# Information Technology for Digital Humanities

Lecture 2

Mario Verdicchio

Università degli Studi di Bergamo

Academic Year 2023-2024

# Lecture 2 (September 27 2023)

- Fundamental concepts: technology

# Information **Technology** for Digital Humanities

Lecture 2

Mario Verdicchio

Università degli Studi di Bergamo

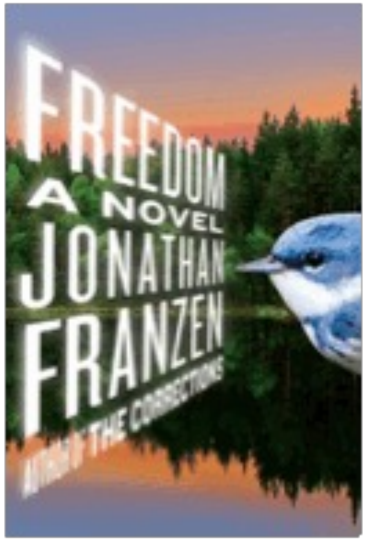
Academic Year 2023-2024

# What is “technology”?

- Exercise: provide one or more definitions of “technology”.
- Fabio: «domain of human activity aimed at action on matter with the purpose of solving problems or achieving goals that involves the development of tools and methodologies to do so.»
- Darja: «a tool that is intertwined with innovation; usually used to improve human condition; with both positive and negative effects.»


Similarity:
facilitates practical purposes
Difference:
found

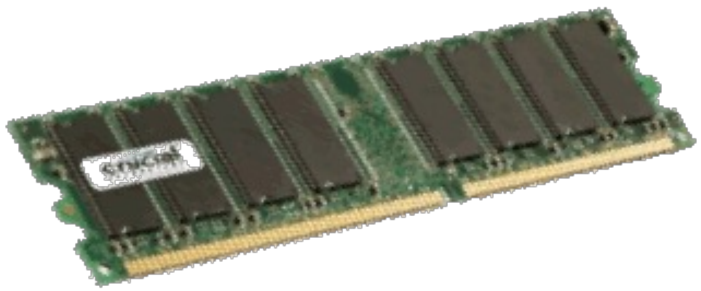

Similarity:
facilitates practical purposes
Difference:
made


FREEDOM
A NOVEL
JONATHAN
FRANZEN
AUTHOR OF THE CORRECTIONS

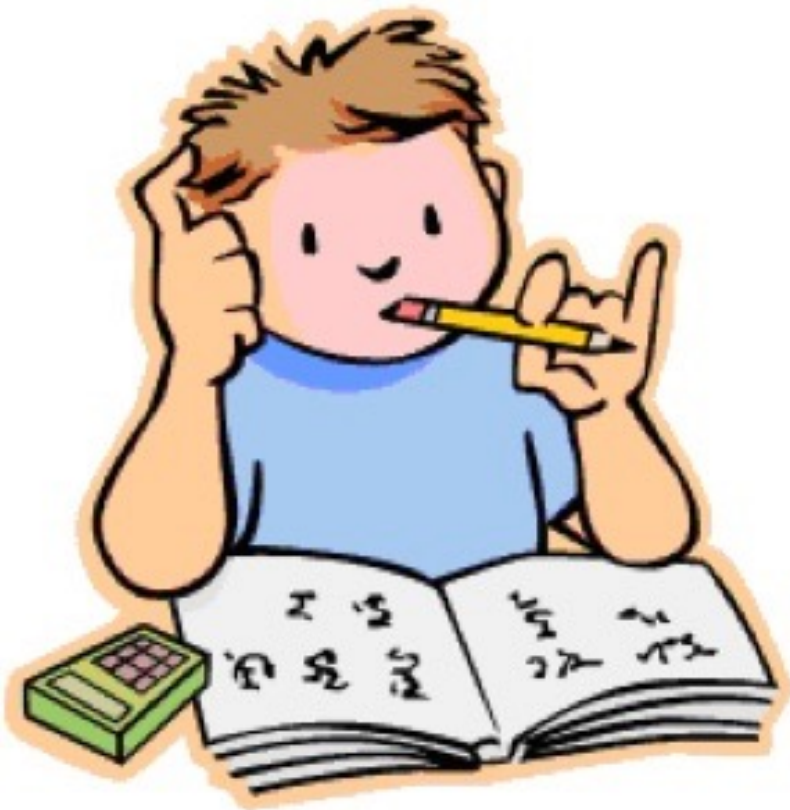

Tuesday
9

We have seen some (important) limitations of a computer: we examined a problem, and showed that a computer is not capable of solving it.
Typical example: understanding the meaning of a word. Yet computers are able to perform many operations, and solve a large number of problems (showing us the pages of a book, connecting with friends on social networks, downloading and listening to music, organizing our agenda, watching films, etc. )
The time has come to delineate the field of work of computers.
Let's examine what all the solutions a computer can offer have in common.

# Software

Software is defined as the complex of commands that make the computer perform operations.
The term contrasts with hardware, which instead designates everything that is material and tangible in a computer.
A computer is a set of electronic devices: a computer is hardware.
However, a computer becomes useful only when it provides a service, and this is only possible when the hardware is controlled by software.
Without software, hardware is a useless agglomeration of plastic, silicon and metals.
Without the hardware, software is just an idea, which could be put into practice, but stays on paper (or in our minds).



Such idea should be the solution to a problem.
It is easy to see that not all problems can be solved using computer software.
The very nature of computers dictates the type of problems they are capable of solving.

**vs.**

If there is a difference in the hardware, there will likely also be a difference in the spectrum of problems that can be solved.
Software is also involved, because the hardware must be controlled appropriately in any case.
Don’t forget that «-matics» in «informatics» comes from «automatic»: the functioning of the hardware-software combination must take place with as little human intervention as possible.
All of these considerations help us define what a computer is capable of performing.

# Basic concepts

**Problem**

**An obstacle, an impediment, a difficulty that we wish to eliminate.**

!

