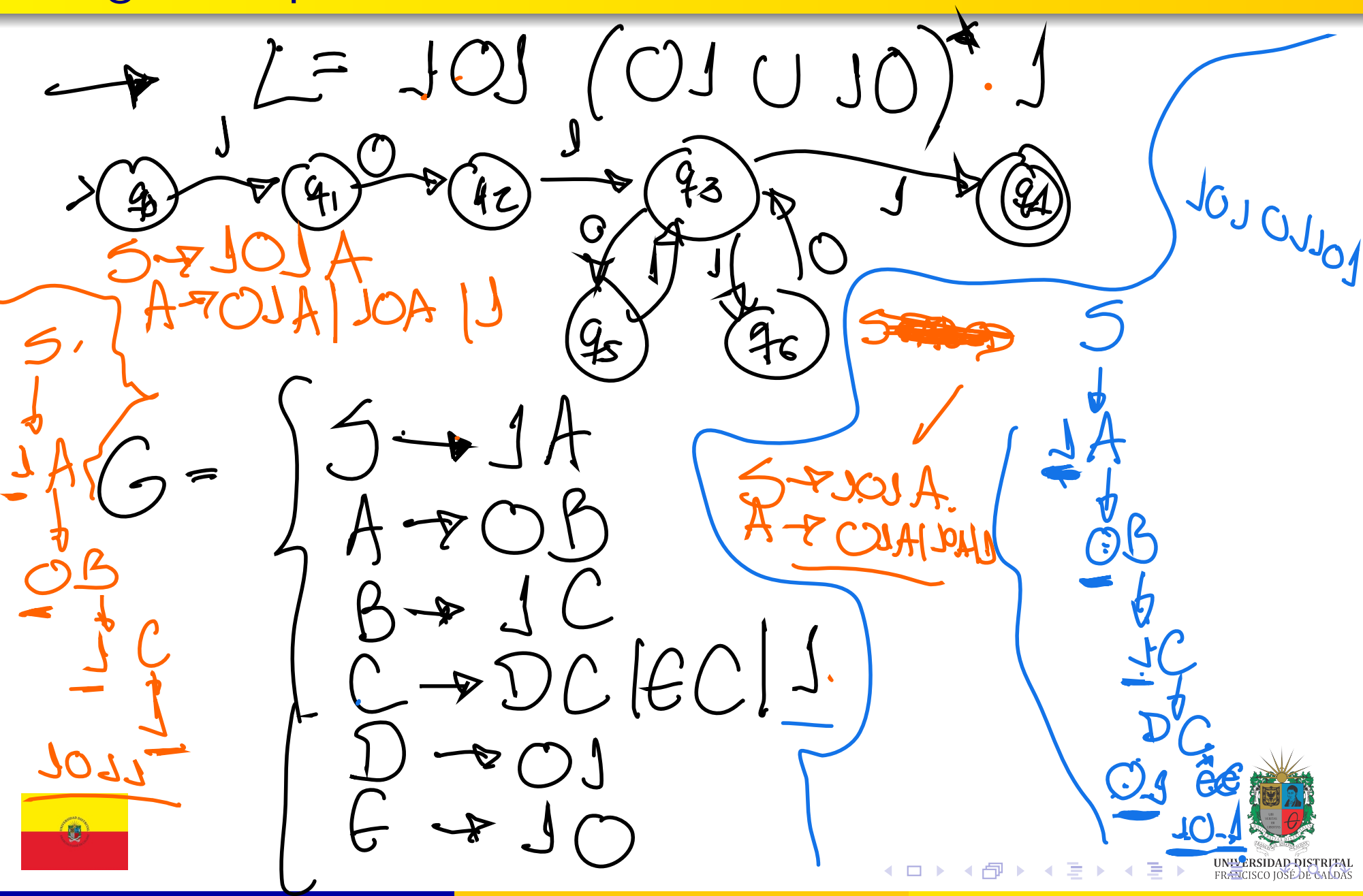


Chomsky Formal Norm

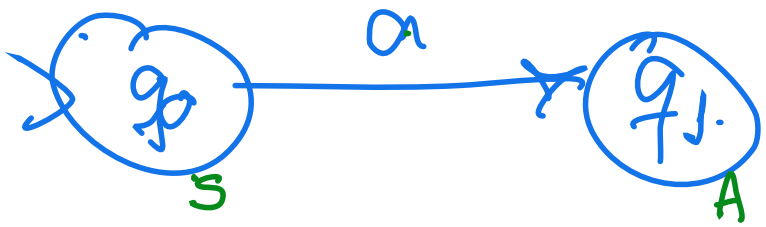
- No redundancy
- No infinite loops
- All transitions should be 2 pair (0A|1B|AB|...), except "a", "0"

