Lesson 1 - Introduction and algorithms

Logical Computational Thinking

Stefano MARTINA

stefano.martina@gmail.com

Tecnológico de Monterrey

 Scuola Leonardo Da Vinci (Firenze)

7 September 2015

Material

Material

all the material will be available at
https://github.com/trianam/courseLCT1516

Book

https://www.dropbox.com/s/umx65z3m9bnm6xj/Metodologia_de_la_programacion__3ra_Edicion_-_Osvaldo_Cairo_Battistutti.pdf

Material

Material

all the material will be available at
https://github.com/trianam/courseLCT1516

Book

https://www.dropbox.com/s/umx65z3m9bnm6xj/Metodologia_de_la_programacion__3ra_Edicion_-_Osvaldo_Cairo_Battistutti.pdf

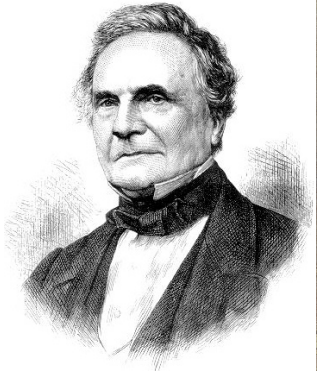
Evaluation

- ✓ You will be evaluated continuously along the lectures
  - exercises
  - questions
  - etc...
- ✓ and with exams
  - 2 partials (maybe 1)
  - 1 final (project)

History

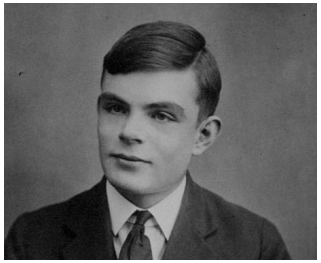
- ✓ classical age and middle ages: algorisms

- ✓ 1833-1842: Analytical engine of Charles Babbage (Ada Byron)

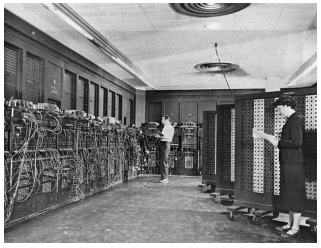


History

- ✓ before and during WW2: first modern computers (single purpose, programmable), Turing studies