**Solution**

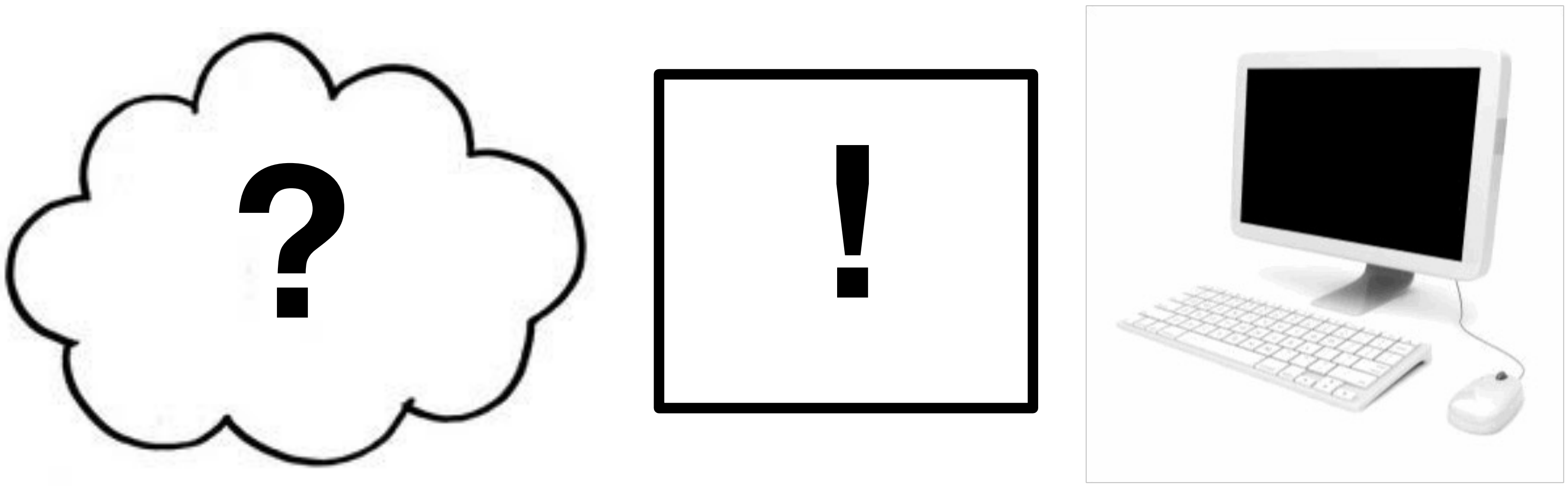
**Elimination of the problem.**

**Computer**

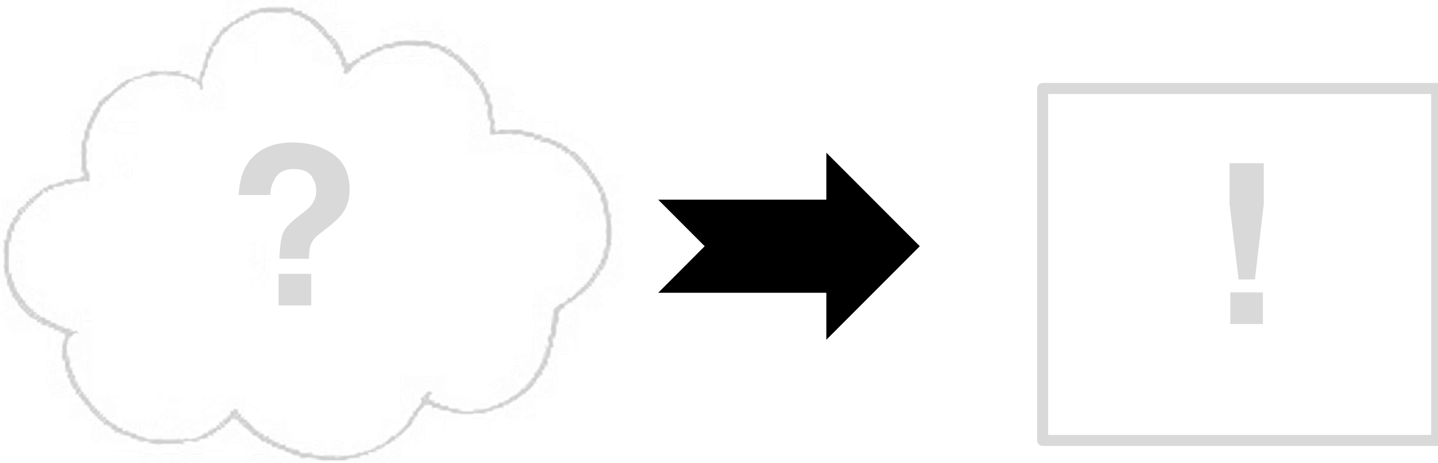
**Automatic symbol processor.**

data

sign




**The problem is said to be solvable when there is a way to reach the solution.**
**If we want the computer to reach the solution, the problem must be expressed in the form of data that the computer can process.**




**Algorithm**

A way to reach the solution that has the following characteristics:

1) it is a finite sequence of well-defined operations;

2) after the execution of each operation, it is clear what the next one is (determinism).

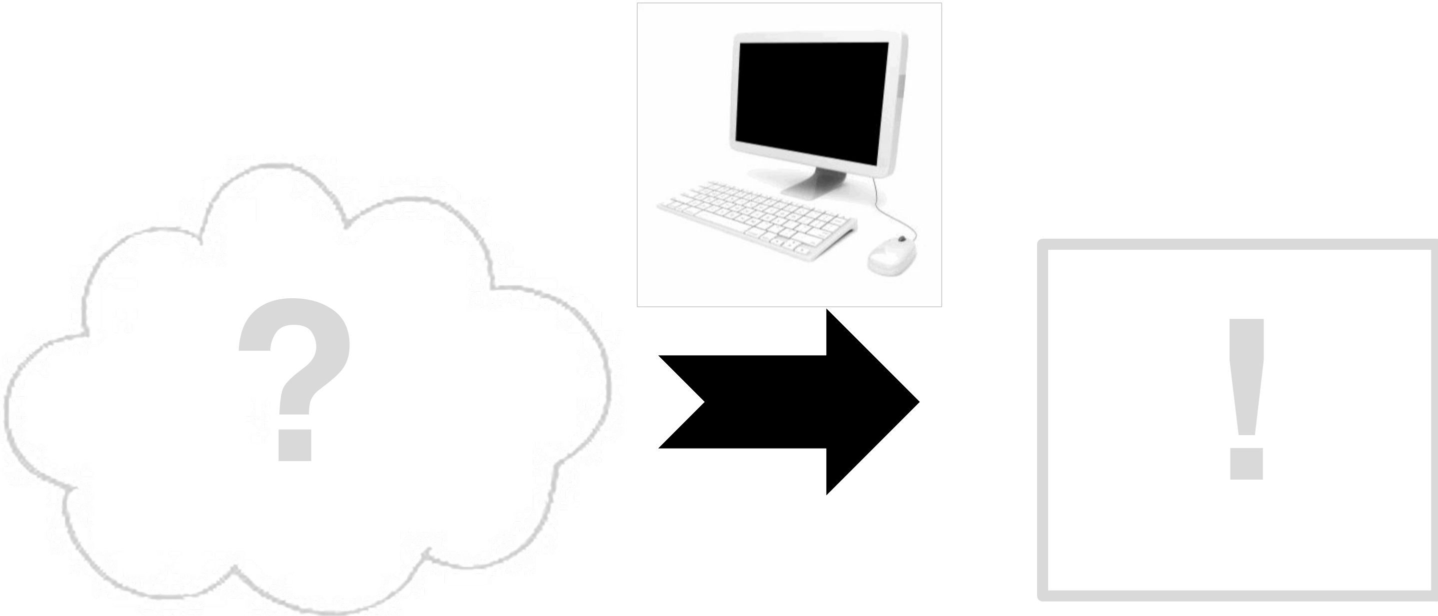

**Muḥammad ibn Mūsā al-Khwārizmī**
**(780 – 850)**

**The term “algorithm” comes from the name of the Persian mathematician al-Kwarizhmi.**

**Among his many merits, the greatest is perhaps that of having imported the decimal number system from India to the West.**
**(The ancient Romans did not have a symbol for zero.)**

