

ARIZONA SCIENCE LAB

How Computers Think!

Help Solve Problems!



**Institute Of Electrical And Electronic Engineers, Phoenix Section
Teacher In Service Program / Engineers In The Classroom (TISP/EIC)
Arizona Science Lab www.azsciencelab.org**

Our Sponsors

The AZ Science Lab is supported through very generous donations from corporations, non-profit organizations, and individuals, including:



What Are We Going To Talk About?

- What does a computer really do?
- How do we solve problems?
- Algorithms and Data – what are they?
- Counting and Sorting algorithms.
- How a computer helps us!

Example Uses of Computers

- Number Computation:
 $6+28+92+71 = ??$
- Data Analysis:
Find all students named “John”
- Graphics, Photography:
Increase the color RED in a photo; Crop a photograph
- Machine (Artificial) Intelligence:
Play a game of checkers; Recognize speech or handwriting or objects in pictures.

What Does a Computer Do?


People Solve Problems:

- **Computers Don't Do Anything By Themselves!**
- Computers simply execute “instructions”
- Computers are given a “Program”, a list of instructions to execute.
- Each instruction or command is itself very simple.
- Computers execute these instructions very quickly and accurately!
- After executing many many instructions, the computer assists humans in solving the problem!

Solving a Problem (1)

- Start with a problem!
- Make sure it is well defined:
 - Understand the problem space: the details of the problem and the situation.
 - What facts are required to solve the problem?
 - Are the facts & assumptions: accurate, timely, complete?
 - What are the inputs?
 - What are the outputs?

Solving a Problem (2)

- Define the “algorithm” or method – the steps, the recipe, to solve the problem, in the finest detail (step by step).
- If a computer will help with the problem, then tell **all of the above** to the computer 
 - Translate the algorithm into a “program”, a list of instructions the computer understands!
 - Define all of the data (facts, inputs).
 - Execute the program and get the outputs!

Algorithms – An Example

Say my friend is arriving at the airport and needs to get to my house.

Here are 4 ways he can do it:

- **The Taxi algorithm:**
 - Go to taxi stand, get in a taxi, give driver my address.
- **The call-me algorithm:**
 - When plane arrives call my cellphone, meet me outside baggage claim.
- **The rent-a-car algorithm:**
 - Take shuttle to rental center, rent a car, follow directions and drive to my house.
- **The bus algorithm:**
 - Outside baggage claim catch bus #70, transfer to bus #14 at Main St., get off on Elm St., walk two blocks to my house.

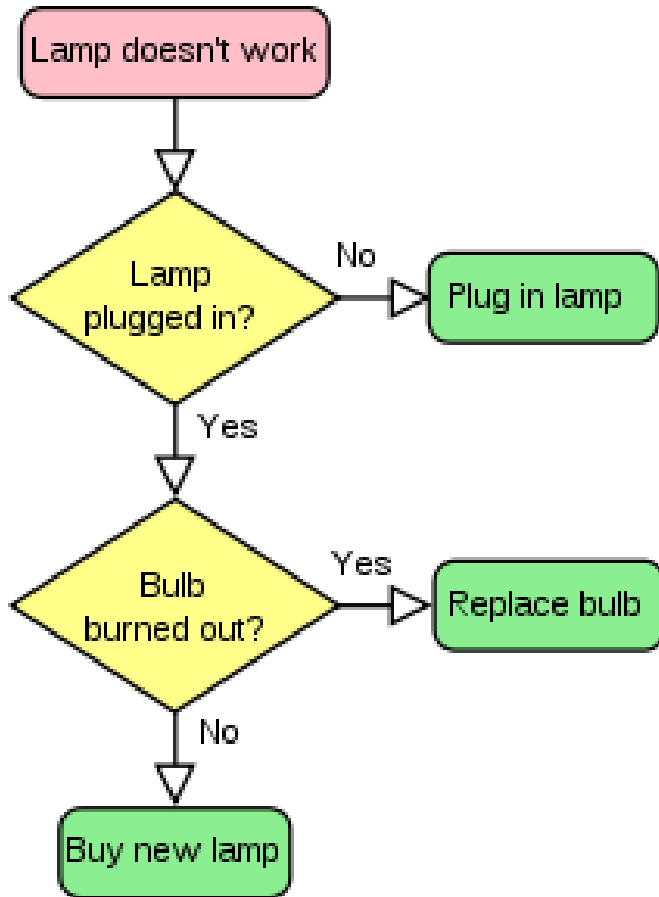
Algorithms – An Example (con't)

