The Center of Applied Data Science

Introduction to Programming

The Center of Applied Data Science

Course Content

1. A Tour of Computer Systems
2. Introduction to Algorithms
3. Basic Concepts (Python)
4. Control Structures (Python)
5. Exercises (Python)

The Center of Applied Data Science

A Tour of Computer Systems

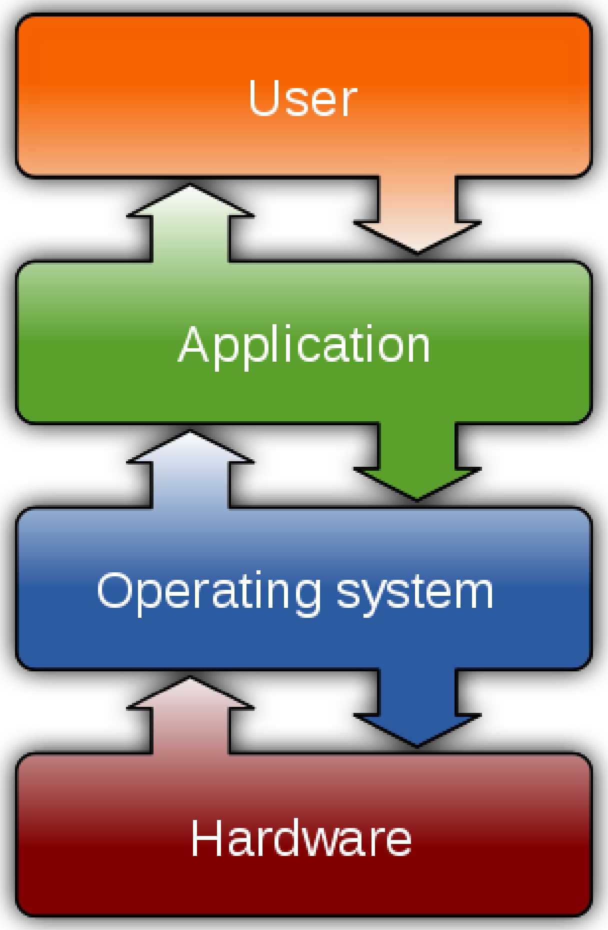
What does Computer mean?

A computer is a machine or device that performs processes, calculations and operations based on instructions provided by a software. It is designed to execute applications and provides a variety of solutions by combining integrated hardware and software components.

A computer has two primary categories:

Hardware: Physical structure that houses a computer's processor, memory, storage, communication ports and peripheral devices

Software: Includes operating system (OS) and software applications


User
Application
Operating system
Hardware

The Center of Applied Data Science

Binary Number System

A computer can understand only the “on” and “off” state of a switch. These two states are represented by 1 and 0.

The combination of 1 and 0 form binary numbers. These numbers represent various data.

As two digits are used to represent numbers, it is called a binary or base 2 number system.

The Center of Applied Data Science

Binary Number System

Converting Binary to Decimal

0	1	0	1	0	1	1	0
128	64	32	16	8	4	2	1
2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
0x128	1x64	0x32	1x16	0x8	1x4	1x2	1x0
64 + 16 + 4 + 2 = 86							

The Center of Applied Data Science

Binary Number System

Converting Decimal to Binary

2	25		
2	12	1	← 1st Remainder
2	6	0	← 2nd Remainder
2	3	0	← 3rd Remainder
2	1	1	← 4th Remainder
	0	1	← 5th Remainder

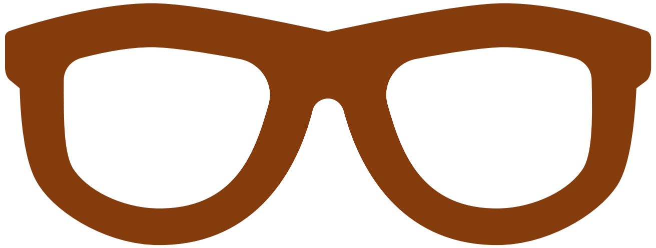
Read Upwards

Binary Number = 11001

The Center of Applied Data Science

Binary Number System

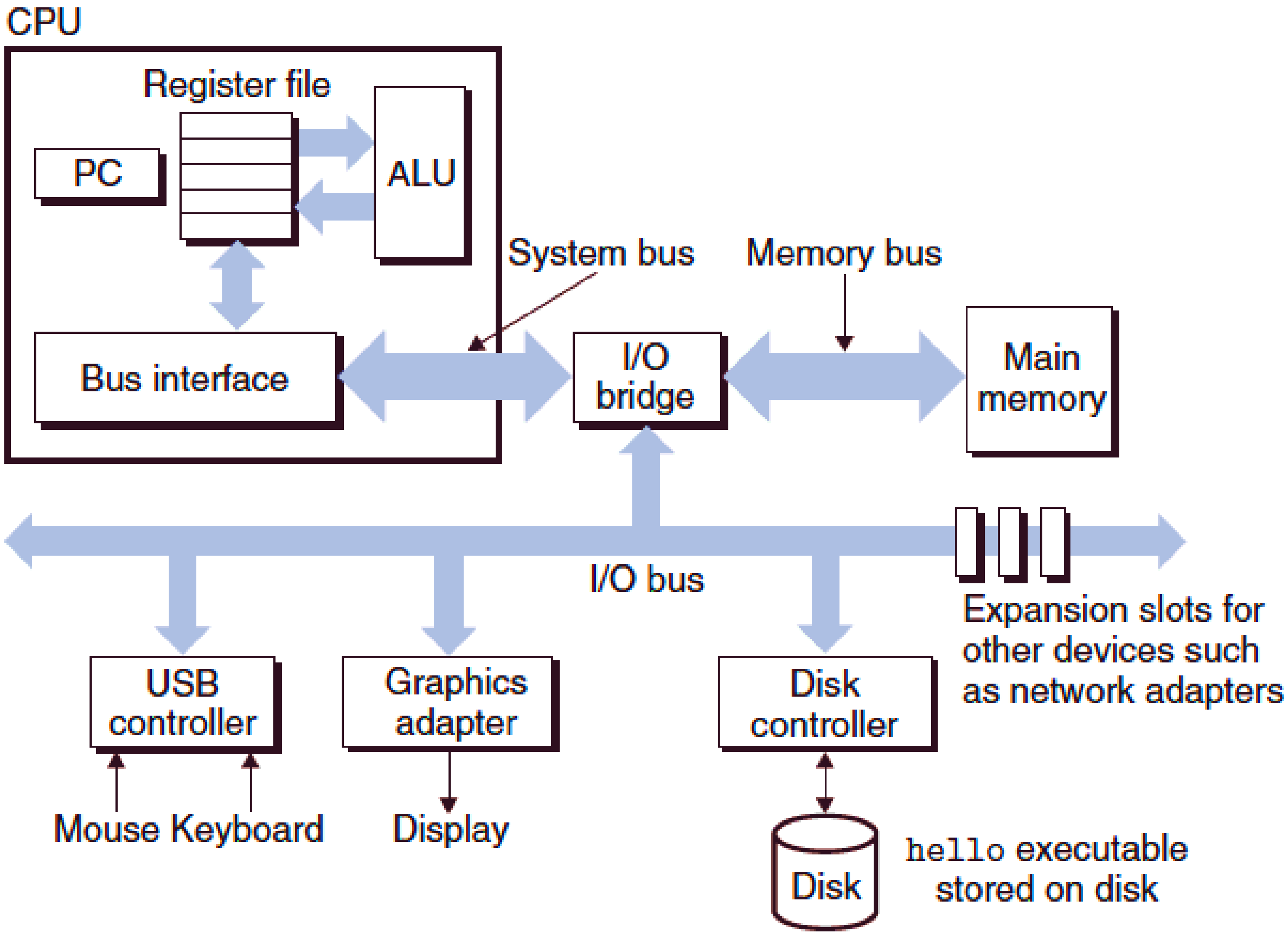
2^4	2^3	2^2	2^1	2^0	
16	8	4	2	1	
0	0	0	0	0	0 0



The Center of Applied Data Science

Computer Hardware

To understand what happens to a program when we run it, we need to understand the hardware organization of a typical system, which is shown in the next slide.