UNIVERSIDAD DISTRITAL
FRANCISCO JOSÉ DE CALDAS

Regular Expressions

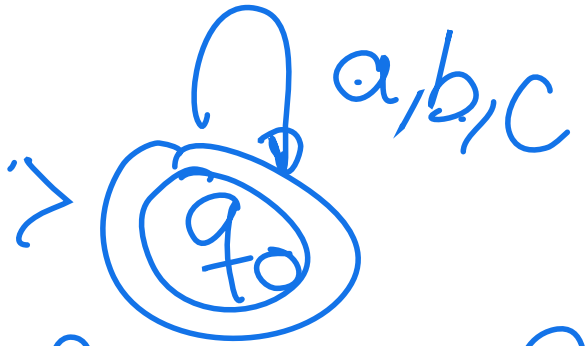


Equivalence between Grammar and Finite Automatas

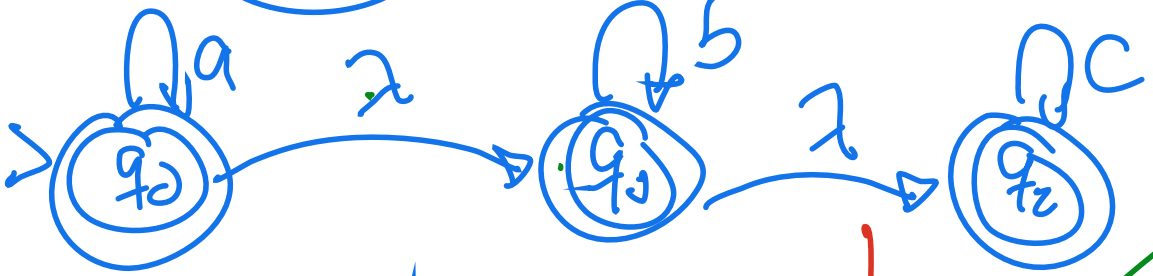


$(q_0)\ S \to aA$

$(q_1)\ A \to$



$S \to aS \mid bS \mid cS \mid \lambda$



$L = a^* b^* c^*$

$$G = \begin{cases} S \to aS \mid A \\ A \to bA \mid B \\ B \to cB \mid \lambda \end{cases}$$

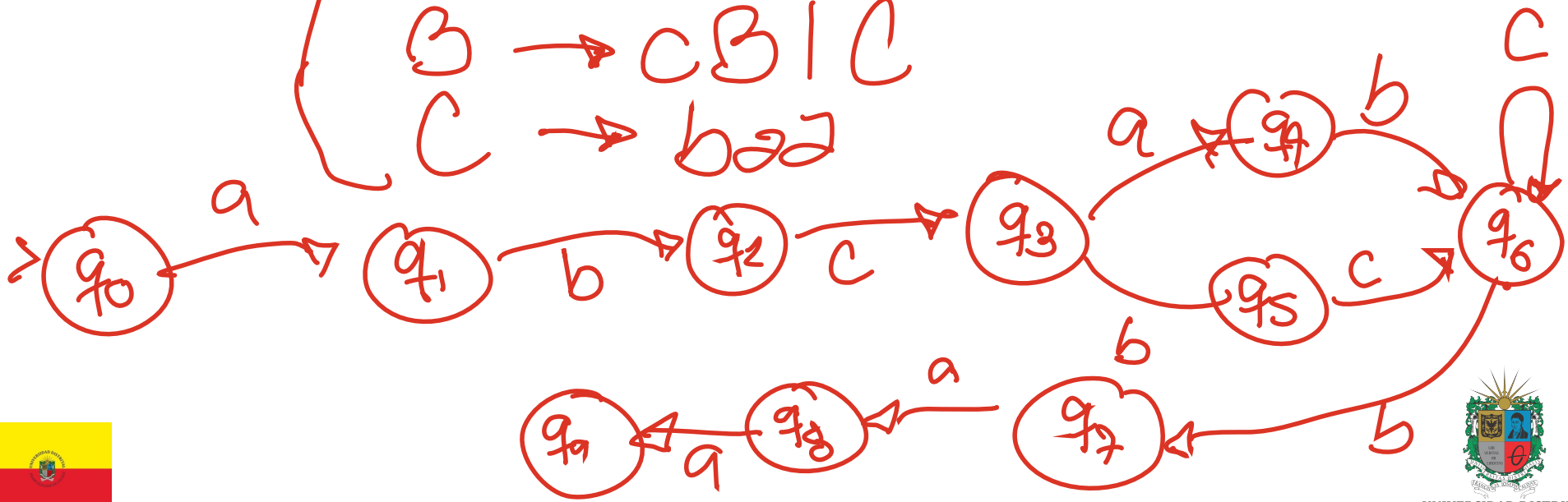
UTM ← compile ← code

UNIVERSIDAD DISTRITAL
FRANCISCO JOSÉ DE CALDAS

Grammars Example

- $L = abc(ab \cup bc)c^* baa$

$$G = \begin{cases} S \to abcA \\ A \to abB \mid bcB \\ B \to cB \mid C \\ C \to baa \end{cases}$$



UNIVERSIDAD DISTRITAL
FRANCISCO JOSÉ DE CALDAS

Outline

1 Programming Languages

2 Finite State Machines

3 Generative Grammars

UNIVERSIDAD DISTRITAL
FRANCISCO JOSÉ DE CALDAS

Thanks!

Questions?

Repo: *https://github.com/EngAndres/ud-public/tree/main/courses/computer-science-iii*

UNIVERSIDAD DISTRITAL
FRANCISCO JOSÉ DE CALDAS