COMPUTER SKILLS

LESSON 1

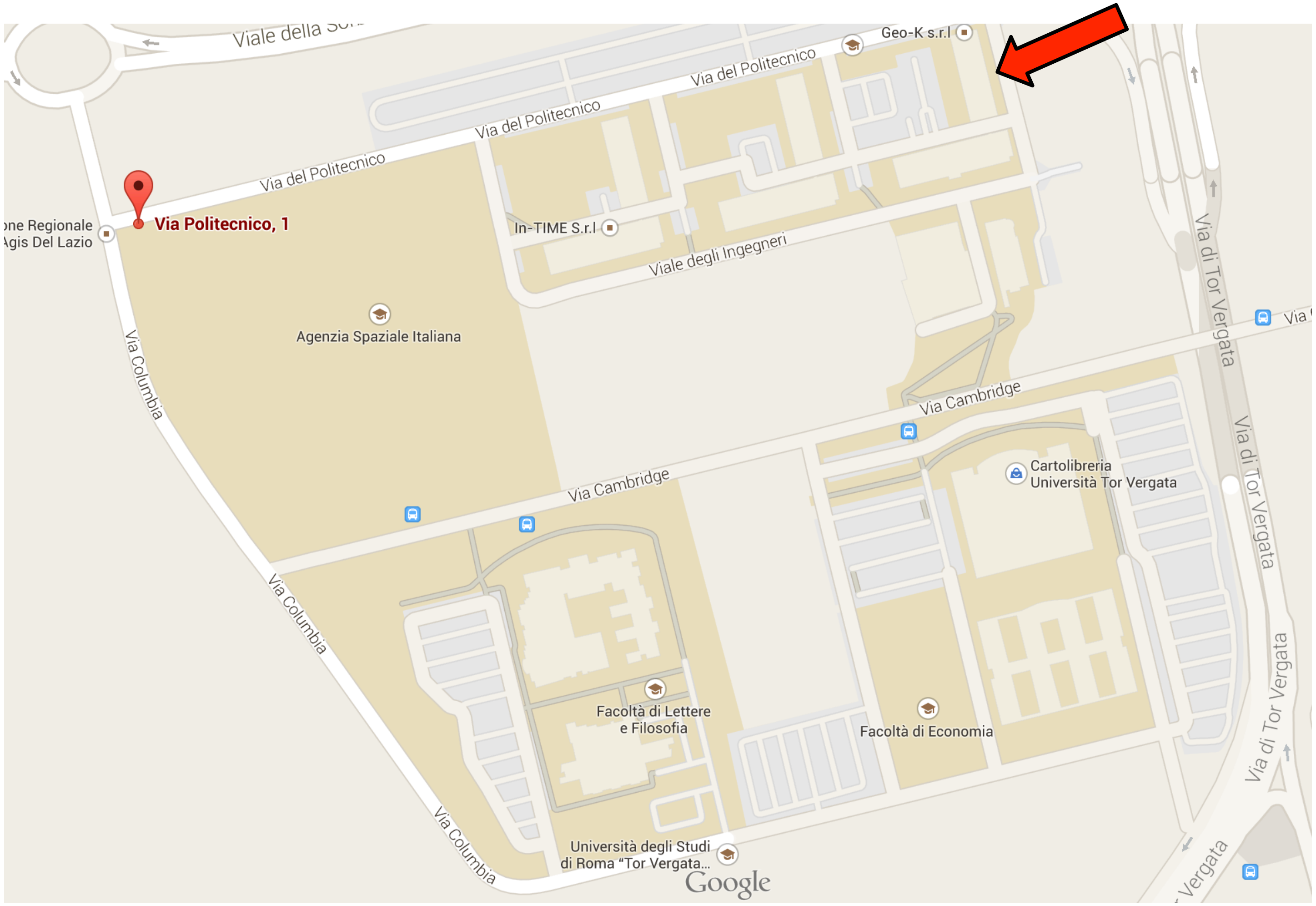
Valeria Cardellini
cardellini@ing.uniroma2.it

A.Y. 2015/16

Instructor

- Valeria Cardellini
- Associate Professor in Computer Science Department of Civil Engineering and Computer Science Engineering (DICII)
- Email: cardellini@ing.uniroma2.it
- Tel: 06 7259 7388
- Office hours
  - When: Tuesday 12:00-13:30
  - Where: room D1-17 (wing D, 1st floor), "Ingegneria dell'Informazione" building

Office location



Course Web site

- Official web site (authoritative source!)