CPU
Register file
PC
ALU
System bus
Memory bus
Bus interface
I/O bridge
Main memory
I/O bus
Expansion slots for other devices such as network adapters
USB controller
Graphics adapter
Disk controller
Mouse Keyboard
Display
Disk
hello executable stored on disk

The Center of Applied Data Science

Buses

- Running throughout the system is a collection of electrical conduits called buses that carry bytes of information back and forth between the components.

- Buses are typically designed to transfer fixed-sized chunks of bytes known as words. The number of bytes in a word (the word size) is a fundamental system parameter that varies across systems.

- Most machines today have word sizes of either 4 bytes (32 bits) or 8 bytes (64 bits).

I/O Devices

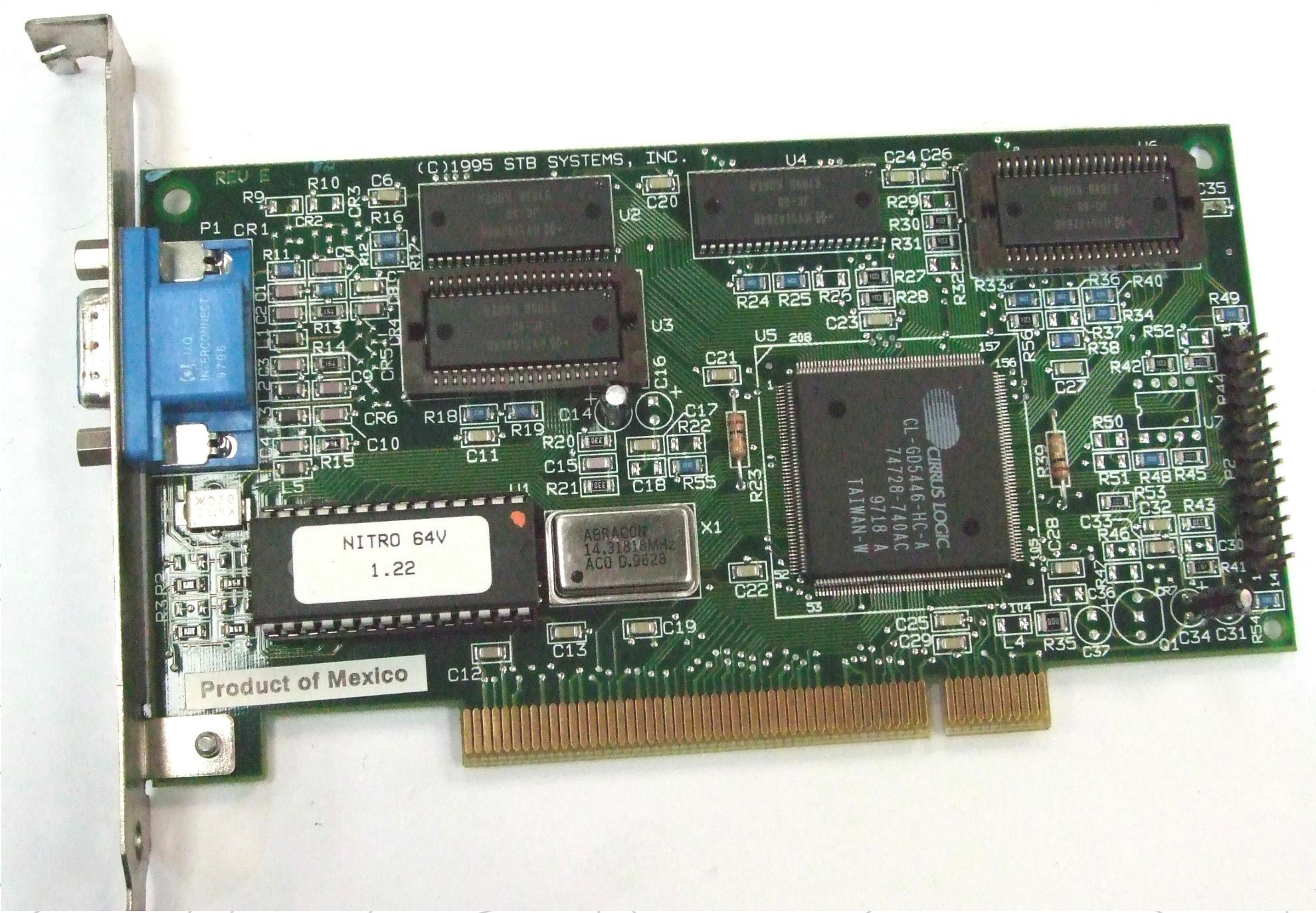
- Input/output (I/O) devices are the system's connection to the external world.

