Programming Paradigms

General Introduction

Didier Verna

didier@didierverna.net


didierverna.net


@didierverna


didier.verna


in/didierverna

Build 2025-06-13 11:40:27+02:00

Outline

Plan

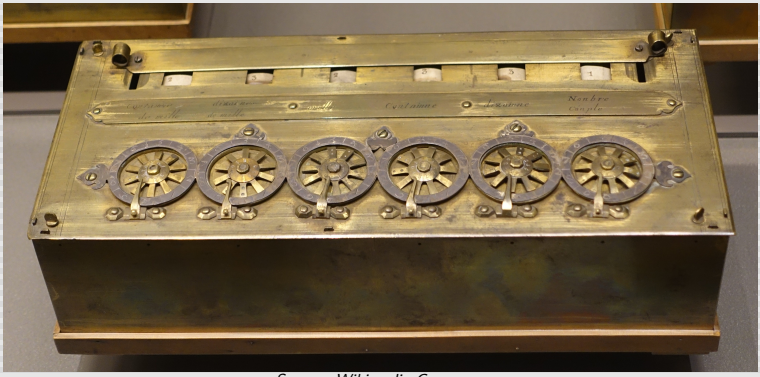
Build 2025-06-13 11:40:27+02:00

Plan

Build 2025-06-13 11:40:27+02:00

1642: The Pascaline

Source: Wikimedia Commons



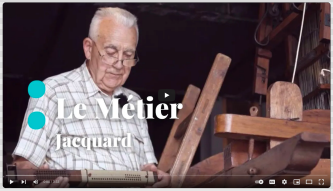
Source: Wikimedia Commons

- Abstract work, manual processing

1801: The Jacquard Loom

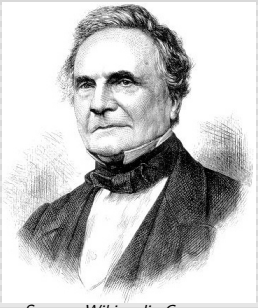
Source: Wikimedia Commons

Source: Wikimedia Commons



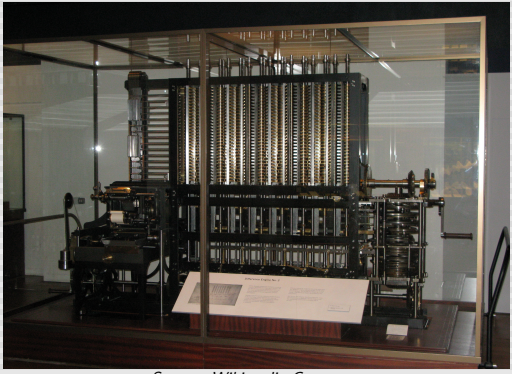
- Concrete work, automatic processing

1834: The Difference Engine (Babbage / Byron)



Source: Wikimedia Commons

Source: Wikimedia Commons

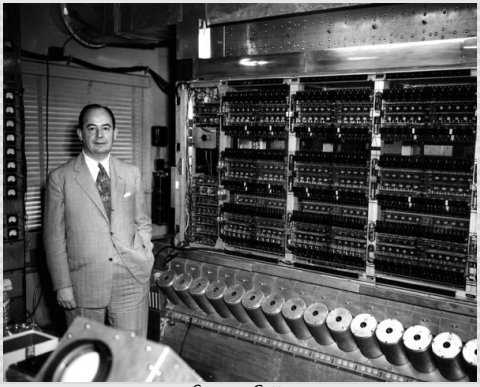


Source: Wikimedia Commons

- Abstract work, automatic processing
  - program encoded on a punch card
  - “magazine” for storing the results

Plan

1945: John Von Neumann



Source: Getty

- Program also stored in the ~~magazine~~ “memory”
- Encoding of *all* information in binary (“homoiconicity”)

1936: Alan Turing

Source: Wikimedia Commons

- Formalization of the concept of “algorithm”
- Turing Machine

Plan

Plan

Build 2025-06-13 11:40:27+02:00

Imperative Programming

Program

```
instruction 1
instruction 2
...
instruction n
```

- **Instruction:** for producing side effects
- **Side effect:** modification of the surrounding environment

⚠ The execution order is important!

Example

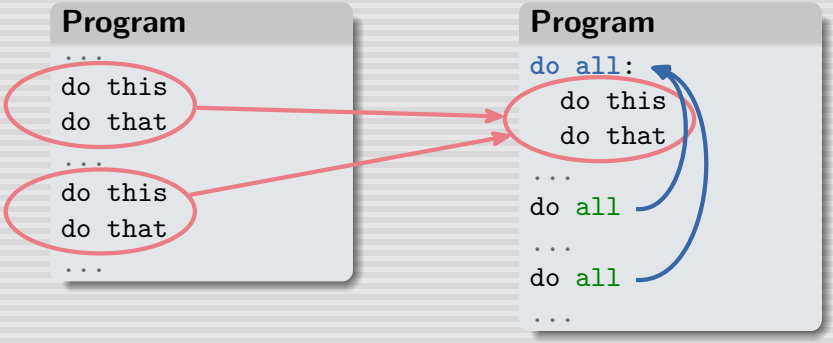
```
LDA $4242 ; load mem. address 0x4242 into acc.
INA       ; increment acc.
STA $4242 ; store the value of acc. back into mem. address 0x4242
```