**Another term that has its roots in this mathematician's name is "algebra."**

**The figure in the previous slide is found on a Soviet stamp, which in 1980 celebrated the 1200th anniversary of al-Kwarizhmi's birth.**

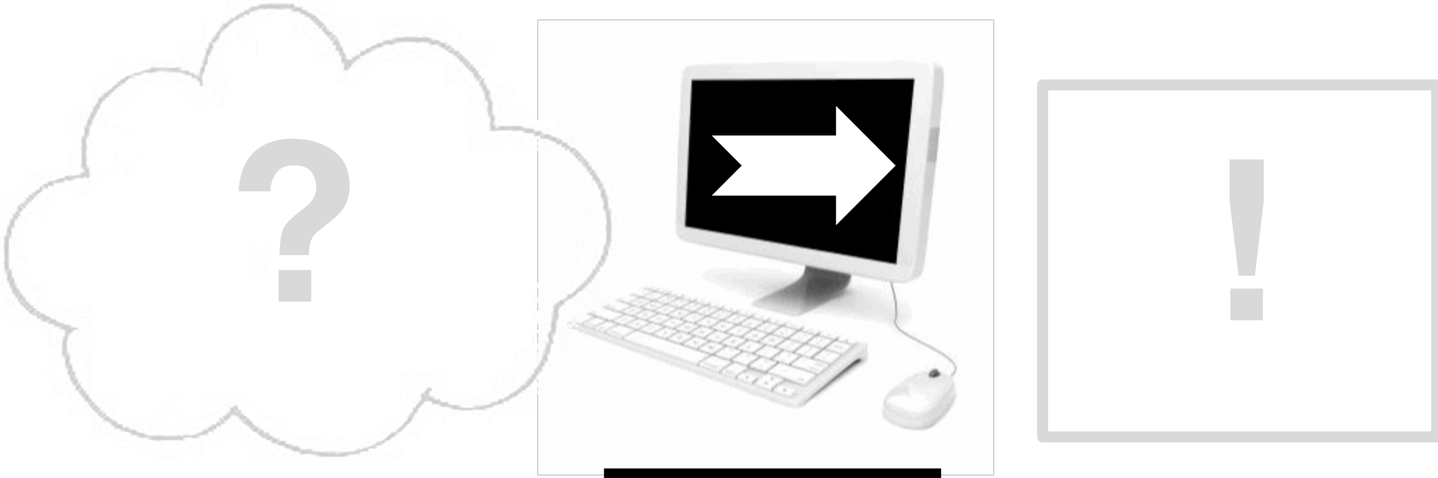



**Program**

**An algorithm written in such a way that it can be executed by a computer:**

**3) all operations consist of symbol processing;**

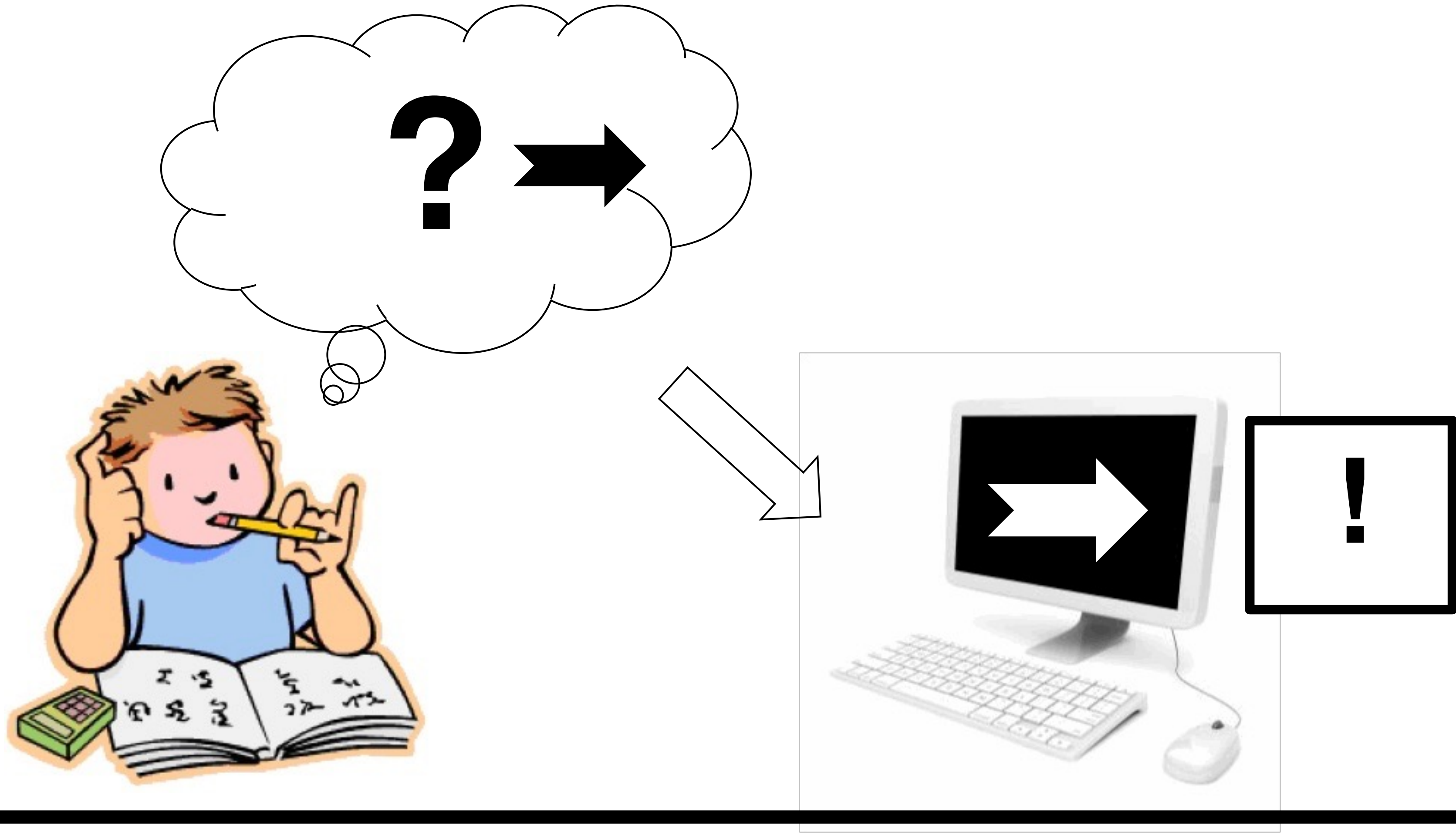
**4) all operations can be performed by the computer.**