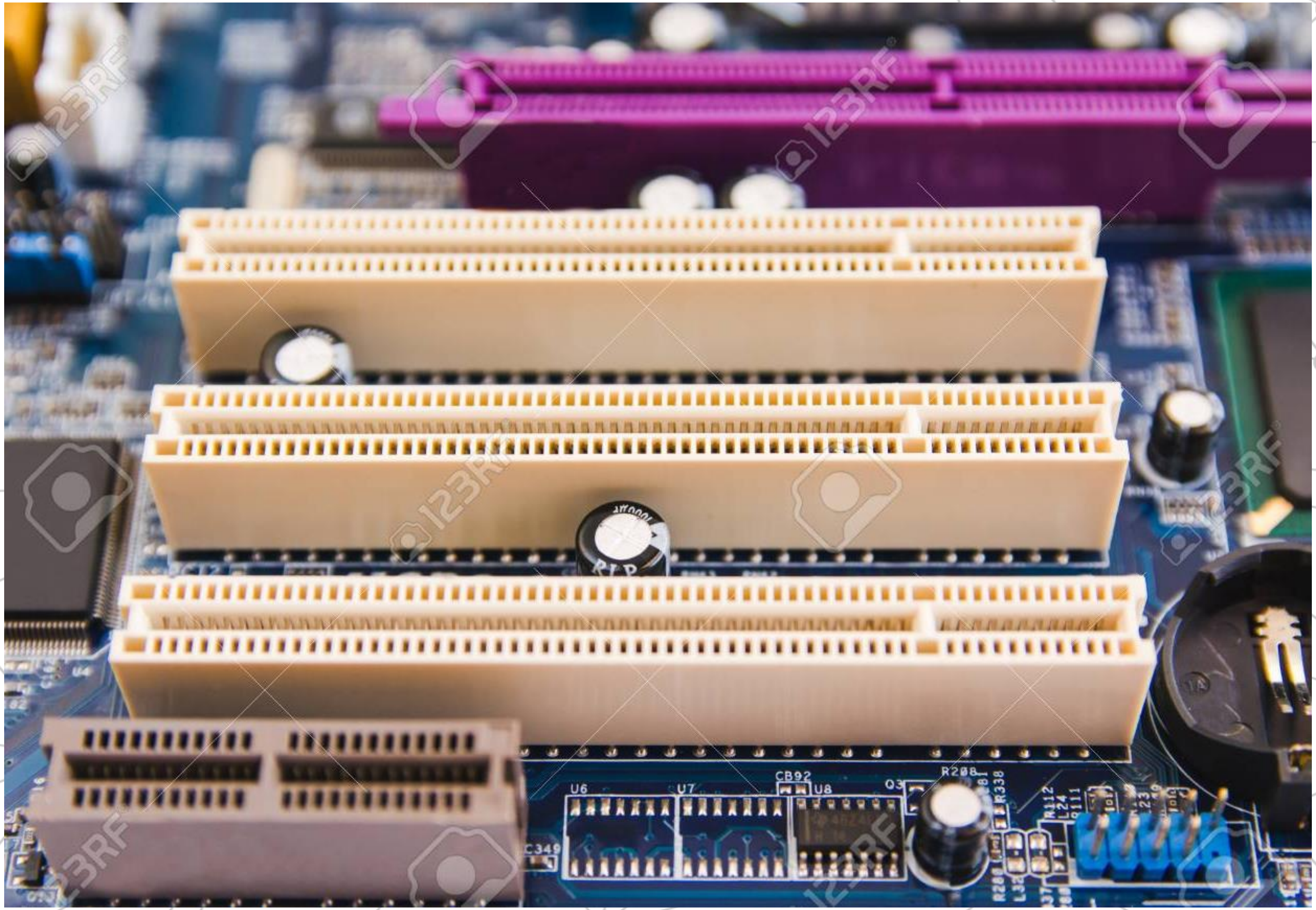
- Each I/O device is connected to the I/O bus by either a controller or an adapter.

- Each I/O device is connected to the I/O bus by either a controller or an adapter.

- The distinction between the two is mainly one of packaging. Controllers are chipsets in the device itself or on the system's main printed circuit board (often called the motherboard). An adapter is a card that plugs into a slot on the motherboard.

15-98
JUN
JC-07 V0
U1
L2
C
D
V
G
G
R2593
SILITEK 1.0
MPA41000
9711 QM
L1
R1
NUM
CAPS
J12
J9
J10
J11
Silitek
PBAAS217
1
9
10
CN2


(C)1995 STB SYSTEMS, INC.
REV E
NITRO 64V
1.22
ABRACON
14.31818MHz
ACO D.9628
CIRRUS LOGIC
CL-GD5446-HC-A
74728-740AC
9718 A
TAIWAN-W
Product of Mexico



The Center of Applied Data Science

Main Memory

- The main memory is a temporary storage device that holds both a program and the data it manipulates while the processor is executing the program.
- Physically, the main memory consists of a collection of dynamic random access memory (DRAM) chips.
- Logically, memory is organized as a linear array of bytes, each with its unique address (array index) starting at zero.
- In general, each of the machine instructions that constitute a program can consist of a variable number of bytes.

The Center of Applied Data Science

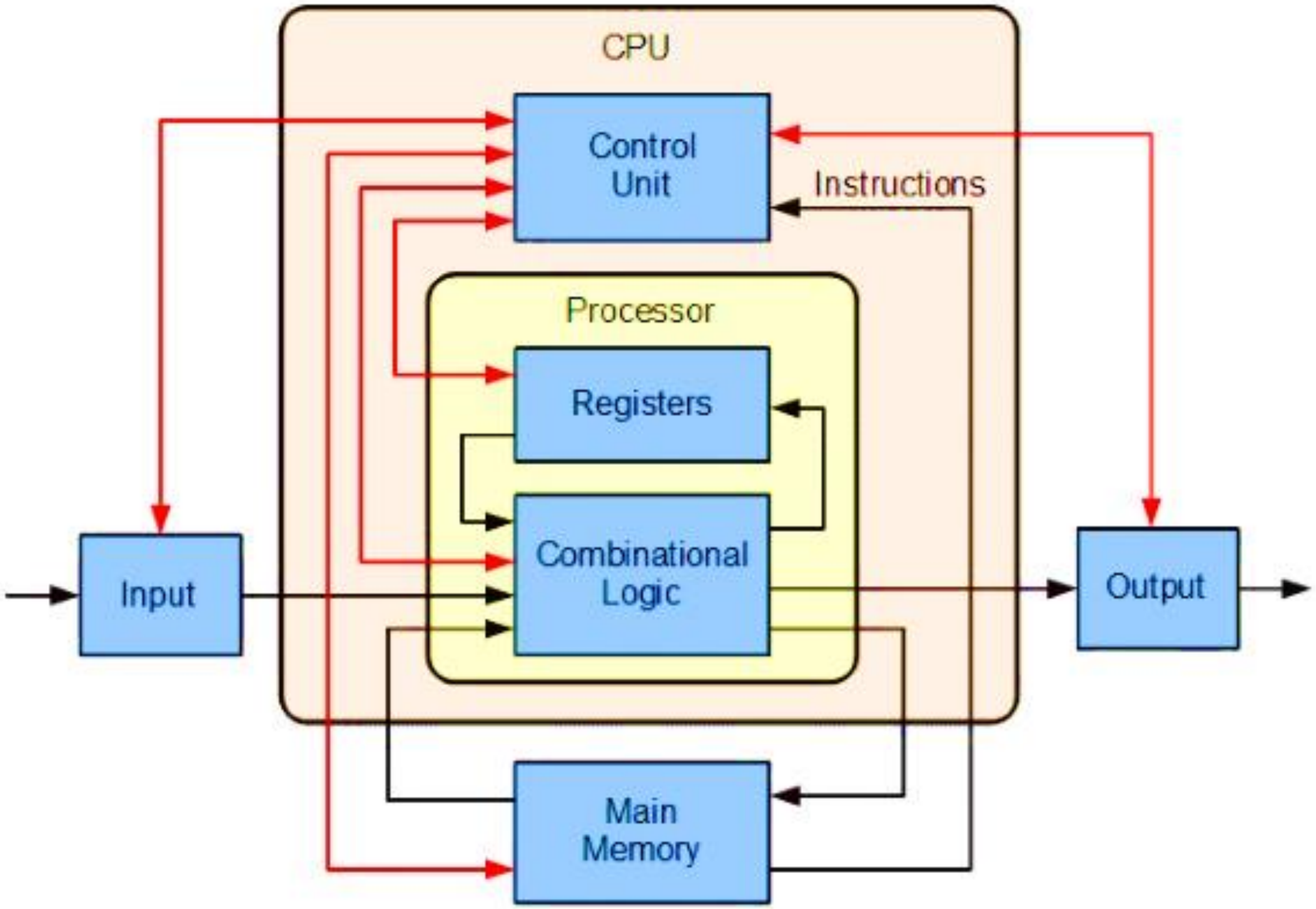
Processor

- The central processing unit (CPU), or simply processor, is the engine that interprets (or executes) instructions stored in the main memory.