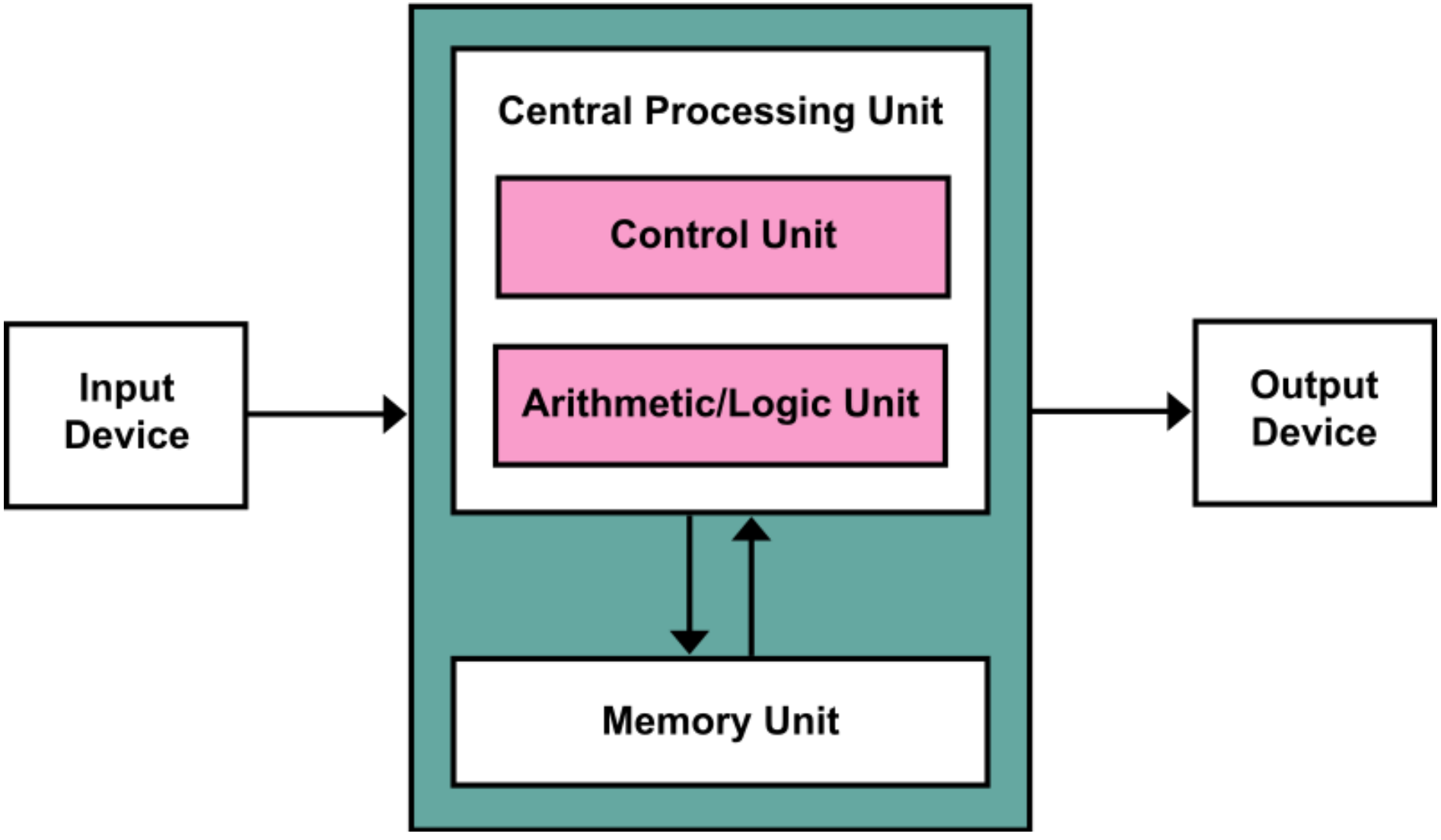


- ✓ 1946: ENIAC (general purpose)



- ✓ 1951: EDVAC, Von Neumann architecture

Computer architecture (Von Neumann)



Input/Output

Input devices

- ✓ Keyboard
- ✓ Mouse

Output devices

- ✓ Screen
- ✓ Printer

Input/output devices

- ✓ Hard disk
- ✓ Network card

Input/Output

Input devices

- ✓ Keyboard
- ✓ Mouse

Output devices

- ✓ Screen
- ✓ Printer

Input/output devices

- ✓ Hard disk
- ✓ Network card

Input/Output

Input devices

- ✓ Keyboard
- ✓ Mouse

Output devices

- ✓ Screen
- ✓ Printer

Input/output devices

- ✓ Hard disk
- ✓ Network card

Algorithms

- ✓ a series of ordered steps
- ✓ with the goal of performing a task

Examples

- ✓ a recipe
- ✓ an algebraic procedure

Algorithms

- ✓ a series of ordered steps
- ✓ with the goal of performing a task

Examples

- ✓ a recipe
- ✓ an algebraic procedure

Program

- ✓ An implementation of an algorithm in a certain programming language (software)
- ✓ A program can be executed by a machine (hardware)
- ✓ Often a program need to be compiled before the execution (transformed in something understandable from the machine)

Program

- ✓ An implementation of an algorithm in a certain programming language (software)
- ✓ A program can be executed by a machine (hardware)
- ✓ Often a program need to be compiled before the execution (transformed in something understandable from the machine)

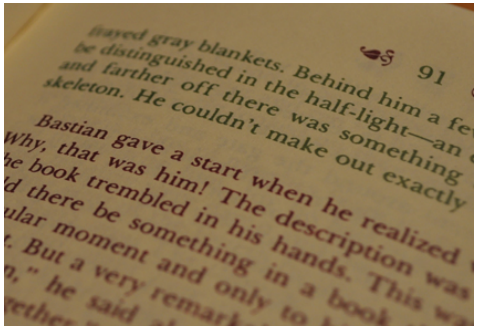
Program

- ✓ An implementation of an algorithm in a certain programming language (software)
- ✓ A program can be executed by a machine (hardware)
- ✓ Often a program need to be compiled before the execution (transformed in something understandable from the machine)