**Process**

**A program running on a computer.**

**Roughly speaking, programs usually reside on a computer's hard disk, processes in the RAM.**




**Warning: a computer cannot find a solution on its own. The algorithm must be conceived first, and then transferred into the computer in the form of a program, to be executed and to provide a solution.**

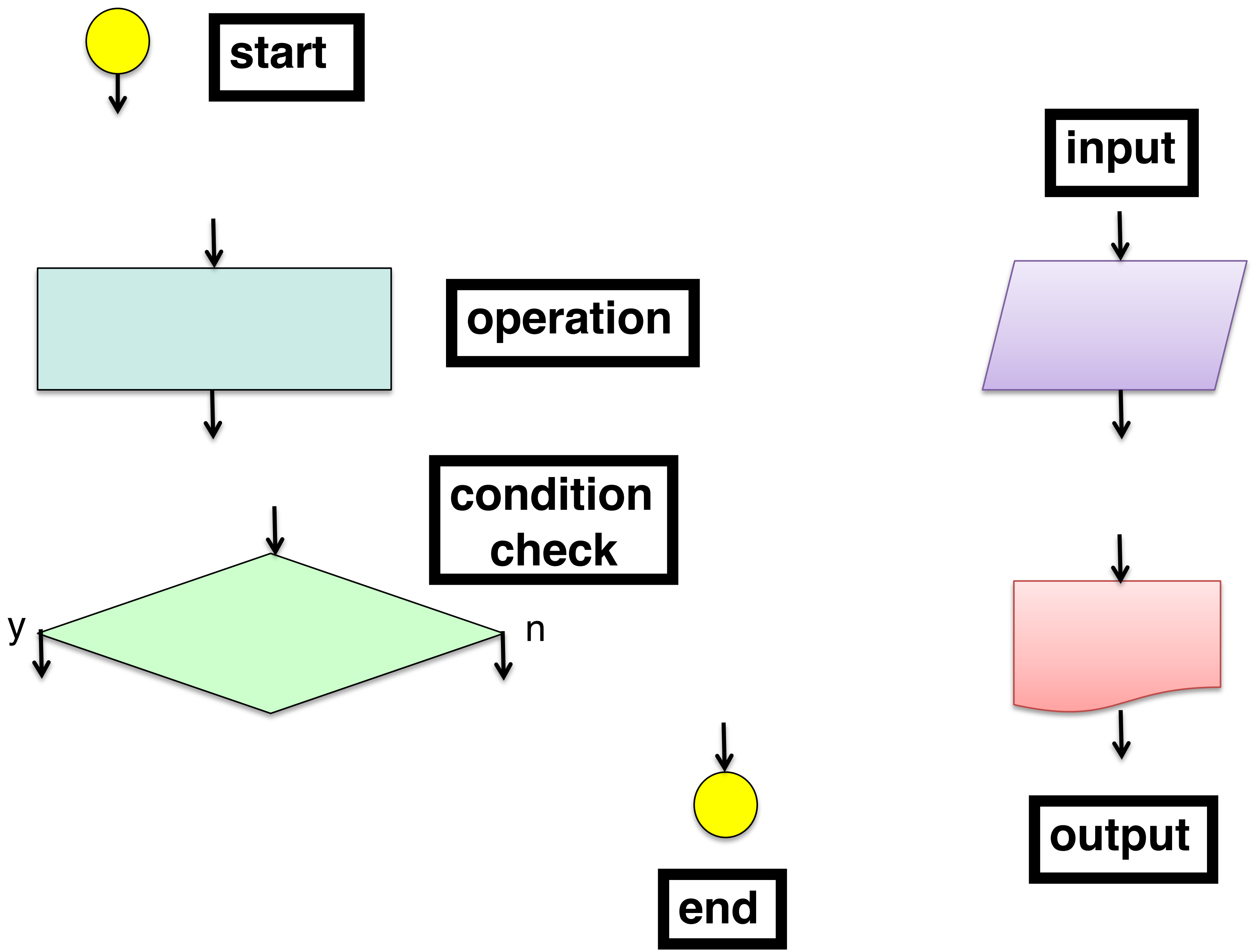
# Flowcharts

# Flowchart

Graphical notation for expressing algorithms.

There is a graphic element (called a block) for each fundamental step in the algorithms.

The blocks are connected to each other by arrows, which symbolize the execution flow, i.e. the sequence of operations that are carried out one after the other.


start
input
operation
condition
check
y
n
end
output

**The arrow entering a block represents the instant at which the operation described by the block is about to be executed.**

**Within each block, the operation to be performed is described in a language of your choice, as long as it is understood by the recipient of the flowchart.**

**The arrow coming out of a block represents the instant at which the operation described by the block has been executed and we move on to the next one.**

**All blocks have one or more input arrows and only one output arrow, except the start block (zero entering, only one exiting), the end block (one or more entering, zero exiting) and the condition check block (one or more incoming, two outgoing: one for each possible outcome of the check).**

**The yellow circles represent the beginning and the end of the algorithm. Determinism implies that there can only be one beginning (no choice), while there can be multiple ending blocks (depending on the path that was followed).**