- At its core is a word-sized storage device (or register) called the program counter (PC).

- At any point in time, the PC points at (contains the address of) some machine-language instruction in main memory.

The Center of Applied Data Science



The Center of Applied Data Science

What is an Operating System?

An **operating system** (OS) is system software that manages computer hardware and software resources and provides common services for computer programs.

Computer Software

Computer software, also called software or application, is a set of instructions and its documentations that tells a computer what to do or how to perform a task.

Software includes all different software programs on a computer, such as applications and the operating system.

The next chapter details the set of instructions in a computer software, also known as **algorithms.**

The Center of Applied Data Science

Introduction to Algorithms

The Center of Applied Data Science

1. **What is an Algorithm?**
2. Describing Algorithms with Flowcharts
3. Introduction to Programming

Definition

A set of step-by-step instructions to solve a problem.

Requirements

- Can be described in a formal language
- Consist of a finite number of steps
- Operate on zero or more inputs
- Result in an output
- Individual steps are sufficiently basic and can be executed in finite time

What is an Algorithm?
Examples of Algorithms

- Calculate the sum of the first 10 positive numbers
- Recipe for baking a cake
- Recommend products to consumers based on previous transactions

The Center of Applied Data Science

Exercise 1

Write an algorithm to find the page number of the chapter *Little Em'ly* of the book *David Copperfield* by Charles Dickens.

Write an algorithm to find the page number of the chapter *Little Em'ly* of the book *David Copperfield* by Charles Dickens.

Solution A:

- Open book
- Flip through pages we see the chapter title *Little Em'ly* at the top of the page
- Write down page number

Solution B:

- Open book
- Turn to the table of contents
- Write down page number of chapter *Little Em'ly*

Write an algorithm to find the page number of the chapter *Little Em'ly* of the book *David Copperfield* by Charles Dickens.

Solution A:

- Open book
- Flip through pages we see the chapter title *Little Em'ly* at the top of the page
- Write down page number

Solution B:

- Open book
- Turn to the table of contents
- Write down page number of chapter *Little Em'ly*

Many different solutions are possible!

Exercise 2

Write an algorithm that takes two numbers and adds their squares.

Write an algorithm that takes two numbers and adds their squares.

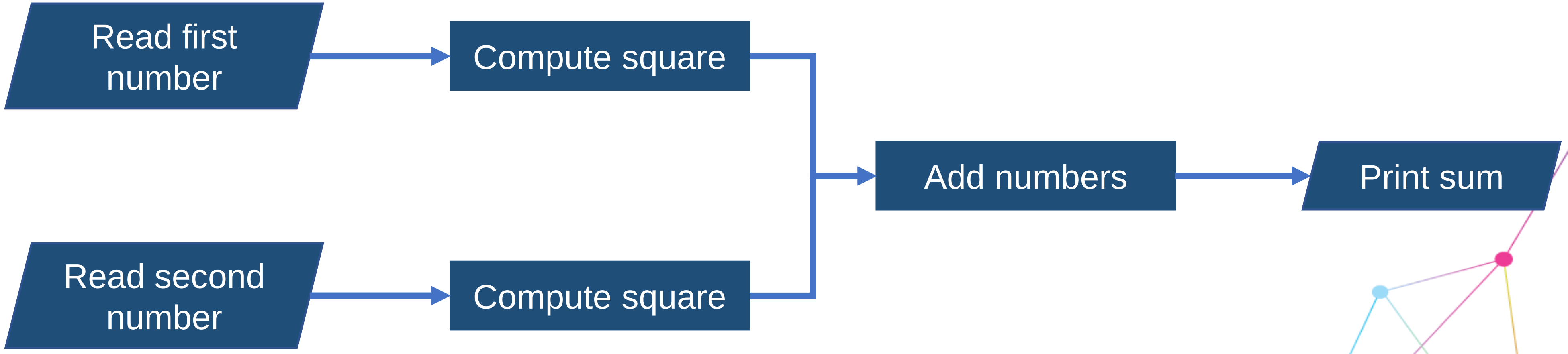
Write an algorithm that takes two numbers and adds their squares.

The Center of Applied Data Science

1. What is an Algorithm?
2. **Describing Algorithms with Flowcharts**
3. Introduction to Programming

Describing Algorithms with Flowcharts
What are Flowcharts?

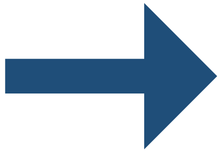
Flowcharts are connected sequences of instructions



- Allows for easy visualization of algorithmic logic
- Provides a common language for algorithmic logic
- Typically go from top to bottom and left to right

The Center of Applied Data Science

Describing Algorithms with Flowcharts
Common Flowchart Symbols



- **Flowline**: Shows the flow of the algorithm

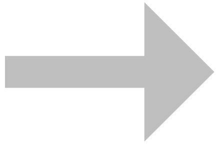
Symbols are defined by the International Standards Organization (ISO). More info at https://en.wikipedia.org/wiki/Flowchart#Common_symbols

Describing Algorithms with Flowcharts
Common Flowchart Symbols

- **Flowline**: Shows the flow of the algorithm
- **Input/Output**: Data read or produced by the algorithm. Represented by a parallelogram.

Symbols are defined by the International Standards Organization (ISO). More info at https://en.wikipedia.org/wiki/Flowchart#Common_symbols

Describing Algorithms with Flowcharts
Common Flowchart Symbols



- **Flowline**: Shows the flow of the algorithm

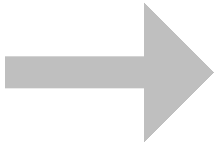
- **Input/Output**: Data read or produced by the algorithm. Represented by a parallelogram.

- **Process**: An action, e.g. addition. Represented by a rectangle

Symbols are defined by the International Standards Organization (ISO). More info at https://en.wikipedia.org/wiki/Flowchart#Common_symbols

The Center of Applied Data Science

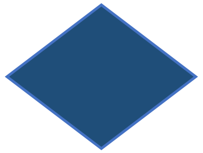
Describing Algorithms with Flowcharts
Common Flowchart Symbols



- **Flowline**: Shows the flow of the algorithm

- **Input/Output**: Data read or produced by the algorithm. Represented by a parallelogram.

- **Process**: An action, e.g. addition. Represented by a rectangle

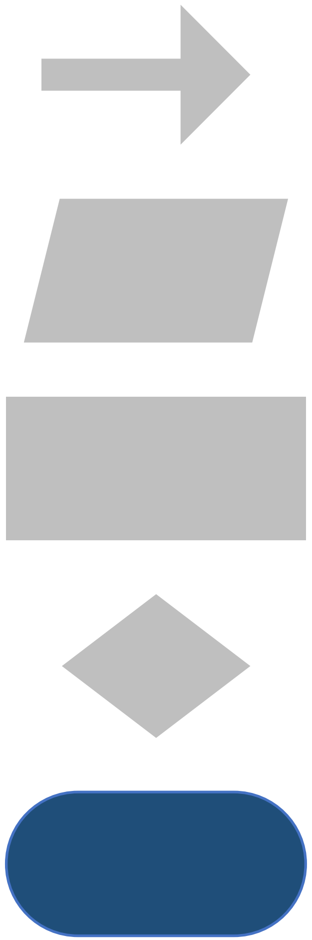


- **Decision**: A conditional query that determines the path the program will take. Commonly a yes/no question. Represented by a diamond.