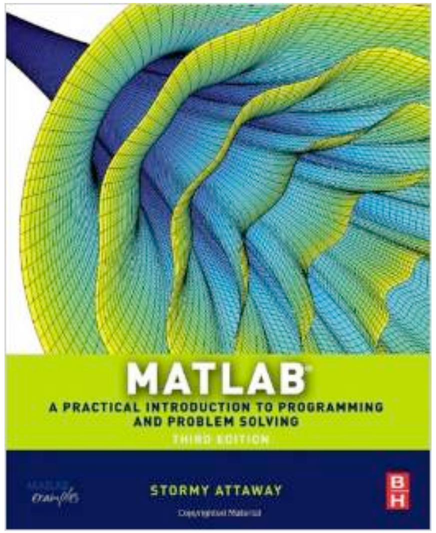
http://economia.uniroma2.it/ba/businesseconomics/corso/389/

- Where to find the course material
  - News, syllabus, lesson slides
  - Slides will be available after the lesson (not before!)

- Friends
- Forums
- Others

Textbook and teaching material

- Textbook
  - S. Attaway, “MATLAB: A Practical Introduction to Programming and Problem Solving, 3rd ed.”, Elsevier, 2014, ISBN: 978-0-12-405876-7
- Programming tools
  - MATLAB - http://www.mathworks.com
  - Octave - http://www.gnu.org/software/octave/
- Complementary material
  - Slides and suggested readings on the course site
  - Slides: credits to Dr. Emiliano Casalicchio, who shared the course teaching last year



Slides don’t replace the textbook. Slides are complementary material.

Both (slides and textbook) are essential to succeed!

Grading policy for the course

The Computer Skills exam is composed by

- **Written exam**
- **Practical exam** in the lab to evaluate the acquired programming skills

Grading policy

- Vote: A, B, C, and D
  - A is the best, D is the worst
- You have to pass ***both*** the written and the practical exams with at least C
- I will also take into account class participation
- The final vote is: passed/not passed

When

- Two exams in the period January-February 2016
- One exam in June 2016
- One exam in September 2016

About you and your background

- Do you own a computer (PC/laptop/tablet)?
- How do you use your PC/laptop? For which purpose?
  - What applications do you mainly use?
- Do you know about or have you ever used…
  - Microsoft Excel (or some spreadsheet)?
  - Microsoft Access (or some database management system)?
  - Programming languages?
- Do you know what an algorithm is?
- Do you know how a computer works?
  - Computer architecture
  - Operating system

Why?!

- Why should you develop computer skills?
  - To learn the art of solving problems with a computer (it will turn useful in your life!)
  - To learn and practice with a set of "tools" useful to solve problems related with your specialization and profession

- This is not my vision, but the vision of
  - Mr. Obama (US President): **"Don't just play on your phone, program it"**
    - US government initiative, now worldwide: Computer Science Education Week, annual campaign about the importance of learning computer science
      - Obama https://www.youtube.com/watch?v=yE6IfCrqg3s
      - Famous people https://www.youtube.com/watch?v=nKIu9yen5nc
      - Astronauts https://www.youtube.com/watch?v=WE2s3647awE
    - Hour of Code initiative https://hourofcode.com/

Why MATLAB/Octave as tools

- It is a framework (i.e., a set of tools and a programming language) that allows to manipulate data (e.g., numbers and strings)
  - Statistical analysis
  - Formulation and solving of mathematical problems
  - Data plots
  - Simulation
- MATLAB offers a high level programming language, that makes programming easier than other programming languages (Java, C, C++, Python, Fortran, …)
- It will allow you to turn theory in practice and develop your computer skills

Definition of computer science or informatics

- The “science of computers” (aka: Computer Science or CS)
  - CS is the discipline dealing with representing, storing, retrieving, and processing information by automated means

“Computer science is no more about computers than astronomy is about telescopes.”