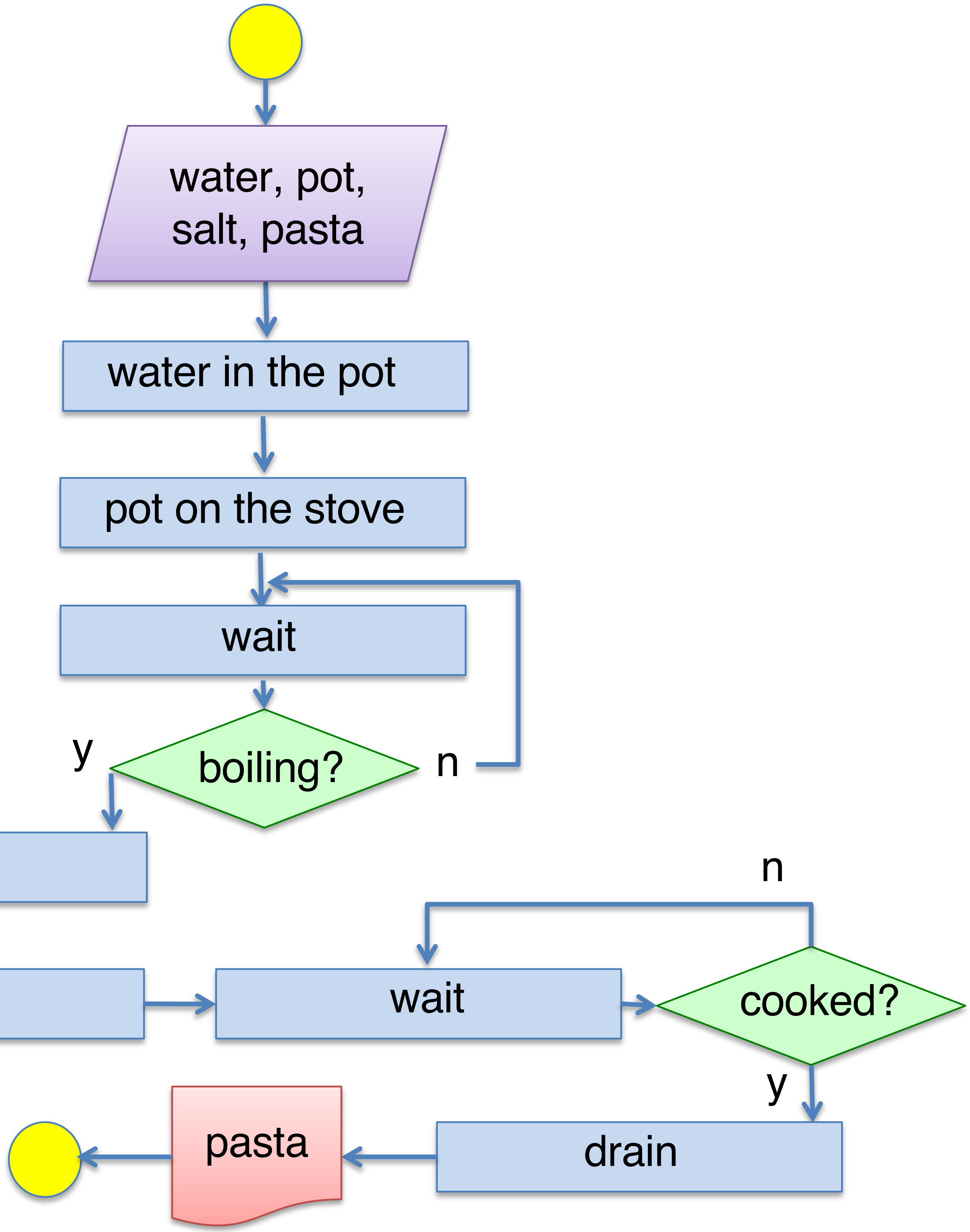
**An operation to be performed is described within a rectangular block, unless it is a special type of operation: input (parallelogram), output (paper slip), or condition check (rhombus).**

**Input: data (or more concrete entities) arrive from outside the context of the algorithm's execution in order to be processed.**

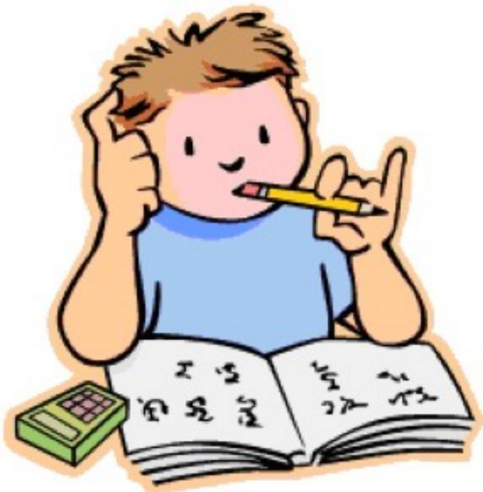
**Output: a result (partial or final) of the algorithm is sent outwards to be used by the users of the system.**

**Condition check: a condition is verified and depending on the result of the verification the algorithm follows one path or another.**

**Algorithm for cooking pasta**



**Such an algorithm cannot become a program.**



**Why?**

**First of all, the algorithm is described in English. To become a program it would have to be rewritten in a language understandable by a computer.**

**Such translation is very difficult to achieve, due to the very approximate descriptions of the various operations, which only a human being can manage.**

**For example: where do you get the water? On which stove should the pot be placed? How should the pot be placed on the stove?**

**Furthermore, the algorithm cannot be expressed in terms of symbol processing: we are dealing with real water and real pots, and a computer does not have the hardware to manipulate such objects.**

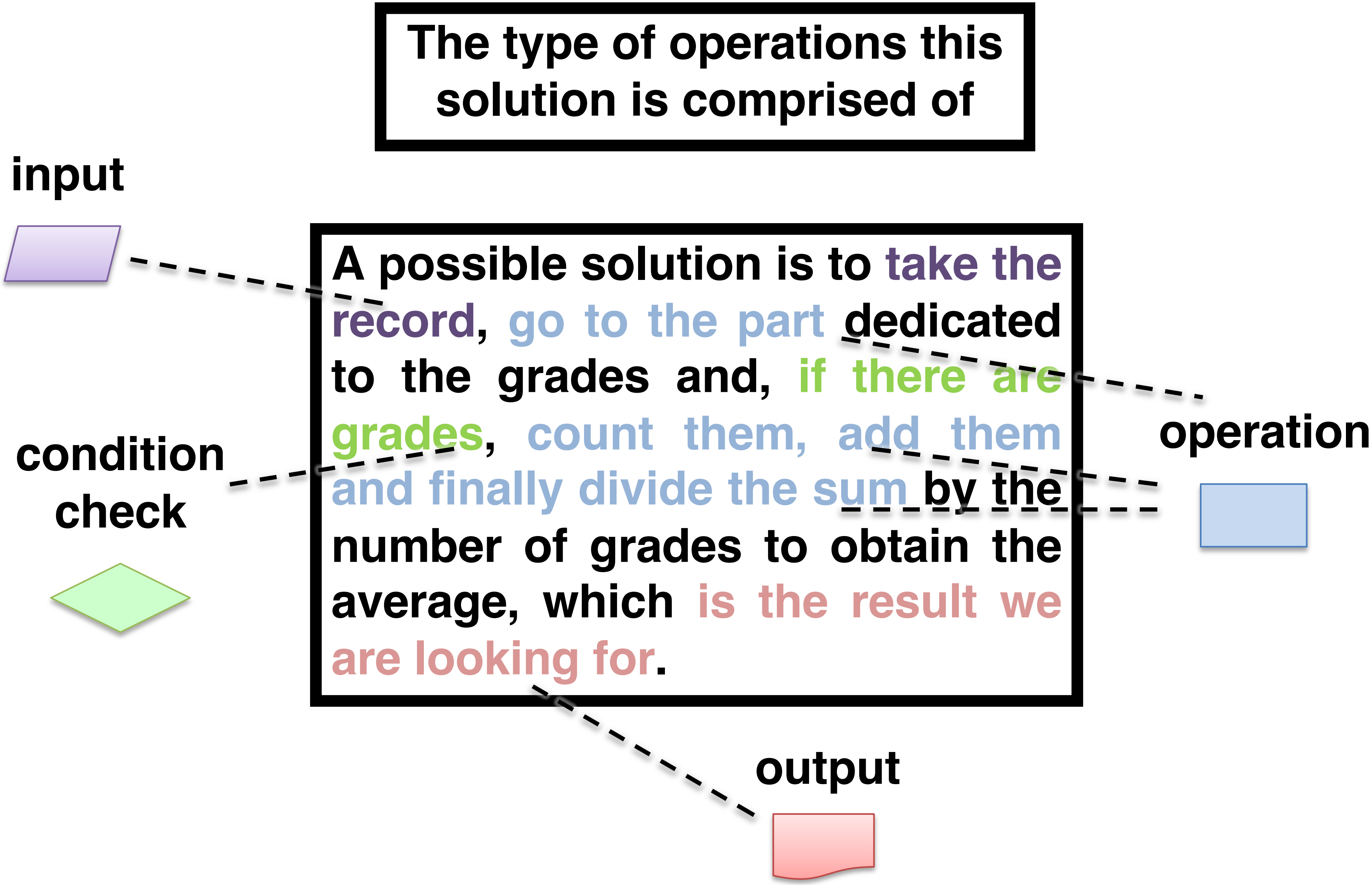
**Perhaps a robot with cameras, sensors, and adequate mechanical arms could solve the pasta problem, but you still need software to control it.**