Symbols are defined by the International Standards Organization (ISO). More info at https://en.wikipedia.org/wiki/Flowchart#Common_symbols

The Center of Applied Data Science

Describing Algorithms with Flowcharts
Common Flowchart Symbols



- **Flowline**: Shows the flow of the algorithm
- **Input/Output**: Data read or produced by the algorithm. Represented by a parallelogram.
- **Process**: An action, e.g. addition. Represented by a rectangle
- **Decision**: A conditional query that determines the path the program will take. Commonly a yes/no question. Represented by a diamond.
- **Terminal**: The beginning and end of an algorithm. Represented by a stadium (rectangle with half-circles on either side).

Symbols are defined by the International Standards Organization (ISO). More info at https://en.wikipedia.org/wiki/Flowchart#Common_symbols

Describing Algorithms with Flowcharts
Common Flowchart Symbols

- **Flowline**: Shows the flow of the algorithm
- **Input/Output**: Data read or produced by the algorithm. Represented by a parallelogram.
- **Process**: An action, e.g. addition. Represented by a rectangle
- **Decision**: A conditional query that determines the path the program will take. Commonly a yes/no question. Represented by a diamond.
- **Terminal**: The beginning and end of an algorithm. Represented by a stadium (rectangle with half-circles on either side).

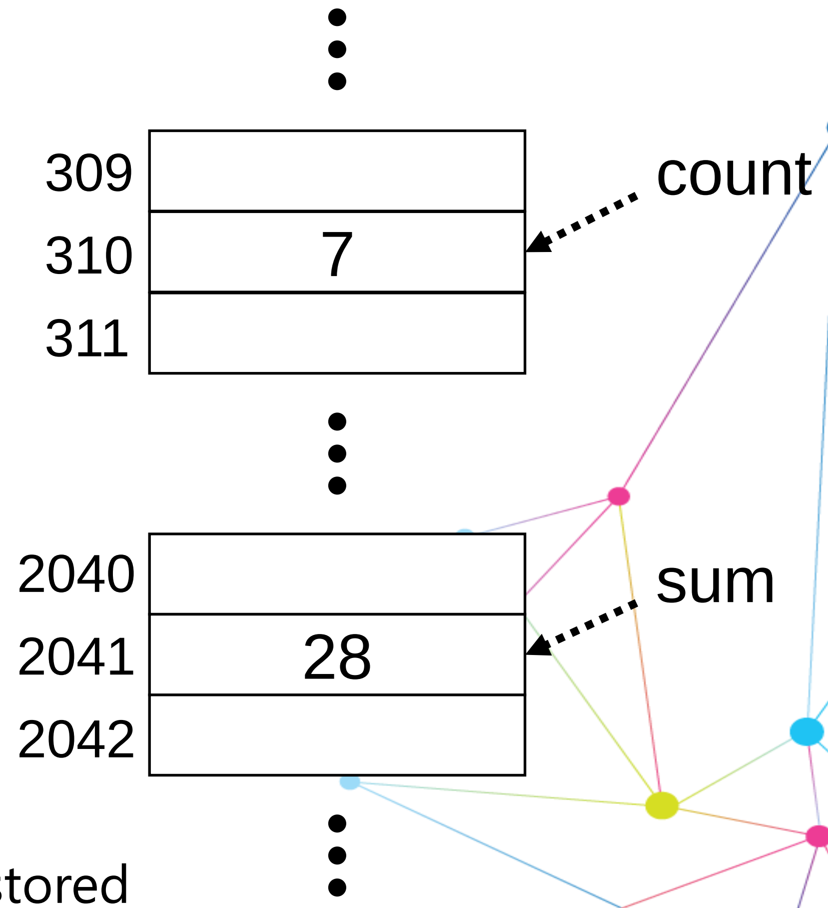
Symbols are defined by the International Standards Organization (ISO). More info at https://en.wikipedia.org/wiki/Flowchart#Common_symbols

Programming Elements
Variables

Definition

Variables are symbols that represent underlying, changeable values, e.g.