E. Dijkstra (Dutch computer Scientist. Turing Award in 1972. 1930-2002)

- The “science of information” (aka: Informatics)
  - The science of information representation and (automatic) processing

Definitions

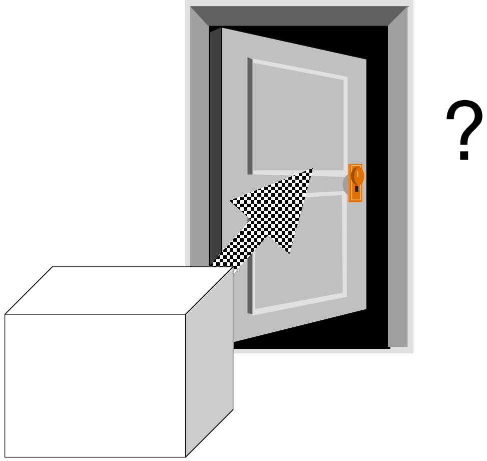
- “The science of information representation and (automatic) processing”

- What is *information* ?
  - Why are we interested in its *representation* and *(automatic) processing* ?

↕

- Actually, we are interested in “solving problems”
  - Precisely, in “***systematic ways*** for solving problems”
  - *Computer Science* is concerned with this

A problem to be solved



- What do we need to solve this problem “abstractly”?
  - A **representation** of the involved entities
  - A **procedure** to be followed (based on the chosen representation)

1. **Representation**
   - A way to express the relevant information for the problem at hand

2. **Procedure**
   - A step-by-step process to follow in order to find the solution

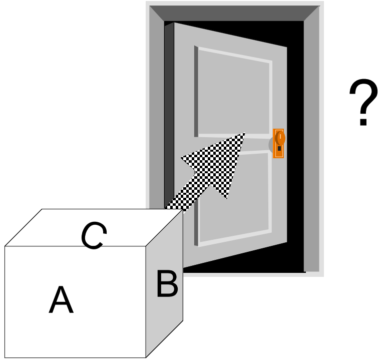
Other examples of problems

- To find a candidate with a specific skill among 1000 CVs
- To find a song on the Web
- To remove “red eyes” or other “flaw” from a photo
- To evaluate the best investment in stocks
- To forecast the growth of the “PIL”

Representation + Process

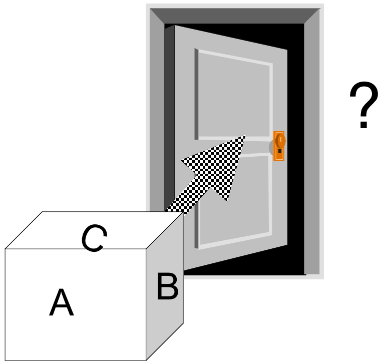
- “The science of information representation and (automatic) processing”
  - Now it should be a bit clearer what we are talking about
  - We still have to discuss more explicitly the meaning of “automatic”
  - Coming soon…

The box-and-door problem again …



- Box representation
  - face A: height(A), width(A)
    - e.g.: height(A) = 220 cm, width(A) = 70 cm
  - face B: height(B), width(B)
    - e.g.: height(B) = 220 cm, width(B) = 110 cm
  - face C: height(C), width(C)
    - e.g.: height(C) = 70 cm, width(C) = 110 cm
- Door representation
  - height(door), width(door)
    - e.g.: height(door) = 210 cm, width(door) = 80 cm

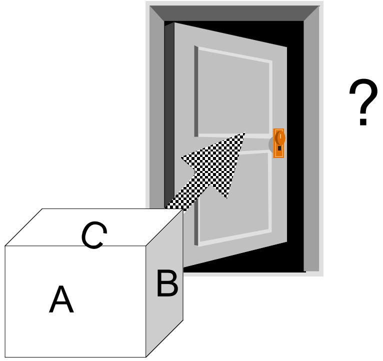
The box-and-door problem again …



- Procedure
    - check face A
        - check: height(A)<height(door)) AND width(A)<width(door)
    - if true, OK ; else, check face B
        - check: height(B)<height(door)) AND width(B)<width(door)
    - if true, OK ; else, check face C
        - check: height(C)<height(door)) AND width(C)<width(door)
    - if true, OK ; else FAILURE
- OK ?
    - hint: try with the numbers given in the previous slide …