- All 4 algorithms accomplish the same goal.
- Each does it in a completely different way.
- Each has a different cost and travel time.
- They each trade off: cost and time.

You have to know the strengths and weaknesses of each algorithm you use!

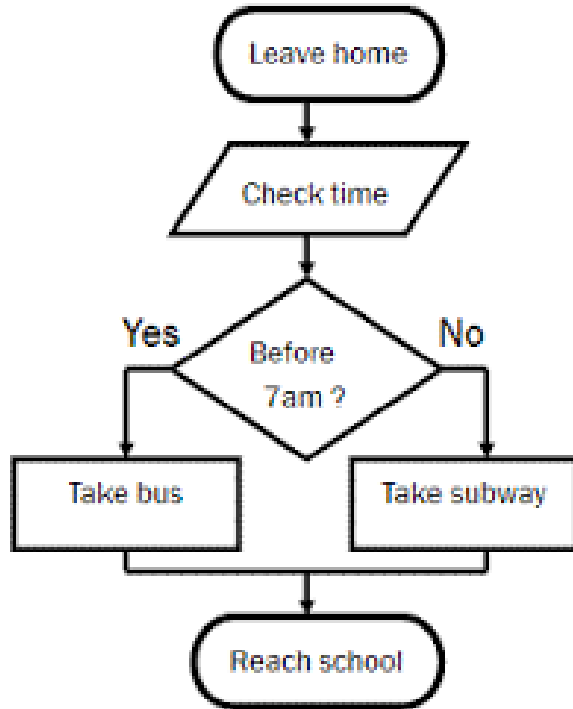
For computer executed algorithms: tradeoff is usually time & space (memory).

A Simple Algorithm



- Problem space?
- Facts?
- Algorithm:
 1. Is lamp plugged in? (Test)
 2. No, plug it in. (Action)
 3. Yes, next step
 4. Is bulb burned out? (Test)
 5. Yes, replace bulb (Action)
 6. No, next step
 7. Buy new lamp!

Another Algorithm



Algorithms have:

- Tests
- Actions
- Jumps

Computers are very good at “testing” information:

Is: $A > B$, $A < B$, $A = B$??
then “Do X or Y”

Computers are very good at “actions”:

$A + B$, $B \leftarrow B + 1$, $B \leftarrow A$
Turn on switch
Sound alarm

Computers can “jump” around in the algorithm based on test conditions.