- words: Name = “Alex”
- numbers: Age = 50

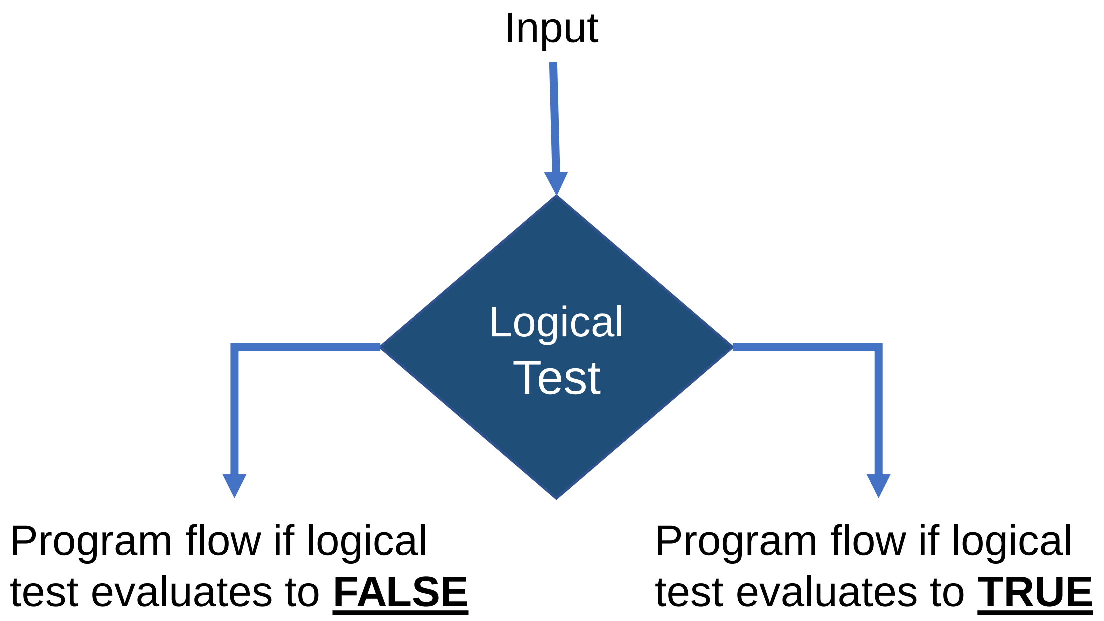


In computers

- Variables reference blocks of memory at which data is stored

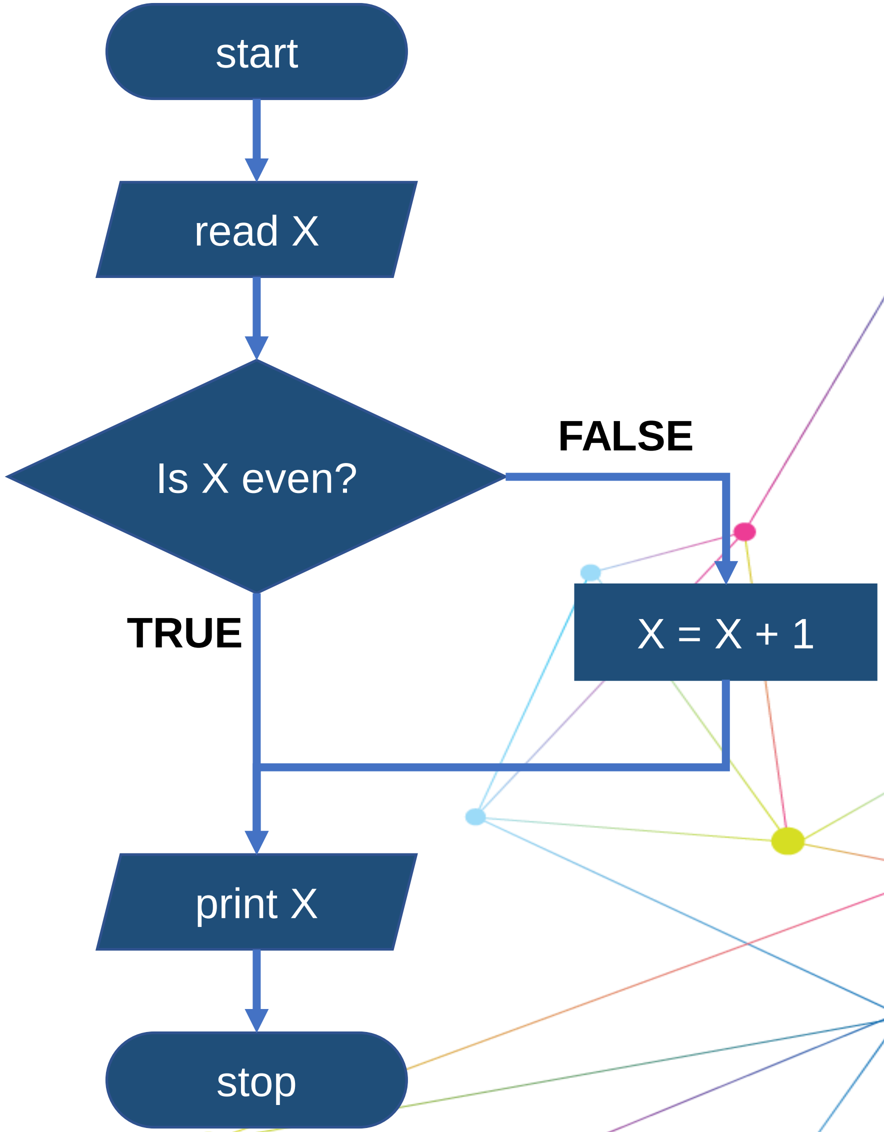
Definition

Conditional statements, also called if-else statements, perform a logical test and direct algorithm flow depending on the output.



Conditional statements can also be used to skip over an action,

- e.g. an algorithm that adds 1 to odd inputs but leaves even inputs unchanged.



Conditional statements require logical operations. Logical operations should result in a **Boolean value**:

- **TRUE**, i.e. 'yes'
- **FALSE**, i.e. 'no'

Programming languages have different rules for how they interpret non-Boolean values

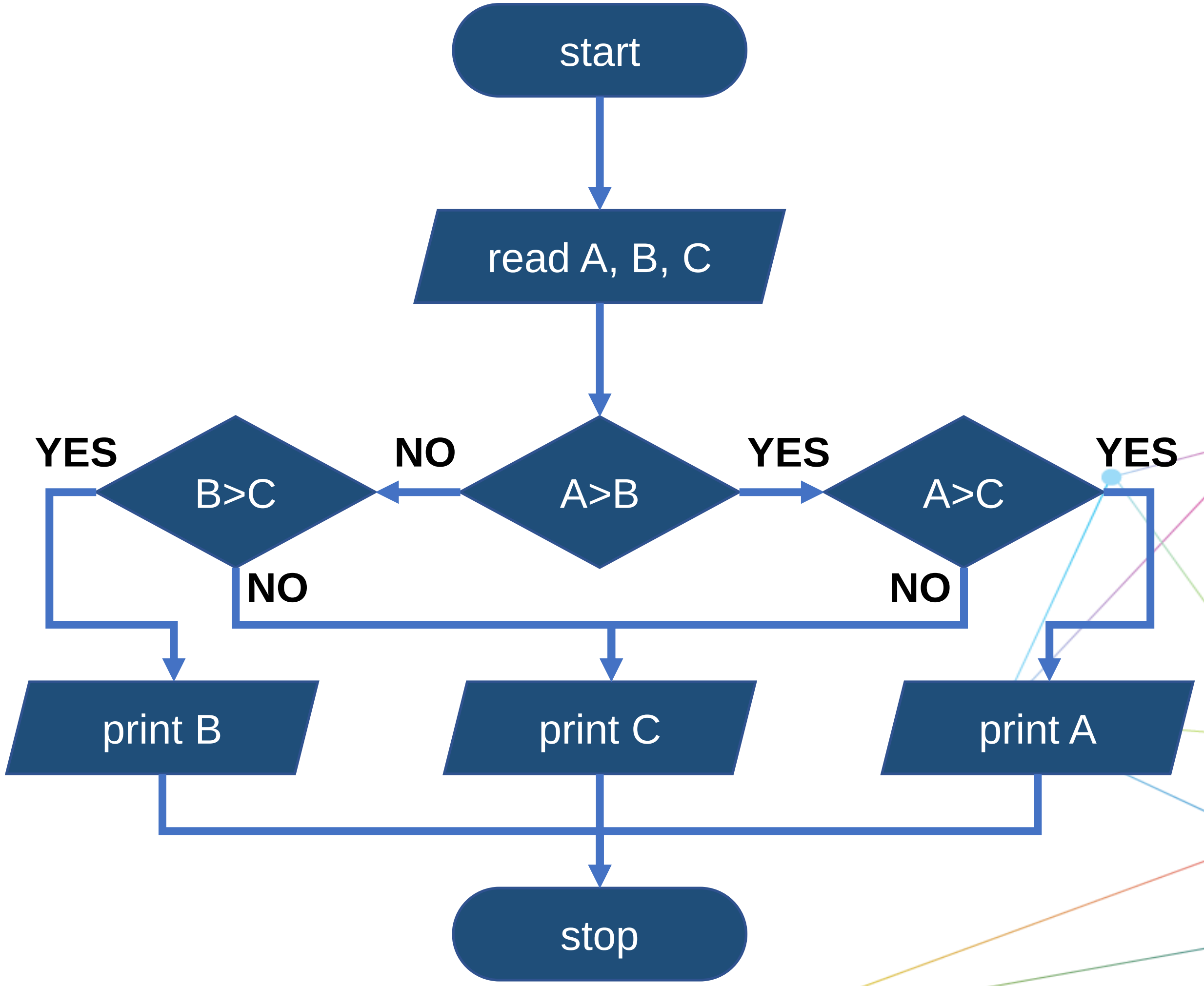
- e.g. R and Python interpret 0 as FALSE and any non-zero number as TRUE

Relational Operators (for numerical values)	
x == y	Is x equal to y?
x ≠ y (x != y)	Is x not equal to y?
x < y	Is x less than y?
x > y	Is x greater than y?
x ≤ y (x <= y)	Is x less than or equal to y?
x ≥ y (x >= y)	Is x greater than or equal to y?