- A new procedure
    - check: height(A)<height(door)) AND width(A)<width(door)
    - if true, OK ; else, **"rotate" face A**
        - **check: width(A)<height(door)) AND height(A)<width(door)**
    - if true, OK ; else, check face B
        - …

"General" procedure



- check: height(A)<height(door)) AND width(A)<width(door)
- if true, OK ; else, "rotate" face A
- check: width(A)<height(door)) AND height(A)<width(door)
- if true, OK ; else, check face B

- Does this procedure work only for these specific box and door?

- We are interested in general procedures
  - **"Parametric" procedures**

“Automatic” processing

- box representation
  - face A: height(A), width(A)
  - face B: height(B), width(B)
  - face C: height(C), width(C)
- door representation
  - height(door), width(door)

representation

+

- check: height(A)<height(door)) AND width(A)<width(door)
  - if true, OK ; else, “rotate” face A
- check: width(A)<height(door)) AND height(A)<width(door)
  - if true, OK ; else, check face B
- …

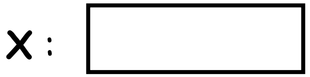
procedure

- “The science of information representation and (automatic) processing”
- What do we mean by “automatic” ?

Automatic processing

- To repeat tedius and “stupid” operations a huge amount of times
  - e.g., to process 1 million of pixels
  - to compare billions of information
  - …

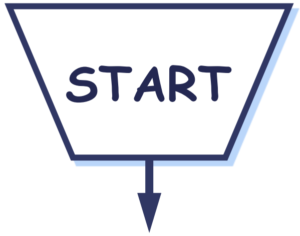
A formalism to write procedures



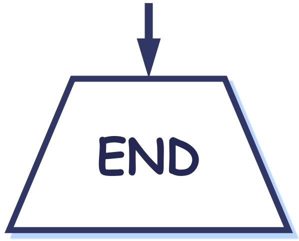
a named "container"

X ← *expr*

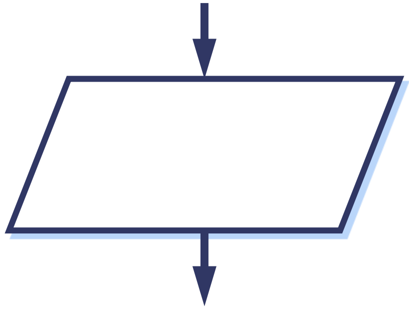
put a value in a "container"