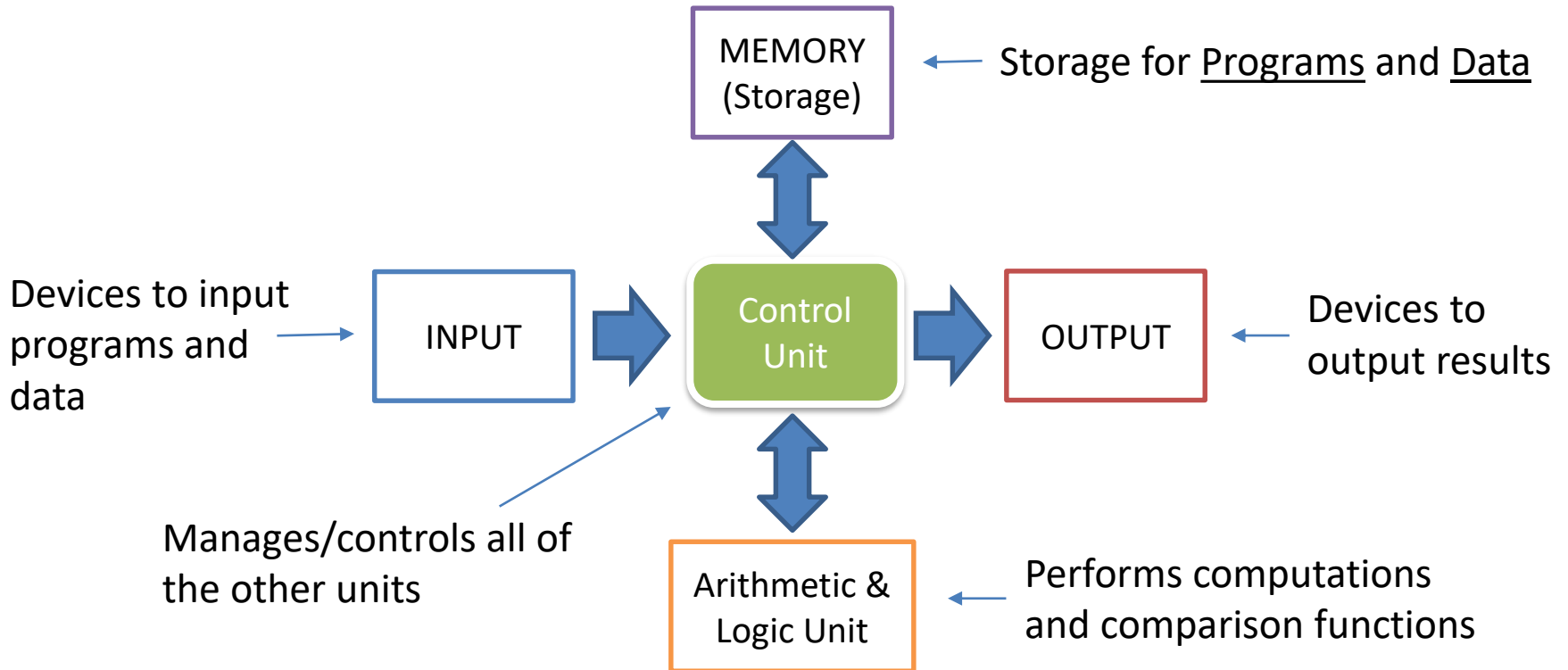
A Problem For You To Solve

- How many days is it from now till the next Labor Day?
- Define: needed facts, input, algorithm.
- Hint: Labor Day is always the first Monday in September.

How Many Days Till Next Labor Day?

- Facts:
 - How many days in each month of the year? Need a calendar.
- Inputs:
 - Today's date: month, day, year.
- Labor Day Algorithm:
 - First, calculate when Labor Day date is this year: first Monday in September.
 - Is the current date before or after Labor Day of this year?
 - If before: compute the number of days between today and Labor Day.
 - If after: Compute days(1) between today and Dec. 31, then calculate when Labor Day date is next year, compute days from Jan. 1 next year to Labor Day date next year, add to (1).
 - When computing days, check if leap year (Feb. 29) is in the calculation and add appropriately.
 - Print out or display the number of days.

The Computer



- Computers manipulate **data** (information) by executing **instructions** (sequence of program steps) stored in the memory.
- The Control Unit and Arithmetic Unit together are usually called the: **CPU (Central Processing Unit)**.