Exercise 3

Design an algorithm as a flowchart that takes three numbers as input and prints the largest of them.

Design an algorithm as a flowchart that takes three numbers as input and prints the largest of them.



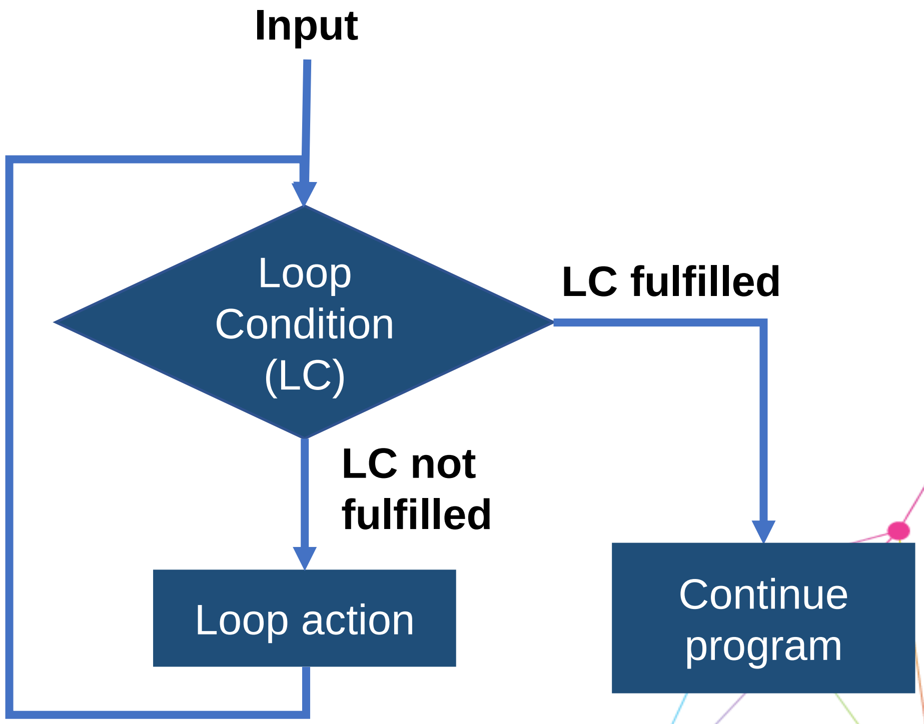
Two general types of loops exist:

count-controlled loops, i.e. loops that execute a pre-defined number of times

- e.g. a loop that adds the first 10 positive numbers.

dynamically terminated loops, i.e. loops that only terminate once a condition, evaluated within the loop, is met.

- e.g. a loop that continues until a



Definition

Loops are sequences of instructions that are executed repeatedly.

Exercise 4

Design an algorithm as a flowchart that adds the squares of the first 10 numbers.

Initialize two variables, placeholders for values.

- ***sum*** is the running sum of all numbers
- ***count*** keeps track of how many numbers have already been added to ***sum***
- If ***count*** is greater than 10, we've added the squares of the first 10 numbers to ***sum*** (since ***count*** started at 1). Output ***sum*** and end the algorithm.
- If ***count*** is not greater than 10, add its square to ***sum***, increase ***count*** by 1, and loop back to the conditional statement.

What happens in the loop?

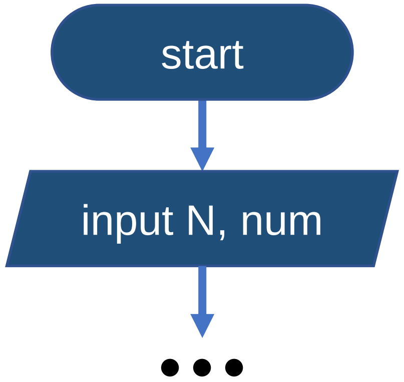
- **Iteration 1 (sum = 0; count = 1 → stay in loop)**
  - increment = count^2 = 1^2 = 1
  - sum = sum + increment = 0 + 1 = 1
  - count = count + 1 = 1 + 1 = 2
- **Iteration 2 (sum = 1; count = 2 → stay in loop)**
  - increment = count^2 = 2^2 = 4
  - sum = sum + increment = 1 + 4 = 5
  - count = count + 1 = 2 + 1 = 3

  …
- **Iteration 10 (sum = 285; count = 10 → stay in loop)**
  - increment = count^2 = 10^2 = 100
  - sum = sum + increment = 285 + 100 = 385
  - count = count + 1 = 10 + 1 = 11
- **Iteration 11 (sum = 385; count = 11 → <u>leave loop!</u>)**
- print ‘385’

Exercise 5

Design an algorithm as a flowchart that lets a user enter ‘N’ numbers and prints out the largest of the numbers. The algorithm should take ‘N’ as an input.

Hint:



The Center of Applied Data Science

Programming Elements
Loops

Exercise 6

Design an algorithm as a flowchart that reads user-entered numbers from the input until the user enters the number 0. Then, calculate and return the average value of all previously entered numbers (excluding the 0).

Programming Elements
Loops

Exercise 7

Design an algorithm as a flowchart that prints the multiplication table for numbers from 1 to 12.

X	1	2	3	4	5	6	7	8	9	10	11	12
1	1	2	3	4	5	6	7	8	9	10	11	12
2	2	4	6	8	10	12	14	16	18	20	22	24
3	3	6	9	12	15	18	21	24	27	30	33	36
4	4	8	12	16	20	24	28	32	36	40	44	48
5	5	10	15	20	25	30	35	40	45	50	55	60
6	6	12	18	24	30	36	42	48	54	60	66	72
7	7	14	21	28	35	42	49	56	63	70	77	84
8	8	16	24	32	40	48	56	64	72	80	88	96
9	9	18	27	36	45	54	63	72	81	90	99	108
10	10	20	30	40	50	60	70	80	90	100	110	120
11	11	22	33	44	55	66	77	88	99	110	121	132
12	12	24	36	48	60	72	84	96	108	120	132	144

Programming Elements

Loops

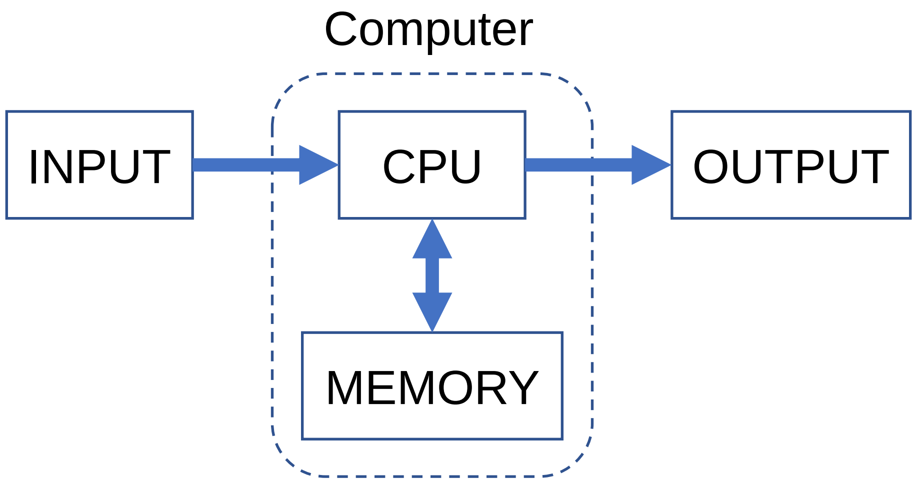
1. What is an Algorithm?
2. Describing Algorithms with Flowcharts
3. **Introduction to Programming**