Plan

Build 2025-06-13 11:40:27+02:00

Procedural Programming

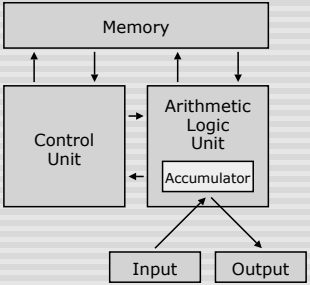
- Natural extension to imperative programming
- Factor out a sequence of frequently repeated instructions into a *procedure* that can be called as many times as needed



Plan

Reminder: Origins

- Hardware Paradigm
  Von Neumann Model
- Machine Language
  1. manipulation of registers
  2. exchanges with memory

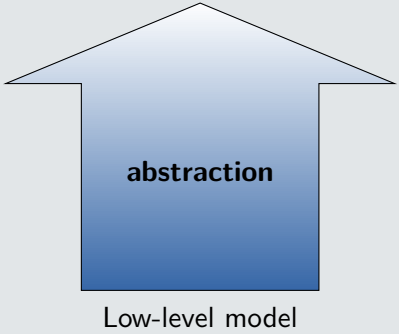


Source: Wikimedia Commons

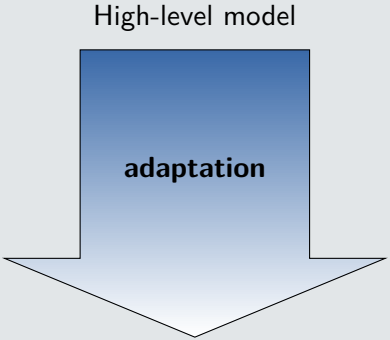
Source: Wikimedia Commons

Two Families of Languages

Bottom-Up



Top-Down



- Imperative / procedural programming is bottom-up

⚠ This is about programming languages design / evolution, not usage!

Plan

Plan

Concept of Paradigm

- Etymology
  - παράδειγμα (paradeigma): pattern, example, sample
- Epistemology
  - «The Structure of Scientific Revolutions» [Kuhn, 1962]
    Consensus on theories and postulates → revolution → ...
  - «The Paradigms of Programming» [Floyd, 1979]
    1978 ACM Turing Award Lecture
- Definition
  - General thought framework ⟺ Form of expression (both are intertwined !)
  - «language» first, «Programming» next

⚠ Sclerosis hazard (cognitive bias) !
Cf. «Teach Yourself Programming in 10 Years» [Norvig, 2001]

Concept of Paradigm

- Etymology
  - παράδειγμα (paradeigma): pattern, example, sample
- Epistemology
  - «The Structure of Scientific Revolutions» [Kuhn, 1962]
    Consensus on theories and postulates → revolution → ...
  - «The Paradigms of Programming» [Floyd, 1979]

> *A language that doesn't affect the way you think about programming, is not worth knowing.*
> — *Alan Perlis*

- Def
  - «Programming» first, «language» next

⚠ Sclerosis hazard (cognitive bias) !
Cf. «Teach Yourself Programming in 10 Years» [Norvig, 2001]

Programming Paradigms

- **A lot:** imperative, procedural, object-oriented, functional, logic, reflexive, *etc.*
- **Turing-Completeness:** identical expressivity from one language to another
- **The Correct question:**
  ~~«Can we express X ?»~~
  «Can we express X *easily* ?»

Source: Wikimedia Commons

Examples

Functional

```haskell
map (+4) [1, 2, 3, 4, 5] -- [5, 6, 7, 8, 9]
```

Logic

```prolog
father(didier,coline).
man(X) :- father(X,_).
? - man(didier).      % yes
```

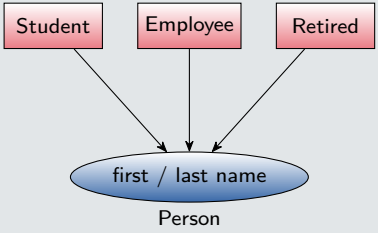
Reflexive

```lisp
(defvar program (list '+ 1 2))
(eval program)                  ;; 3
```

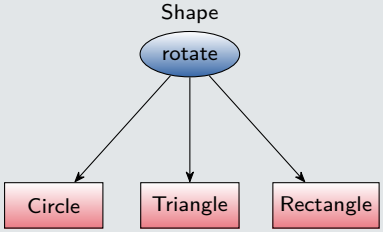
Plan

Two Limitations in Procedural Programming

State convergence



Behavior divergence



- OOP is here to lift those limitations

  Welcome to " Object-Oriented Approaches to Programming!" 😉

Plan

Bibliography

Bibliography

Thomas Kuhn.
The Structure of Scientific Revolutions.
University of Chicago, 1962.

Robert W. Floyd.
The Paradigms of Programming.
Communications of the ACM, vol.22 n.8, 1979.

Peter Norvig.
Teach Yourself Programming in 10 Years.
2001.