Literary comparison

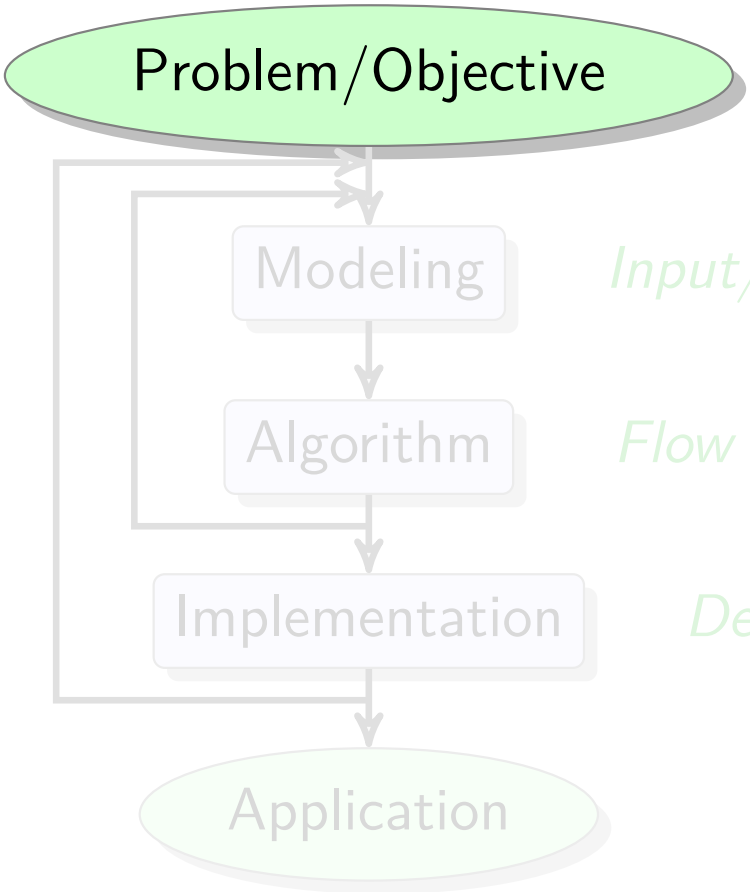
Algorithm: the history

Program: the text



Hardware: the book

How to develop a program

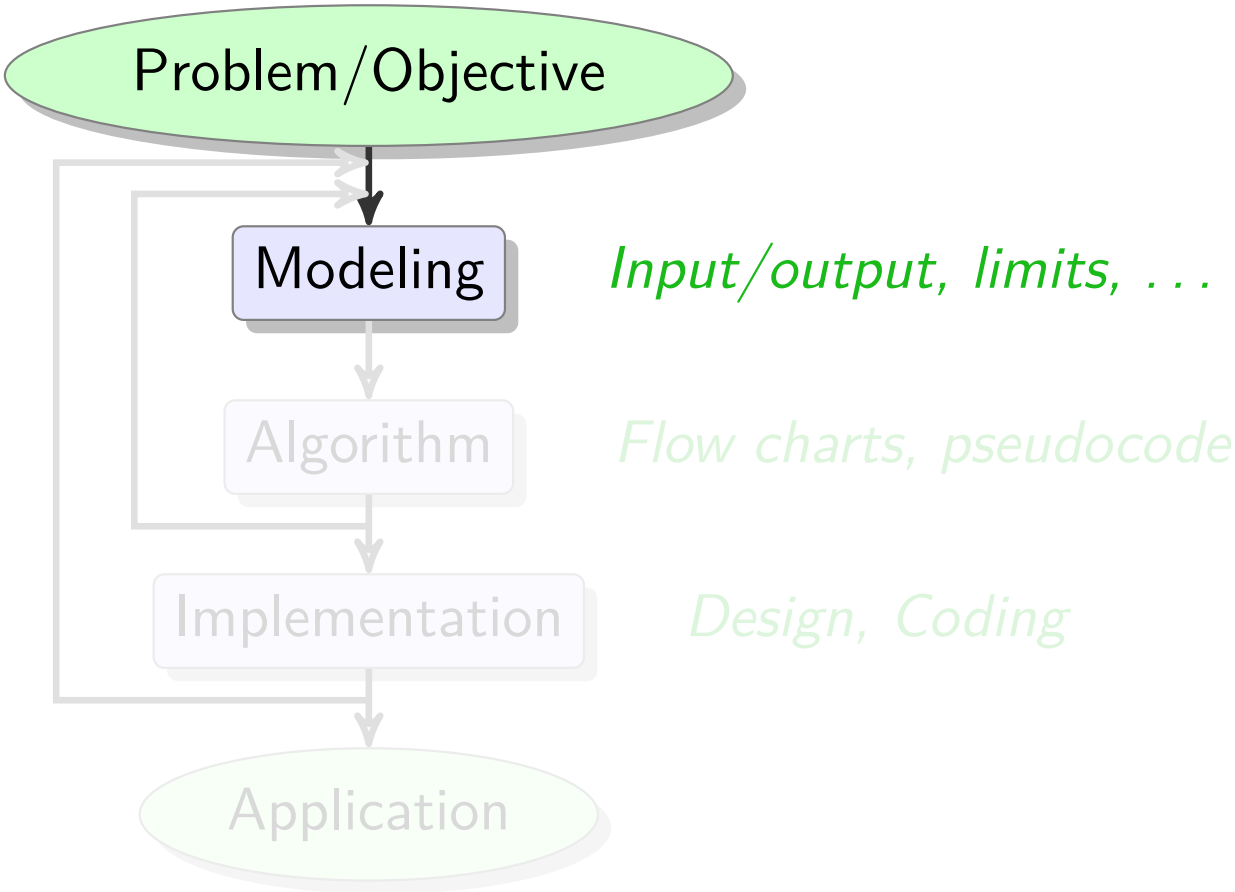


Input/output, limits, . . .