## Exercise

**Draw the flowchart of the algorithm for calculating the average of the exam grades from a student's record.**

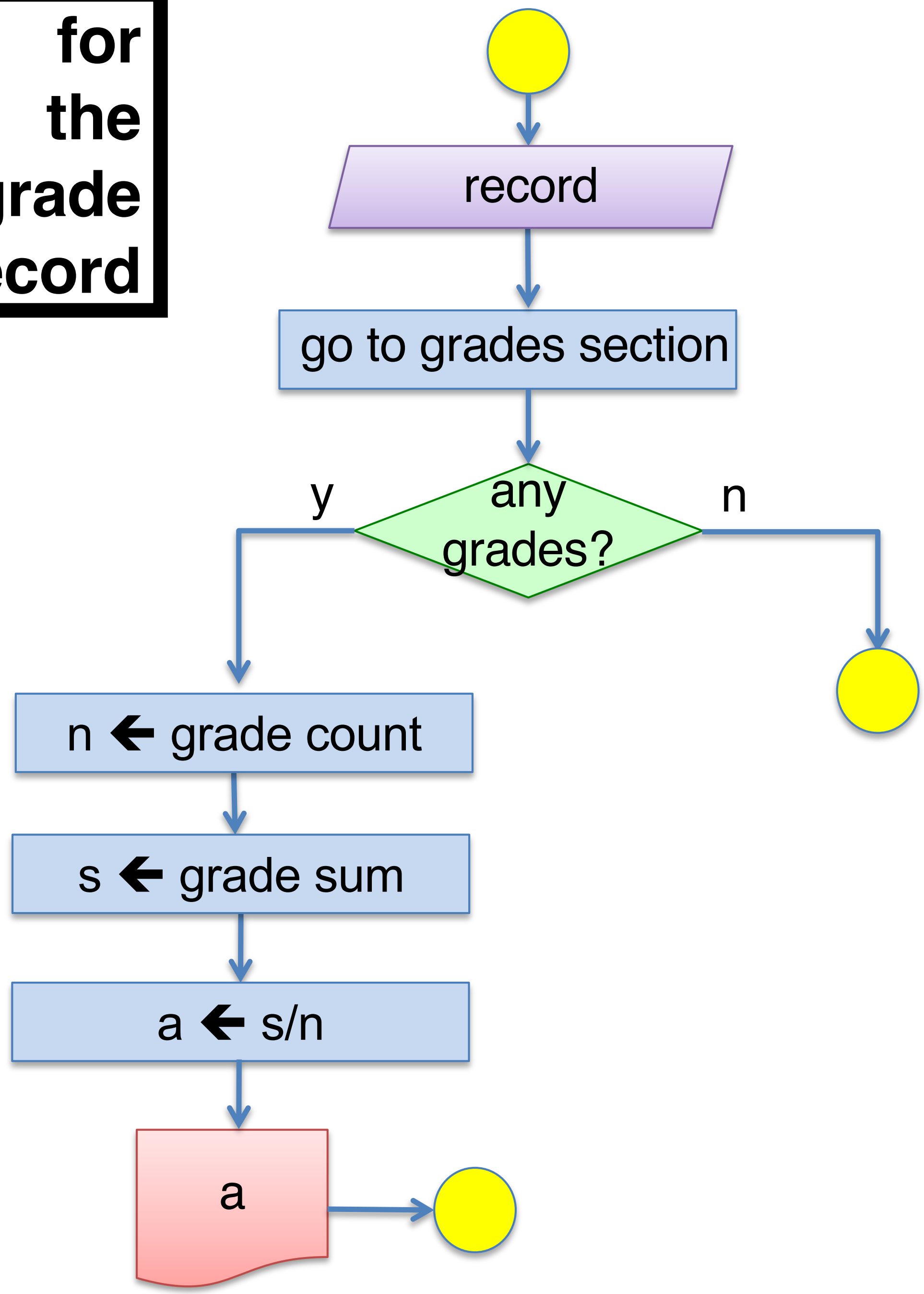
## Solution

**A possible solution is to take the record, go to the part dedicated to the grades and, if there are grades, count them, add them and finally divide the sum by the number of grades to obtain the average, which is the result we are looking for.**



The type of operations this solution is comprised of
input
A possible solution is to take the record, go to the part dedicated to the grades and, if there are grades, count them, add them and finally divide the sum by the number of grades to obtain the average, which is the result we are looking for.
operation
condition check
output

**Algorithm for calculating the average grade from a record**



s ← grade sum

**An operation of the type shown above is complex because it actually includes more than one step.**

**The first step is the one described to the right of the arrow and indicates an operation that typically produces a result.**

**The arrow represents the second step, called “assignment”: the result obtained must be stored in order to be used at a later time.**

**The letter or name to the left of the arrow represents the “place” where the result is saved and can be used as a reference to recall it.**

**In computer science, the “places” where the results of operations are saved are called “variables”. A term borrowed from mathematics which aims to indicate that this place can contain different, variable and non-constant values.**

## Variables

In the case of the algorithm for calculating the average of the grades we used three variables.