Algorithms, Data Structures, & Programs

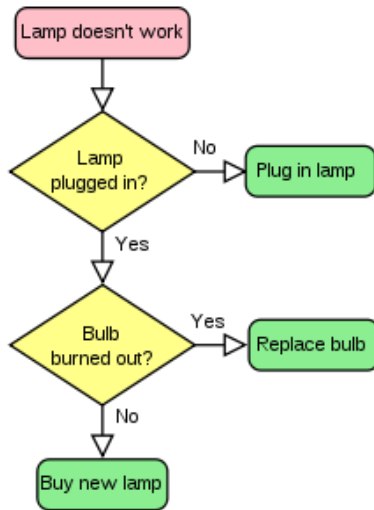
- **Computer Algorithms** define how information is manipulated.
- **Programs** tell the Computer what and how to process: information and the steps in an algorithm.
- Programs are written in a **Language** (code) the computer understands.
- Algorithms are like recipes and are not language specific.
- Good algorithms are:
 - Accurate, Flexible, Fast!
- Algorithms process data stored in an organized structure.
- **Data Structures** are as important as algorithms themselves.
- **Algorithms must be fast!!!**

Computer Instructions

- Computer instructions are the steps in the program the computer executes.
- They are the steps in the algorithm: tests, actions, and jumps (move to different part of algorithm).
- Programming is the translation step into actual code.
- We can write out these steps in an informal language before translating them into “Code”.
- This informal language is “pseudocode”. It can be:
Graphical, List of steps(English like), Informal program code

Pseudocode

Describing an algorithm:



Graphical:

Triangles for “tests”

Boxes for “actions”

Arrows for “jumps”

Text: English like

1. No, plug it in
2. Yes, next step 3
3. Is bulb burned out?
4. No, next step 6
5. Yes, replace bulb, done!
6. Buy new lamp!

Program like:

```
procedure lamp ( A : state of lamp, B: state of bulb)
  if A is false, then
    plug in lamp, A=true; else
  if B is false, then
    replace bulb, B=true; else
  buy new lamp,
endif;
endif;
end procedure;
```

Data or Variables

- The memory of the computer stores both Programs and Data (Variables).
- Each item of data stored in the memory is located by referencing the name of the item.
- Computers can store very large amounts of data.
- Programs and data are usually stored on a disk drive connected to the memory and copied into the memory when executed.

Counting Algorithm Exercise

Algorithm 1:

- Everyone stands up.
- First person says “1” and sits down.
- Next person adds “1” to what previous person said and says that number and sits down.
- Continue till only one person is standing, that person says that number.

Algorithm 2:

- Everyone stands up and thinks “1”.
- Pair off with someone near you, add your numbers together, person with smaller total sits down (or either one of you if equal total).
- Continue till only one person standing, that person says that number.

NOW LET'S COMPARE THESE ALGORITHMS!

Compare Counting Algorithm

- Algorithm 1:

If we have N people – how many steps will be executed in this algorithm?

- Algorithm 2:

If we have N people – how many steps will be executed in this algorithm?

Counting Algorithms



Phonebook Search Algorithm Exercise

We are trying to find the phone number of “Mike Smith” given a very large phone book list (in alphabetized order):

Name1, Number1

Name2, Number2

...

Let’s create an algorithm to do this:

Algorithm 1:

Compare first Name to “Mike Smith”

If not equal, go down to next Name and compare.

Keep doing this till you find “Mike Smith”

When name equals “Mike Smith”, set Number to his phone number.

Exercise - NOW – You create a faster algorithm!

A Faster Phonebook Search

1. Start in the middle of the list.
2. We now have a front half and back half of list.
3. Compare first Name in back half, if Name is “less than” Mike Smith. Less than means: $A < B$, etc.
4. Then the back half of the list is our new list.
5. Go back to step 1 using new shorter list, else:
6. If Name is “greater than” Mike Smith.
7. Then the front half of the list is our new list.
8. Go back to step 1, using new shorter list.
9. If Name equals Mike Smith, we are done!