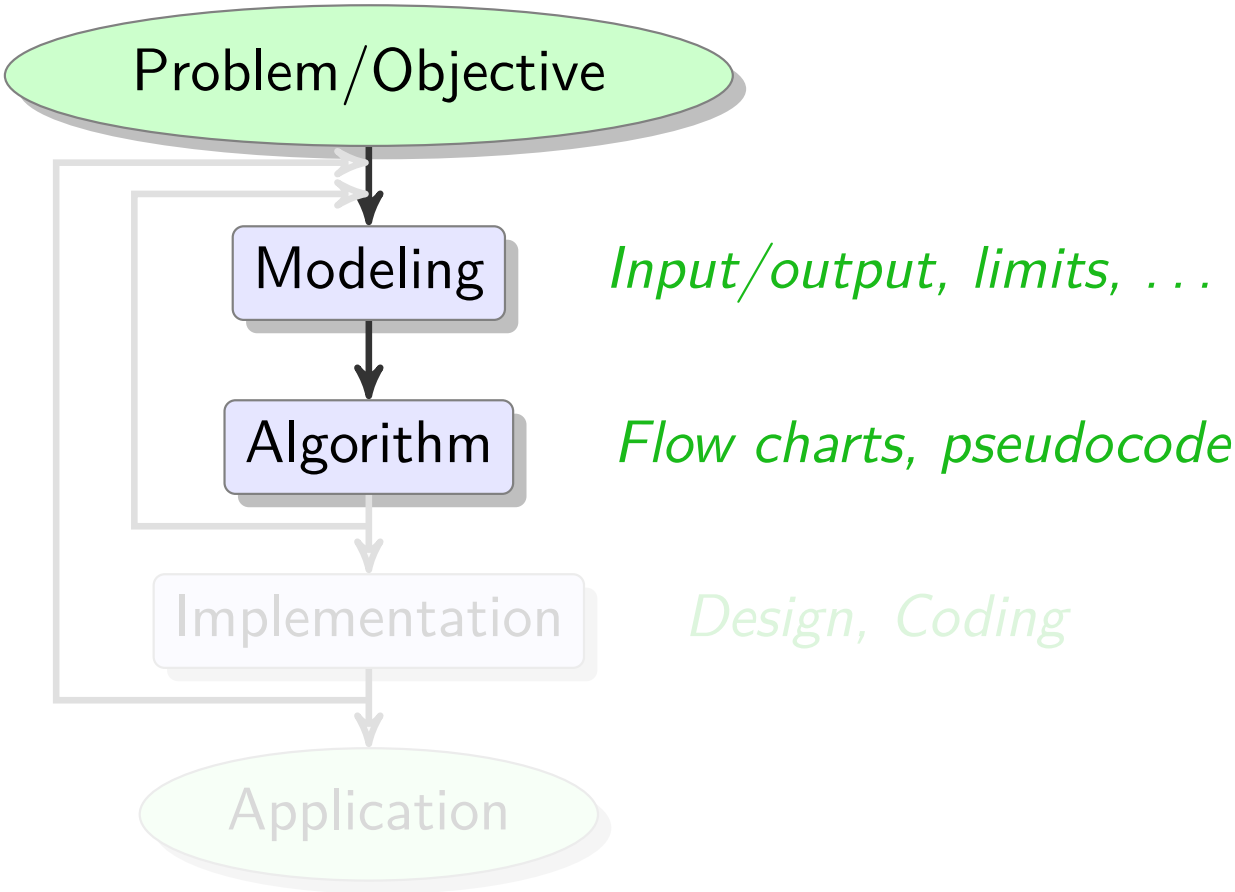
Flow charts, pseudocode

Design, Coding

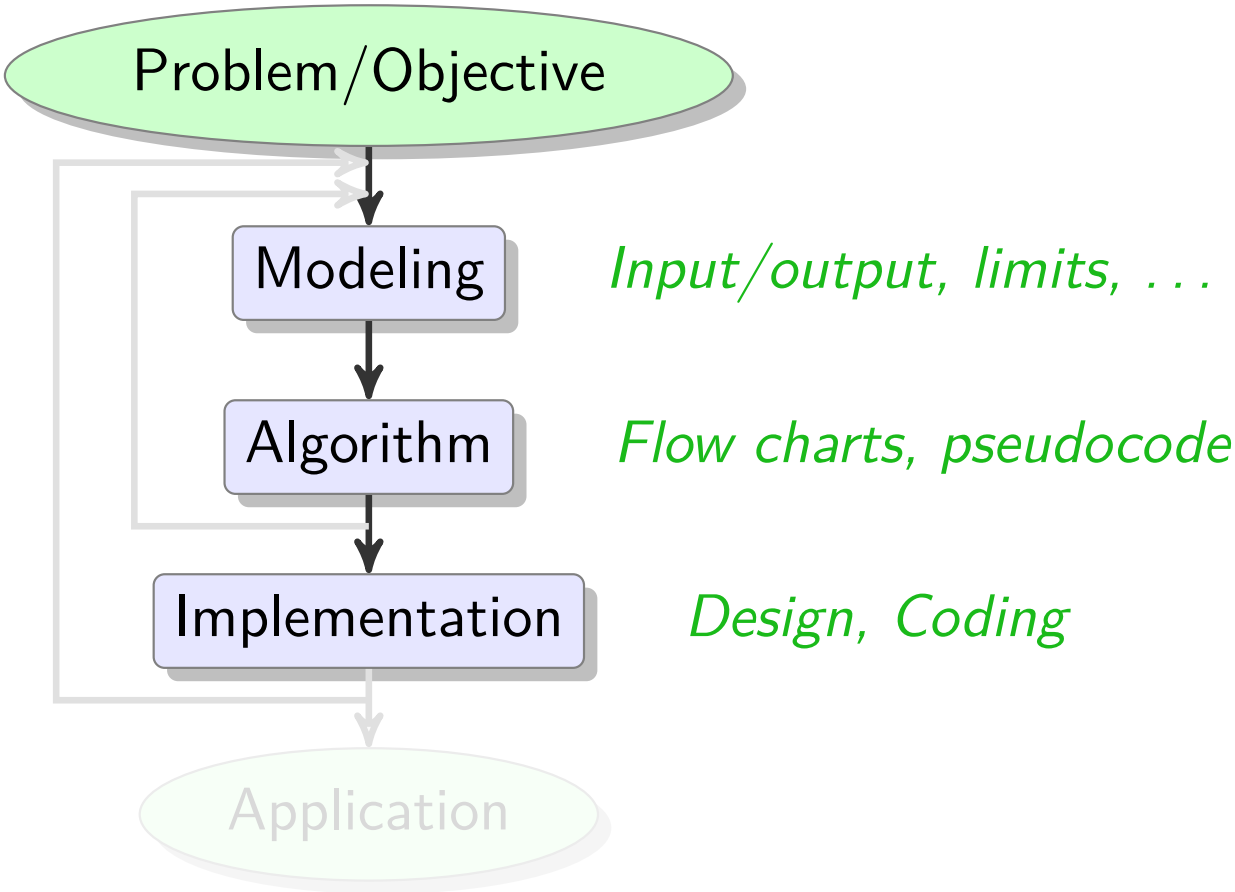