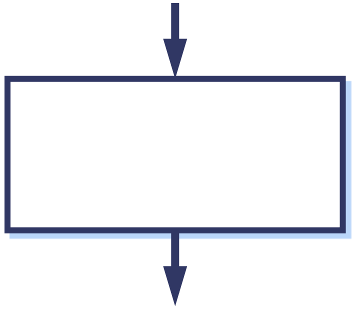
recipe processing begins



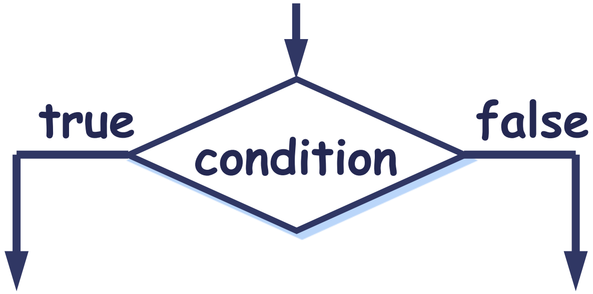
recipe processing ends



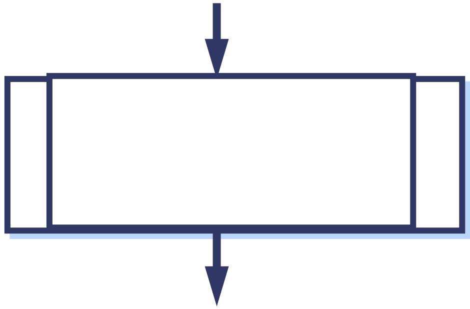
input / output



recipe step

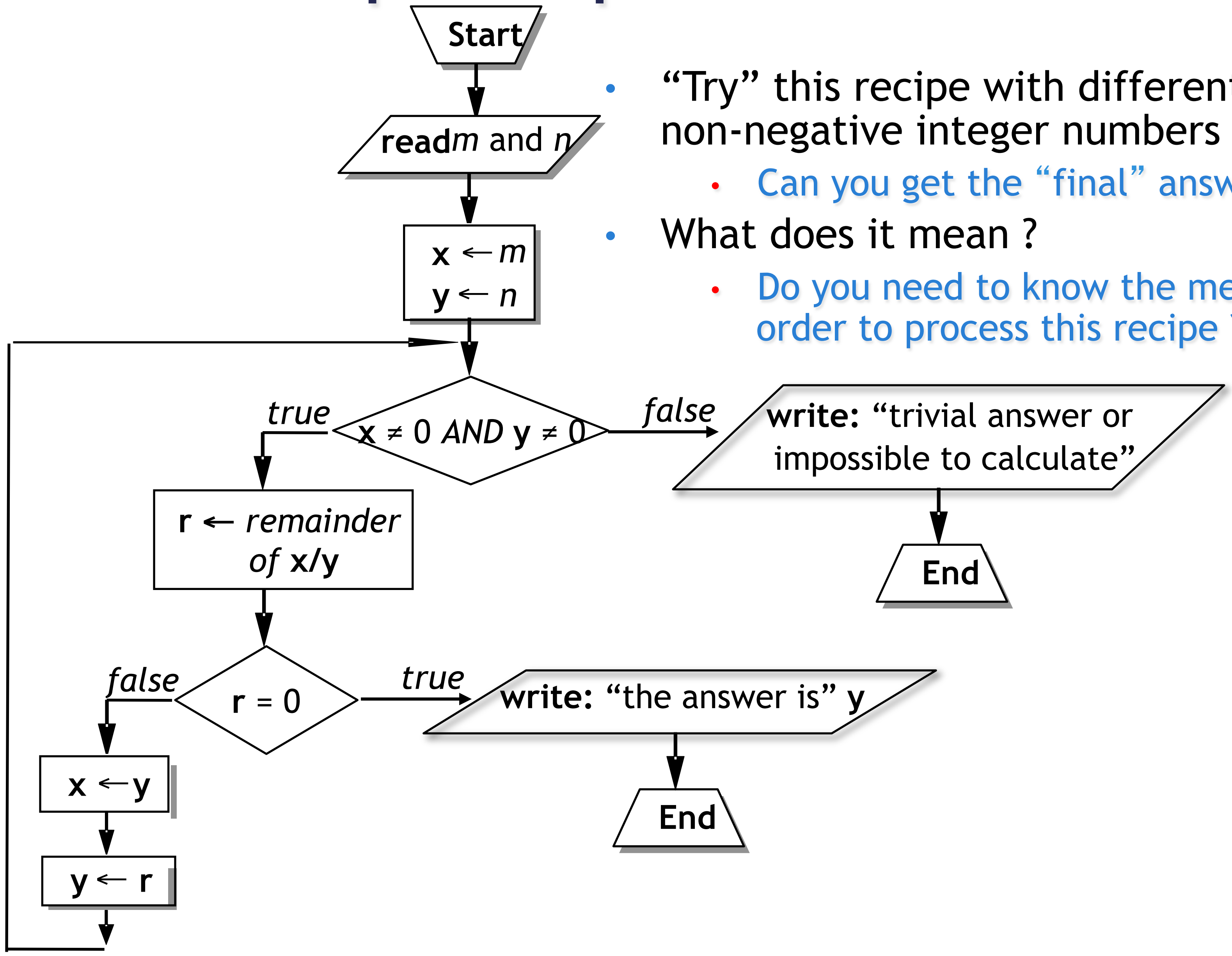


selection of alternative paths



"collateral" recipe

An example of procedure



- "Try" this recipe with different pairs of non-negative integer numbers
  - Can you get the "final" answer?
- What does it mean ?
  - Do you need to know the meaning in order to process this recipe ?