Sorting Algorithms

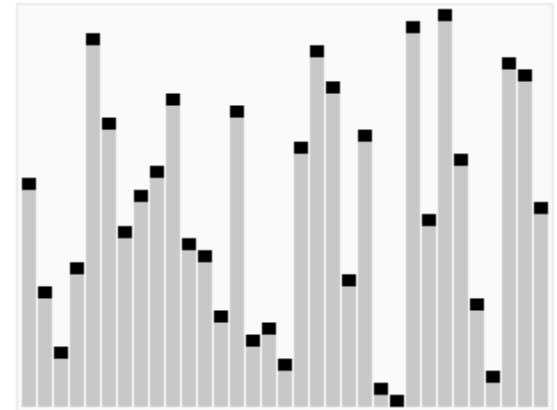
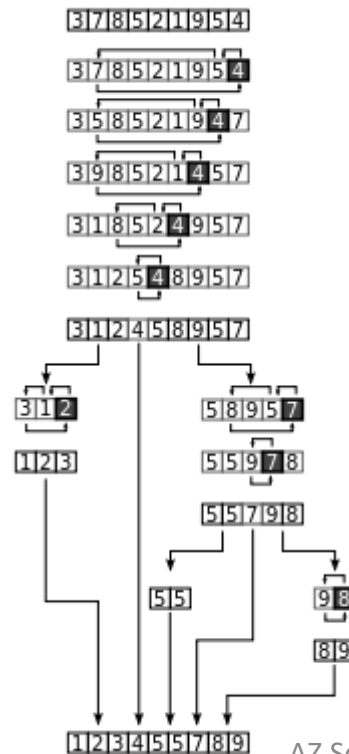
Sorting a list is a common example of a computer algorithm:
For a very short list a bin sort may be best,
But, for a very long list quicksort may be best.

You have to know the strengths and weaknesses of each algorithm you use!

Catalogued Algorithms

- Over time many excellent algorithms have been created and catalogued in books.
- Many algorithms are not specific to a particular business type, but are very general.
- Examples: sorting numbers and counting algorithms.

Example:
quicksort
algorithm



A Sorting Algorithm Exercise

- We have a list of 10 numbers stored in the computer at location address: LIST.
- We want to sort (reorder) the items in the list so that the smallest is the first entry and the largest the last entry. It may be that two items in the list have the same value.
- Write down the algorithm you have created to sort the list.