How to develop a program



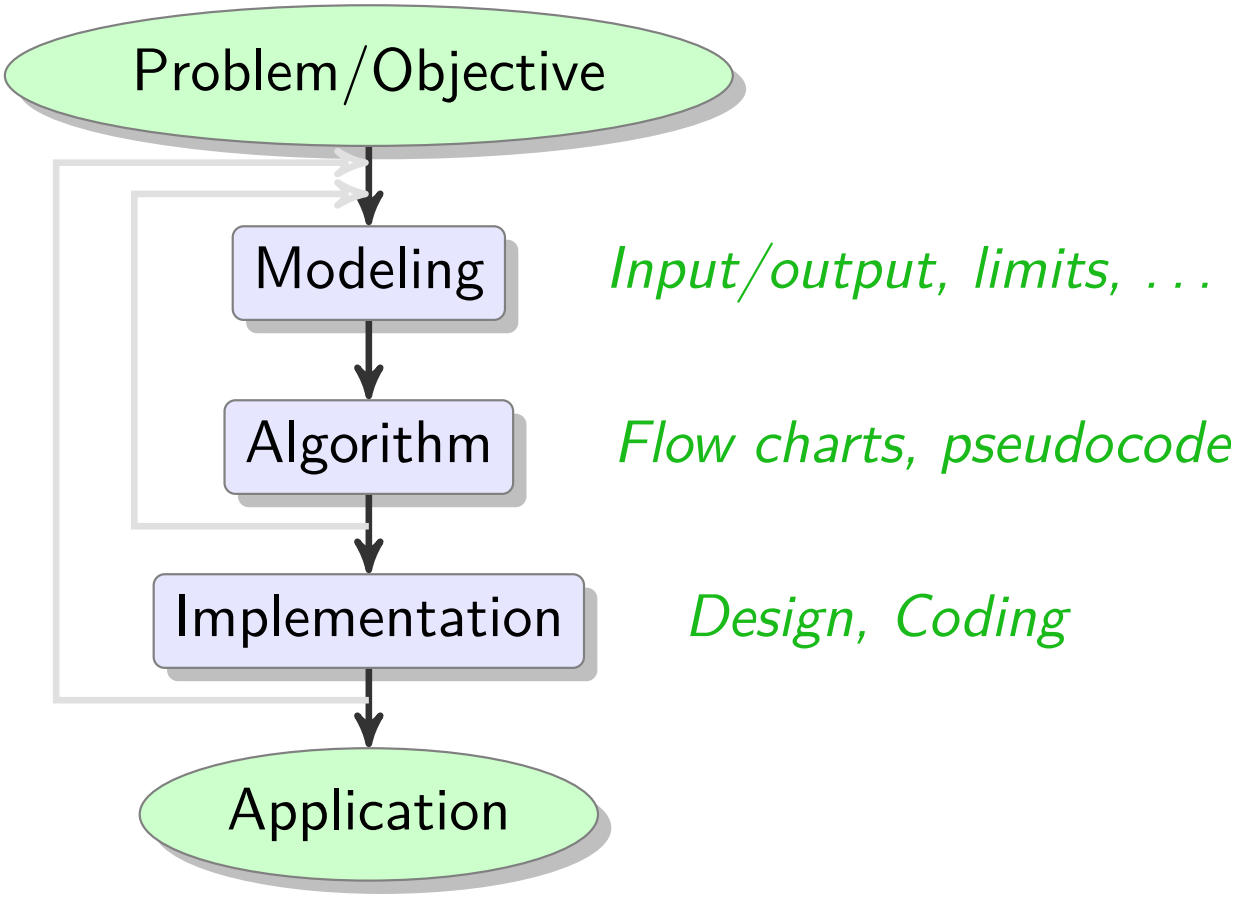
How to develop a program



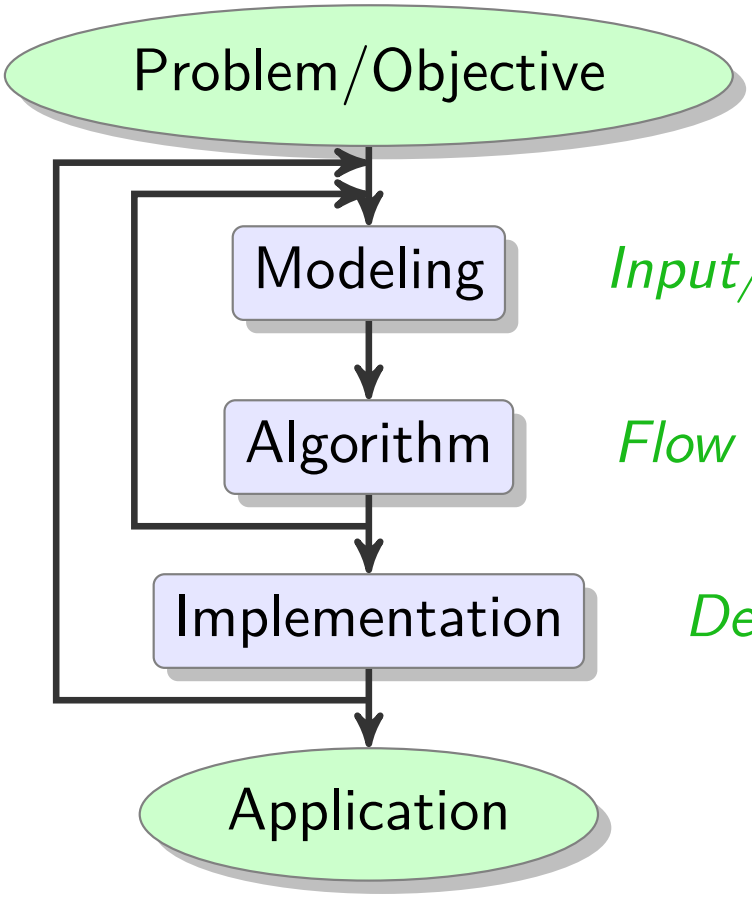
How to develop a program



How to develop a program



How to develop a program



Input/output, limits, . . .