Computers can be instructed to execute algorithms

Computer programs are implementations of algorithms

Programming languages are the <u>formal languages</u> of computers

- e.g. C, Java, Python, R



Programming elements, i.e. variables, conditional statements, and loops, can be represented by nearly all programming languages, albeit with slight differences

Python

```
counter = 0
while counter < 5:
  counter = counter + 1
  if counter != 3:
    print(counter)
```

Java

```
int counter = 0;
while(counter < 5) {
  counter = counter + 1;
  if(counter != 3) {
    System.out.println(counter);
}}
```

R

```
counter <- 0
while(counter < 5) {
  counter <- counter + 1;
  if(counter != 3) {
    print(counter)
}}
```

➔ Identical output!

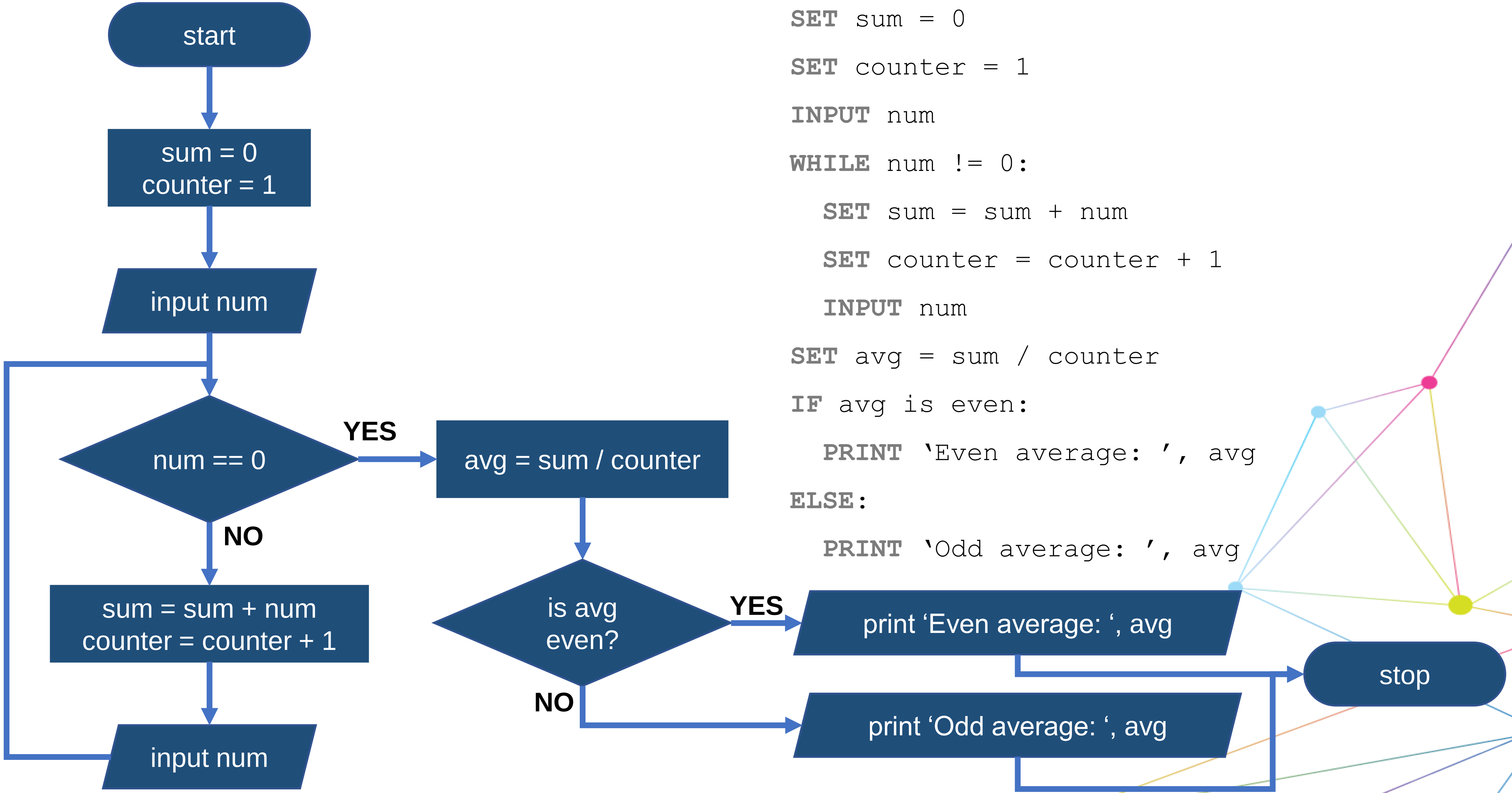
- **Pseudocode** is an informal, human-readable, descriptive language meant to resemble common programming languages in its structure

- There is no formal syntax – many variations exist that resemble different programming languages

The Center of Applied Data Science

Introduction to Programming
Pseudocode Example

```
ALGORITHM "Averages":
  SET sum = 0
  SET counter = 1
  INPUT num
  WHILE num != 0:
    SET sum = sum + num
    SET counter = counter + 1
    INPUT num
  SET avg = sum / counter
  IF avg is even:
    PRINT 'Even average: ', avg
  ELSE:
    PRINT 'Odd average: ', avg
```



Exercise 8

Design an algorithm to play the ‘Fizz Buzz’ game. The algorithm should count from 1 to 100 and print out the numbers. However, if a number is divisible by 3, the algorithm should print ‘Fizz’ instead of the number. If a number is divisible by 5, it should print ’Buzz’ instead of the number. If a number is divisible by both 3 and 5, the algorithm should print ‘Fizz Buzz’ instead of the number. The output should therefore look as follows:

1, 2, Fizz, 4, Buzz, Fizz, 7, 8, Fizz, Buzz, 11, Fizz, 13, 14, Fizz Buzz, 16

Begin by designing a flowchart for this algorithm and then try to translate it into pseudo code.

The Center of Applied Data Science

Introduction to Programming
Pseudocode Example

Introduction to Programming

Pseudocode Example

Summary

- **Algorithms** are detailed instructions, written in a formal language, that describe the solution to a problem
- Common programming elements of algorithms are:
  - **Variables** to easily store, reference, and modify values
  - **Conditional statements** to make decisions in the algorithm flow
  - **Loops** to repeatedly execute certain steps
- Algorithms can be concisely and visually represented as **flowcharts**
- Algorithms can be represented with **pseudocode** to resemble an implementation as a computer program → more intuitive to design a flowchart first and then translate it into pseudocode.

The Center of Applied Data Science

Thank You