*n* to keep the number of votes in the record;

*s* to keep the sum of the votes;

*a* to keep the value of the average, obtained by recalling the values of *s* and *n* and making a division between them.

As the last operation of the algorithm, since the objective was to calculate the average of the votes, we recall the value of *a* and send it to output.

Always remember that an algorithmic solution to a problem is not the only possible solution: you can perform different operations that lead to the same result (for example 2 x 3 or 3 x 2 always give 6 as a result), or the same algorithm can be described at different levels of detail.

A more detailed alternative to the solution shown in these slides, for example, can specify to go to the beginning of the list of scores, check that there is still one score to count, increase the sum and the count each time and move to the position next repeating the previous operations until there are no more scores.

**Indeed, if there exists an algorithm to solve a problem, there are always infinite algorithms which solve it.**

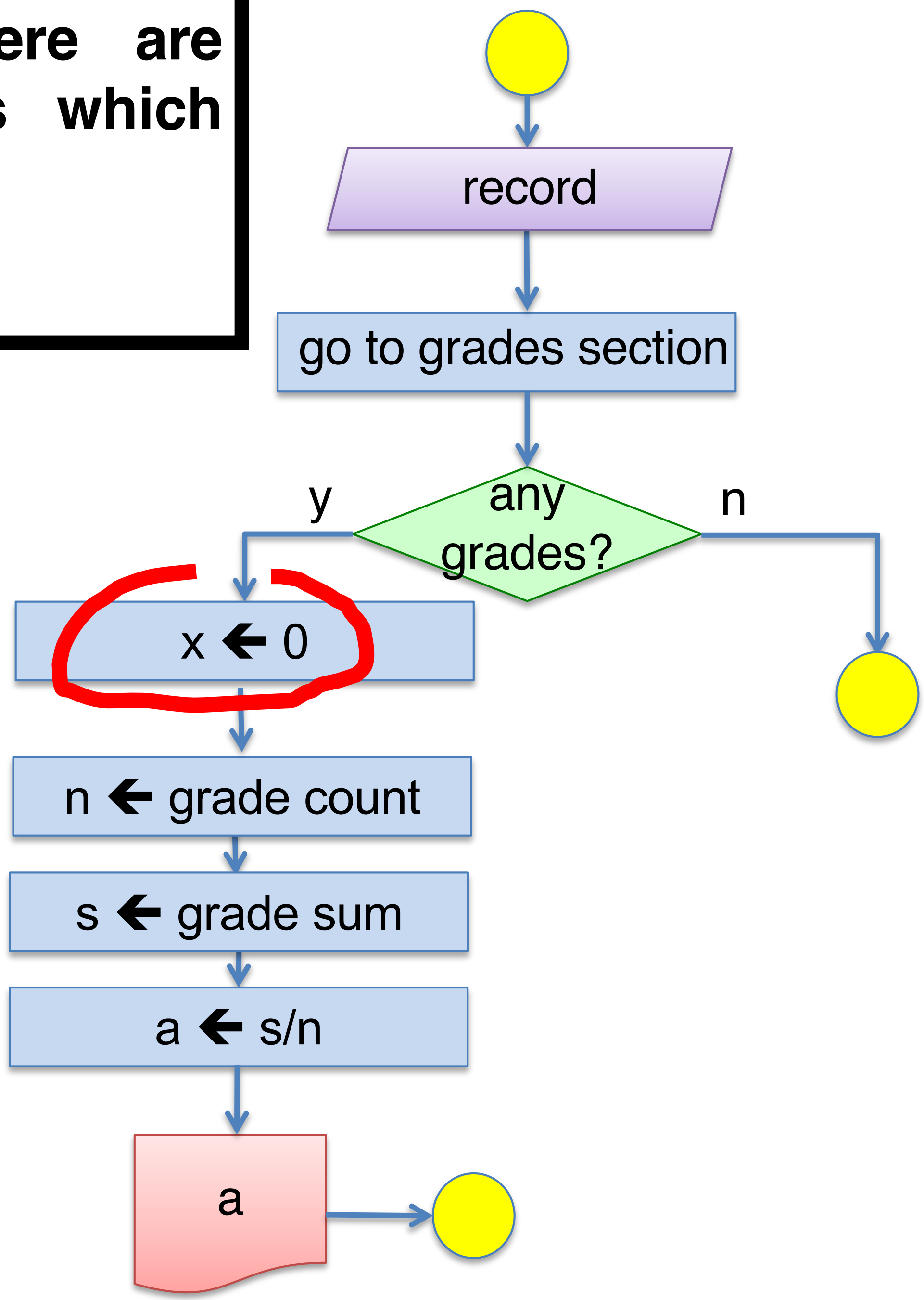
**Check this algorithm:**

**We have added an instruction that:**

1) **creates a new algorithm in terms of syntax/signs/data**