Flow charts, pseudocode

Design, Coding

Model

1. Analyze the problem or the required objective
2. Contextualize in an algorithmic way
3. Identify the key concept/mechanisms, how to divide the problem in subproblems

Algorithm

Concept

Use techniques to "put down" ideas on how to resolve the problem.

1. Flow charts
2. Pseudocode

Implementation

Transform the algorithm in code

Algorithm

Concept

Use techniques to "put down" ideas on how to resolve the problem.

1. Flow charts
2. Pseudocode

Implementation

Transform the algorithm in code

Algorithm

Concept

Use techniques to "put down" ideas on how to resolve the problem.

1. Flow charts
2. Pseudocode

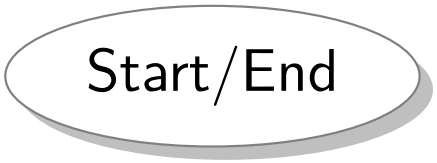
Implementation

Transform the algorithm in code

Flow charts

Definition

Symbols with different meanings and descriptions, combined with the logic of the flow along the time

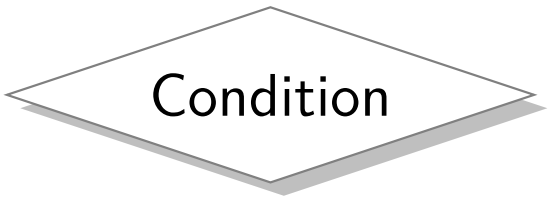


Start and end of the flow chart

Input, reading

Output, writing

Assignation, operation



Selection, split the flow

Data types

Data type

is the mean how the data is stored and manipulated

1. Simple represent single values
   - boolean
   - integer number
   - floating point number (real number approximation)
   - alfanumeric character
2. Structured composed of multiple values
   - string of characters
   - array of values
   - ...

All the data types have specific limits (depending of the programming language)

Data types

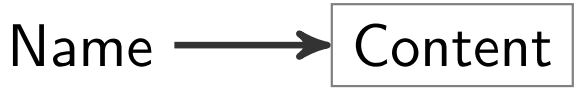
Data type

is the mean how the data is stored and manipulated

1. Simple represent single values
   - boolean
   - integer number
   - floating point number (real number approximation)
   - alfanumeric character
2. Structured composed of multiple values
   - string of characters
   - array of values
   - ...

All the data types have specific limits (depending of the programming language)

Variables and costants



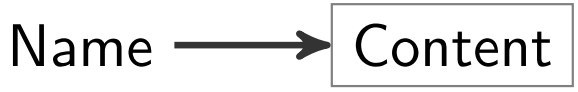
Variable

- ✓ A name associated with a data type
- ✓ Use a certain amount of memory (specific to the data type)
- ✓ The content can be modified

Constant

- ✓ A name associated with a data type
- ✓ Use a certain amount of memory (specific to the data type)
- ✓ The content can't be modified

Variables and costants



Variable

- ✓ A name associated with a data type
- ✓ Use a certain amount of memory (specific to the data type)
- ✓ The content can be modified

Constant

- ✓ A name associated with a data type
- ✓ Use a certain amount of memory (specific to the data type)
- ✓ The content can't be modified