Sort Program Solution

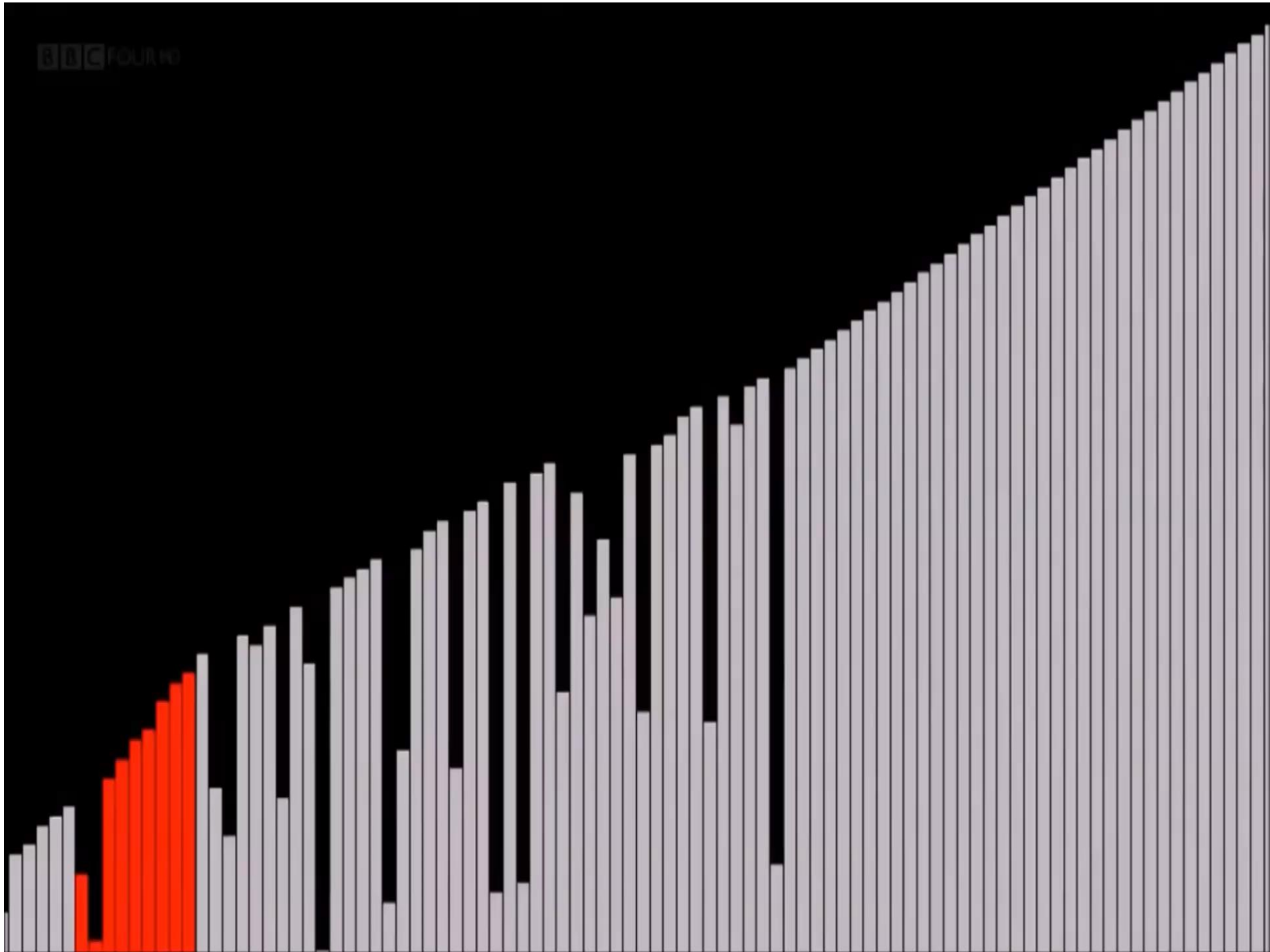
- There are many algorithms that easily solve the sorting problem. A simple one for short lists like this one is the bubble sort.
- In this algorithm, the program repeatedly steps through the list to be sorted, compares each pair of adjacent items and exchanges (swaps) them if they are in the wrong order. The pass through the list is repeated until no swaps are needed, which indicates that the list is sorted. In this way, the smallest items “bubble up” to the top of the list.
- Example of a bubble sort:

6 5 3 1 8 7 2 4

Bubble Sort Program Solution

```
procedure bubbleSort ( A : list of sortable items )
  n = length(A)
  repeat
    swapped = false
    for i = 1 to n-1 inclusive do
      if A[i-1] > A[i] then
        swap(A[i-1], A[i])
        swapped = true
      end if
    end for
    n = n - 1
  until not swapped
end procedure
```

Algorithm Review



The Future of Computers

- **Smaller, faster, cheaper!**
 - Moore's Law - the number of [transistors](#) in a dense [integrated circuit](#) doubles approximately every two years! (Gordon Moore, Intel, 1965)
- **Internet of Things (IOT)**
 - The interconnection (Internetworking) of small sensors in many everyday objects: heart monitors, refrigerator temperature sensors, ... will eventually number in the billions!
- **Machine Intelligence (AI)**
 - Computer programs that “learn” or increase their knowledge base over time and make better decisions.
- **Cybersecurity**
 - The protection of computer information from theft or disruption of services.

What did we learn today?