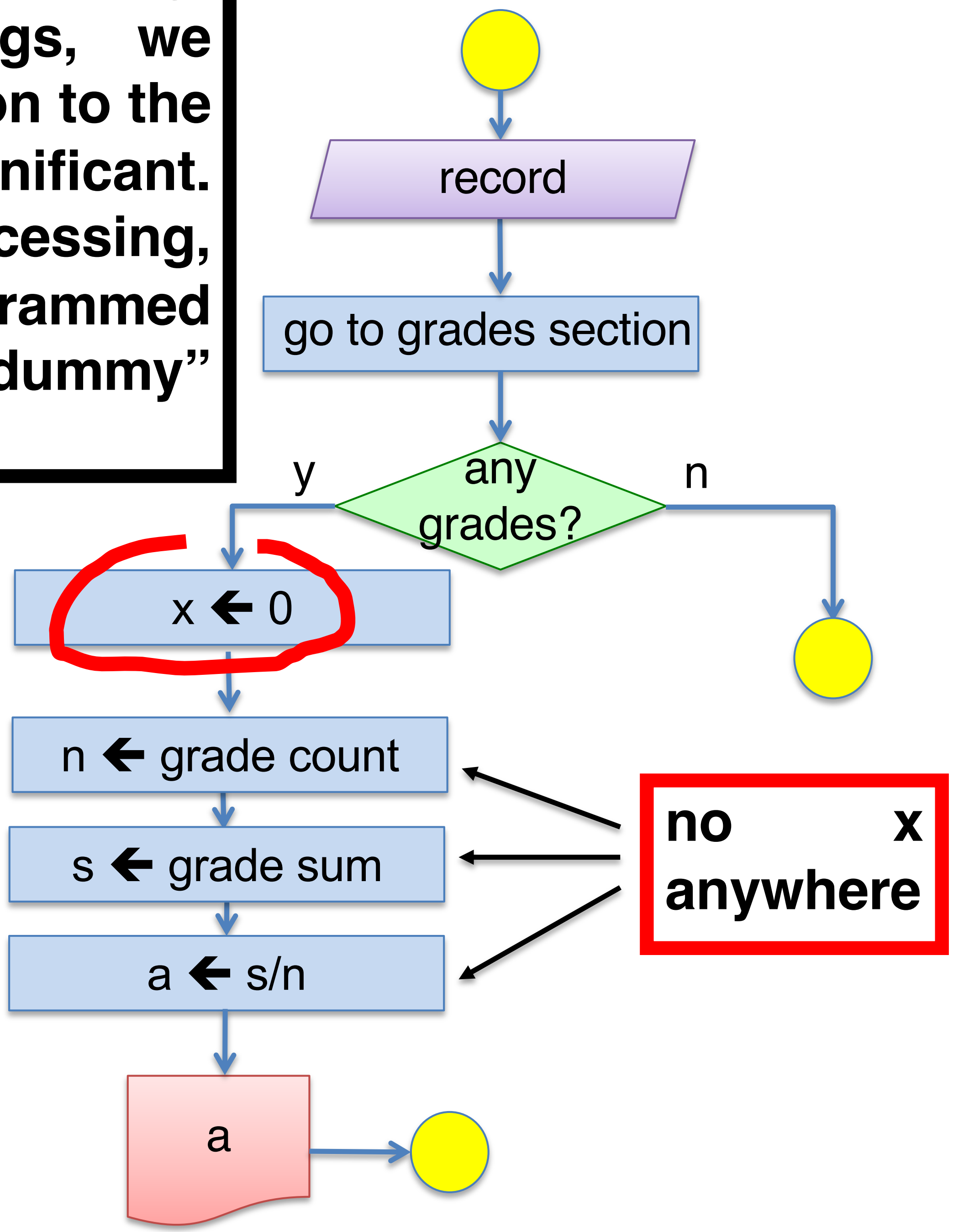
2) **but does not modify it in a meaningful way**

**We can choose any number instead of “0”, so there are infinite variations like this one.**



**As sense-making, meaning-entertaining human beings, we understand why this addition to the original algorithm is not significant. Can the symbol-processing, mindless computer be programmed in a way to catch these "dummy" instructions?**

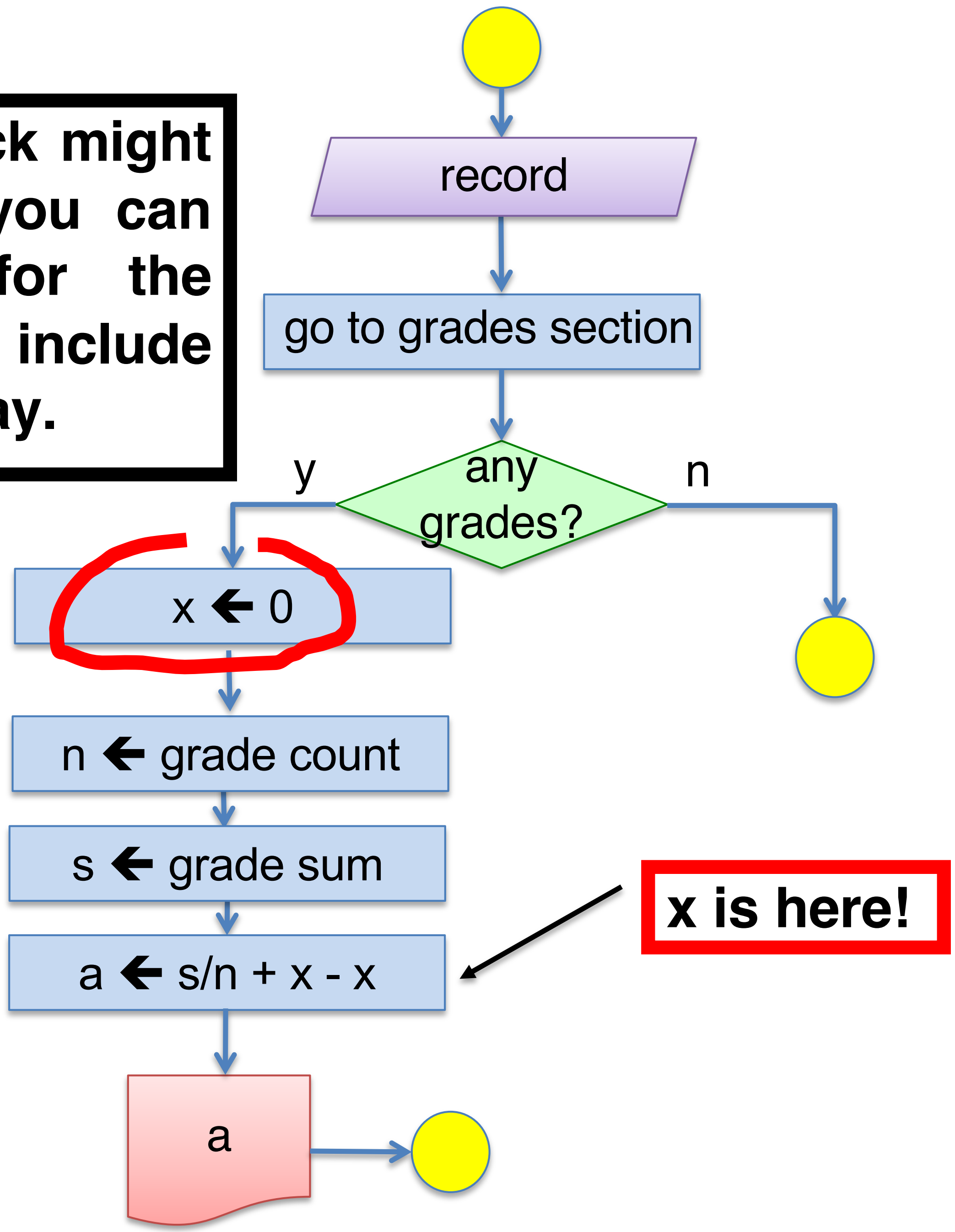
**There seems to be a syntactic way to check on the meaningfulness (or lack thereof) of *x* for the final result *a*: the fact that *x* is nowhere to be seen in the instructions that contribute to the computation of *a*.**



**That kind of syntactic check might be easily tricked. Below you can see a new operation for the computation of *a* that does include *x*, but not in a significant way.**

**In the end, meaningfulness seems to be something that only humans can check.**

**We are back at the syntax vs. semantics issue.**



After all, despite its incredibly more vast repertoire of uses, a computer does not differ so much from a knife, at least from the perspective of the need for a human to use it in a meaningful way.