Example 1

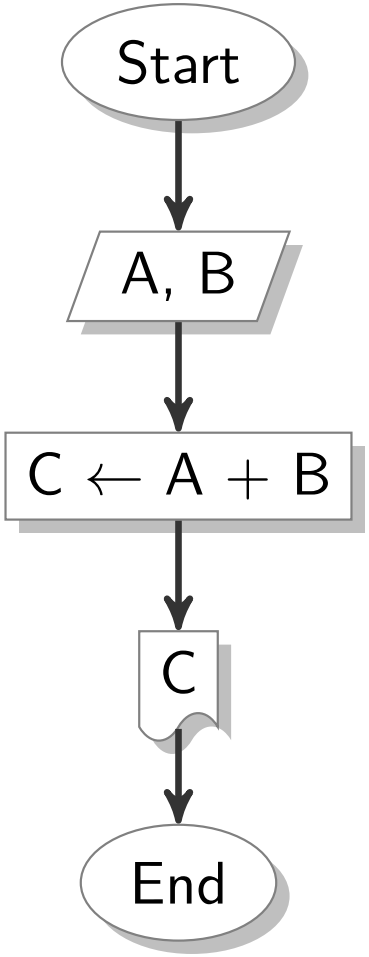
Problem

Given two integer A and B, calculate the sum A+B and return it

Example 1

Problem

Given two integer A and B, calculate the sum A+B and return it



Example 2

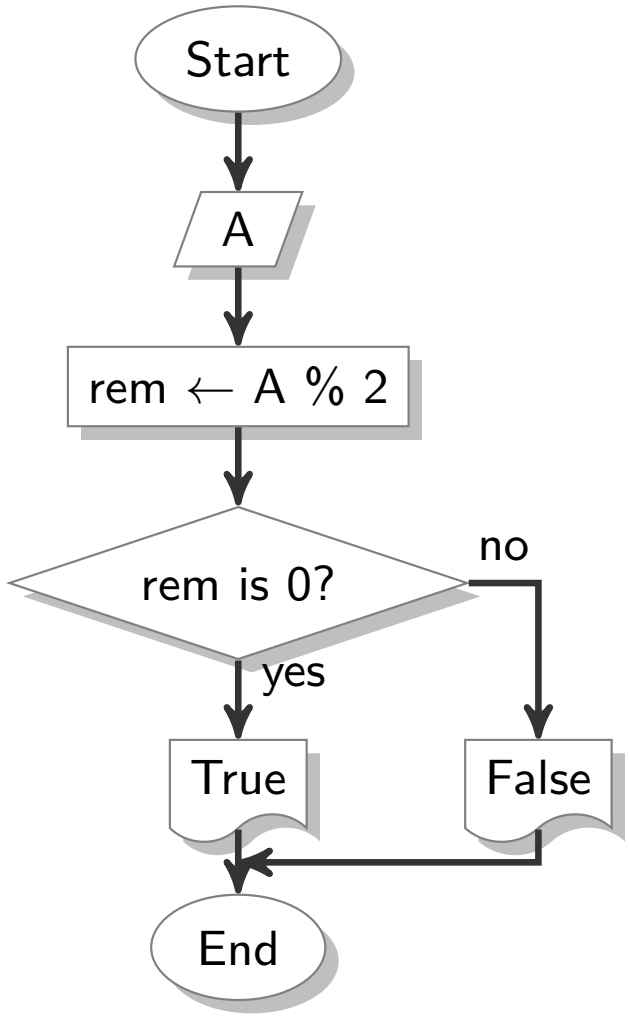
Problem

Given an integer A, evaluate it's evenness value, and return true if A is even, and false if is odd

Example 2

Problem

Given an integer A, evaluate it's evenness value, and return true if A is even, and false if is odd



Exercise 1

Problem

Given the reals A, B, and C (A is $\neq 0$), calculate the solutions x_1 and x_2 of the equation $Ax^2 + Bx + C = 0$ and return them. If the equation doesn't has solutions (real solutions), return $x_1 = 0$ and $x_2 = 0$.

Remember:

1. $\Delta = B^2 - 4AC$
2. if $\Delta \geq 0$ the solutions are:

$$x_1 = \frac{-B+\sqrt{\Delta}}{2A}$$
$$x_2 = \frac{-B-\sqrt{\Delta}}{2A}$$

Exercise 1

Problem

Given the reals A, B, and C (A is $\neq 0$), calculate the solutions x_1 and x_2 of the equation $Ax^2 + Bx + C = 0$ and return them. If the equation doesn't has solutions (real solutions), return $x_1 = 0$ and $x_2 = 0$.

Remember:

1. $\Delta = B^2 - 4AC$
2. if $\Delta \geq 0$ the solutions are:

$$x_1 = \frac{-B+\sqrt{\Delta}}{2A}$$
$$x_2 = \frac{-B-\sqrt{\Delta}}{2A}$$

Exercise 2

Problem

We have a parking with a limited number of 50 places. We also have two sensors that notify the passage of a car, one in the entrance and one in the exit. We want to put a semaphore in the entrance that is red when the parking is full and green when it isn't.

1. Make a proper model, identify the different parts of the application
2. Make a flow chart for each part

Exercise 2

Problem

We have a parking with a limited number of 50 places. We also have two sensors that notify the passage of a car, one in the entrance and one in the exit. We want to put a semaphore in the entrance that is red when the parking is full and green when it isn't.

1. Make a proper model, identify the different parts of the application
2. Make a flow chart for each part