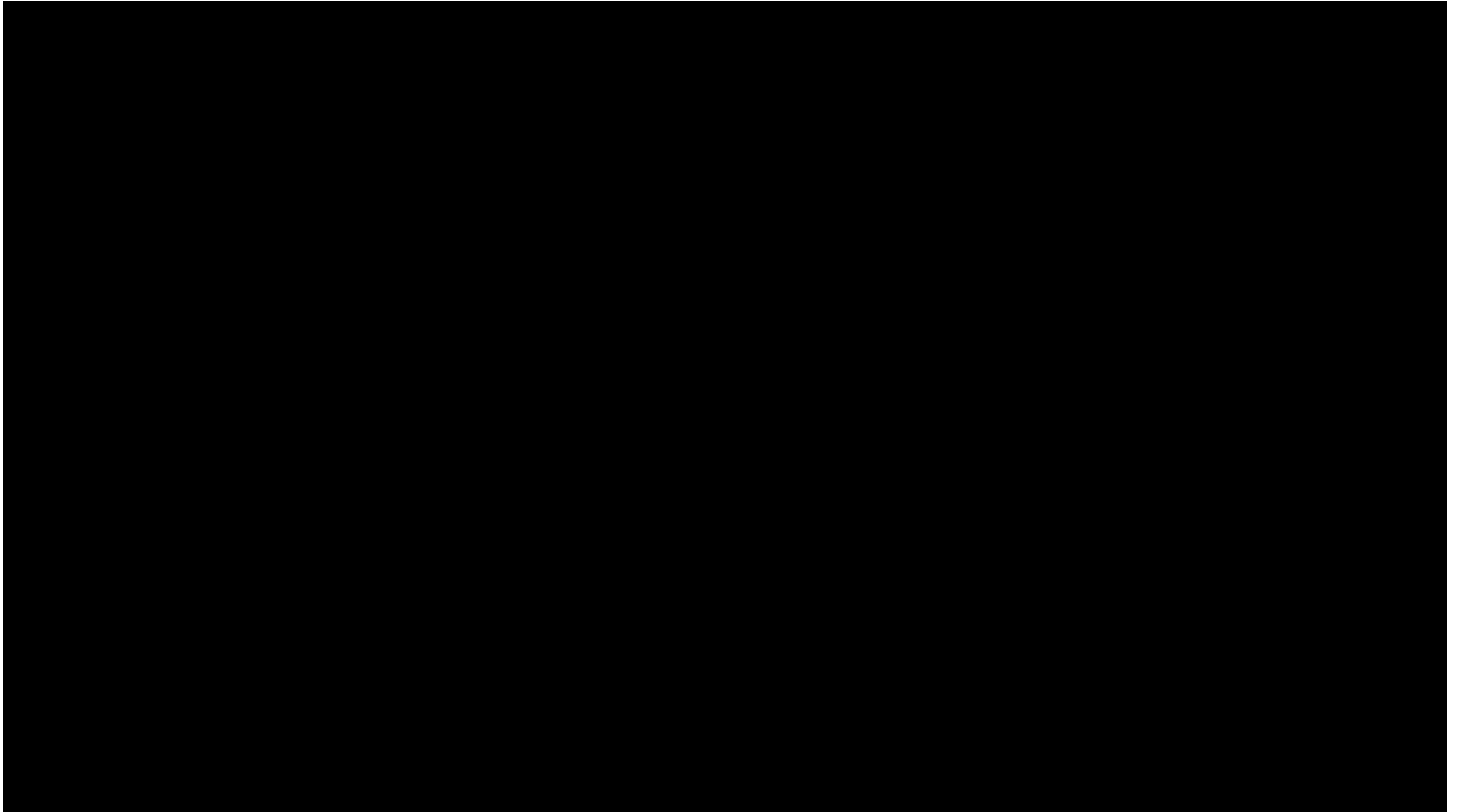
We explored how computers help us in solving problems:

- What computers do and don't do
- Steps in solving a problem
- Using the computer to help in solving problems
- Describing algorithms and data structures
- Counting, searching, and sorting algorithms
- The future

Careers in STEM

- You must find your passion
- You can have a very rewarding career in science and/or engineering:
 - Financial, satisfaction, enjoyment
- Need learning and training (education)
- Maybe you will even become another great scientist or engineer!

Careers in STEM



Have Fun Today?

Check out our website: www.azsciencelab.org
click on the “For Students” tab!

Thanks for coming and exploring with us
the world of how computers ~~think!~~
help solve problems!!!