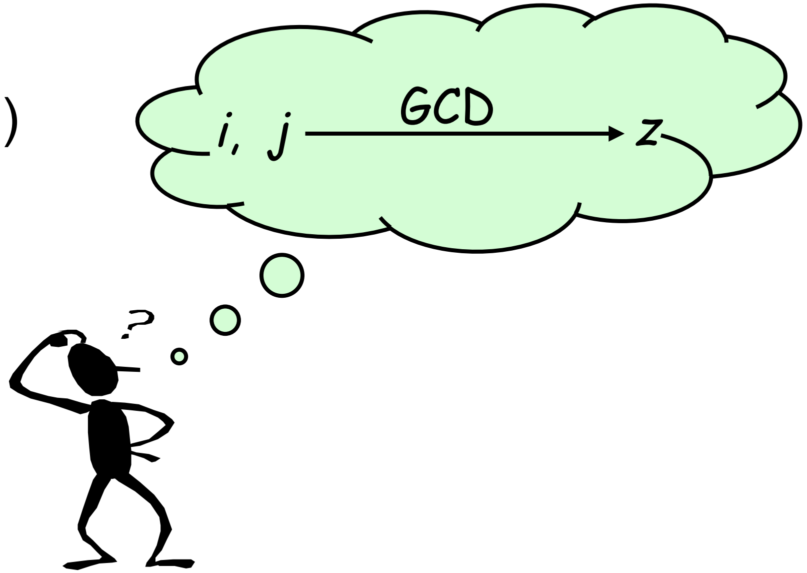
GCD = Greatest Common Divisor

A mathematical problem

- Determine $z = \mathrm{GCD}(i, j) \quad i, j \in \mathrm{N}$

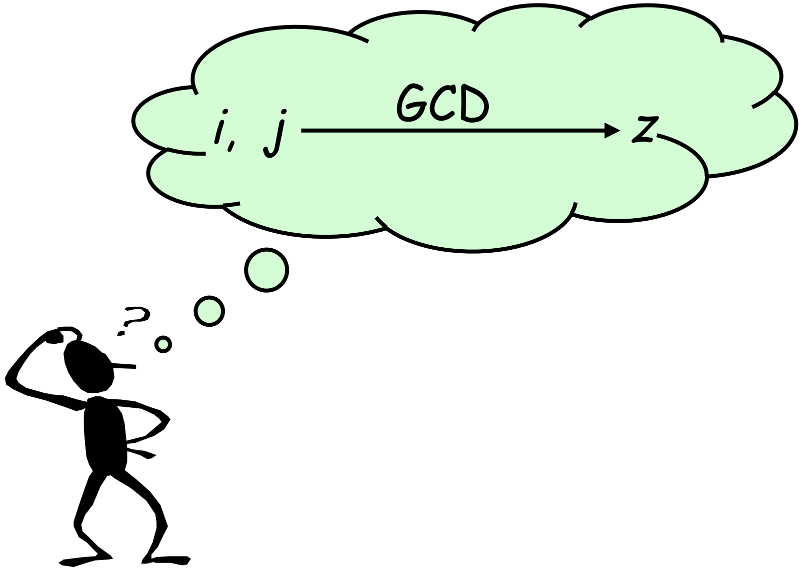
- $\mathrm{D}(i) \overset{def}{=}$ the set of the integer divisors of i
- $\mathrm{D}(i) \;=\; \{\, k \mid i = k \cdot q, \quad k \in \mathrm{N}^{+}, \quad q \in \mathrm{N} \,\} \quad \mathrm{N}^{+} = \mathrm{N} - \{0\}$

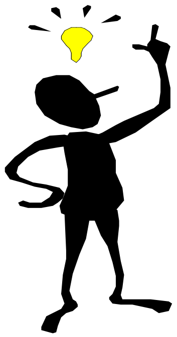
- $z = \mathrm{GCD}(i, j) = \max(\, \mathrm{D}(i) \cap \mathrm{D}(j) \,)$

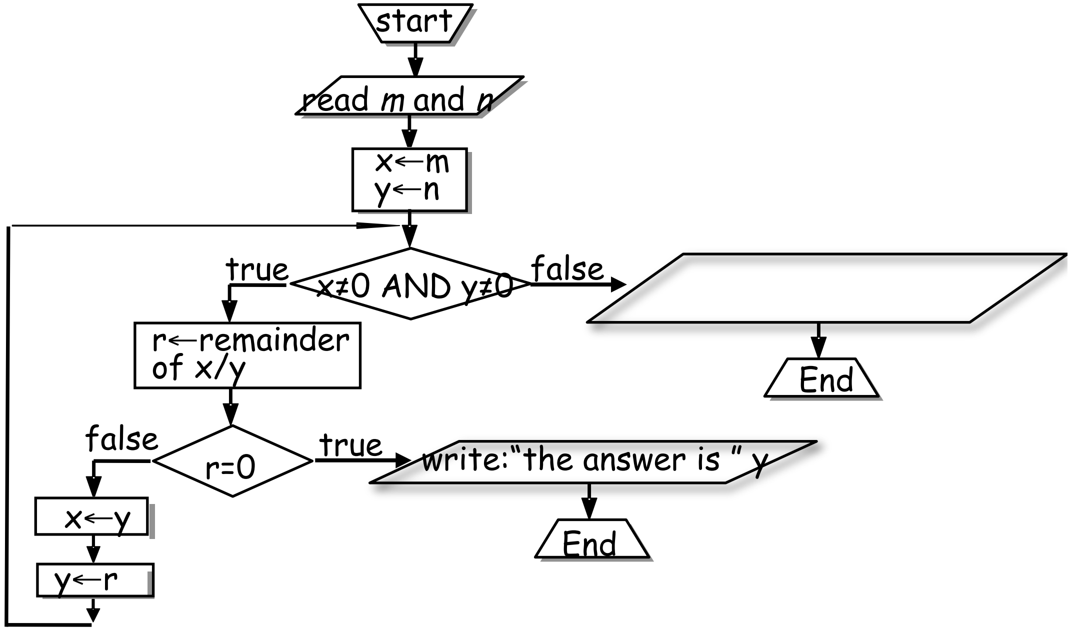


Greatest Common Divisor (GDC) of two or more integers, when at least one of them is not zero, is the largest positive integer that divides the numbers without a remainder.

Matching problems with procedures







BTW: this procedure to solve the GCD problem is known as "**the Euclid's algorithm**" (300 B.C.)

Selecting baseball cards: the problem

- For example, say you have a big collection of baseball cards and you want to find the names of the 10 “qualified” players with the highest lifetime batting averages.
- To qualify, the players must have been in the league at least 5 years, had at least 100 plate appearances per year, and made fewer than 10 errors per year.
- The cards contain all the relevant information for each player. You just have to organize the cards to solve the problem.

Selecting baseball cards: the steps

Clearly there are a number of steps between the stack of cards and the solution. Without any particular order, these steps are:

a. Write down the names of the players from some cards
b. Sort the stack of cards by the lifetime batting average
c. Select all players from the stack with 5 years or more in the league
d. Select all players from the stack with fewer than 10 errors per year
e. Select all players from the stack with over 100 plate appearances per year
f. Keep the first 10 players from the stack

Selecting baseball cards: the procedure

The solution might be:

In any order

c. Select all players from the stack with 5 years or more in the league

d. Select all players from the stack with fewer than 10 errors per year

e. Select all players from the stack with over 100 plate appearances per year

b. Sort the stack of cards by the lifetime batting average

f. Keep the first 10 players from the stack

a. Write down the names of the players from these cards

Computer Science / Informatics is about …

- **Procedures** to solve problems
  - The *methodological* facet
    - Problem solving and information management
- “**Machines**” to process procedures
  - The *technological* facet
    - (presently) electronic computing devices and systems that use them
- Both have a long history
  - Methodologies (Euclid ~300 B.C., …, al-Kuwarizmi ~1000 A.D., …, Hilbert ~1800 A.D., Gödel, Turing, ~1900 A.D., …)
  - Machines (abacus ?B.C., …, Babbage engine ~1800 A.D., … ENIAC ~1940 A.D., …)

Summary

- Course info and organization
- Definitions of computer science and informatics
- Introduction to problem solving
    - The recipe way