pico-Céu:

A minimalist, visual,and interactive programming environment focusing on education.

Francisco Sant'Anna

francisco@ime.uerj.br

https://github.com/fsantanna/pico-ceu

“Hello World!”

```
emit GRAPHICS_DRAW_PIXEL(25,25);
```

```
emit GRAPHICS_DRAW_PIXEL(25,25);
await KEY_PRESS;
emit GRAPHICS_DRAW_PIXEL(26,26);
```

```
emit GRAPHICS_DRAW_PIXEL(25,25);
await 1s;
emit GRAPHICS_DRAW_PIXEL(26,26);
await 1s;
emit GRAPHICS_DRAW_PIXEL(27,27);
await 1s;
emit GRAPHICS_DRAW_PIXEL(28,28);
```

Goals

- Concrete problems
  - avoid abstract/mathematical problems
    - fibonacci, primes, gdc, sort
  - seek visual and playful applications
    - focus on input/output, **time**
    - propose modifications that require maths
- Project oriented
  - portfolio building, creativity
  - individualization
    - no best/single/correct solution
    - make exams optional and reduces plagiarism
- Multiple target audiences
  - computer science, engineering, visual arts, schools, kids

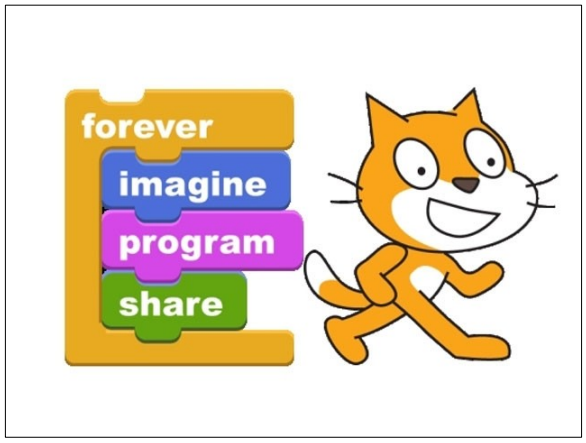
Approach

- Straightforward graphics
  - immediate feedback
  - pixel-level manipulation and visualization
- Simple development cycle
  - minimalist API
  - easy installation and execution
- Explicit I/O operations
  - `await` for input
  - `emit` for output
- Structured synchronous programming model
  - sequential execution, no callbacks
  - logical parallelism, deterministic concurrency

Inspirations

- Basic
  - simple/immediate graphics API

- pico8
  - 8-bit “fantasy console”
- Scratch
  - “constructionist learning”
  - “creative computing”



Input & Output

```
var integer x ← 25;
var integer y ← 25;
emit GRAPHICS_DRAW_PIXEL(x,y);

loop do
    var integer key ← await KEY_PRESS;

    if key = KEY_LEFT then
        x ← x - 1;
    else/if key = KEY_RIGHT then
        x ← x + 1;
    else/if key = KEY_UP then
        y ← y - 1;
    else/if key = KEY_DOWN then
        y ← y + 1;
    end

    emit GRAPHICS_DRAW_PIXEL(x,y);
end
```

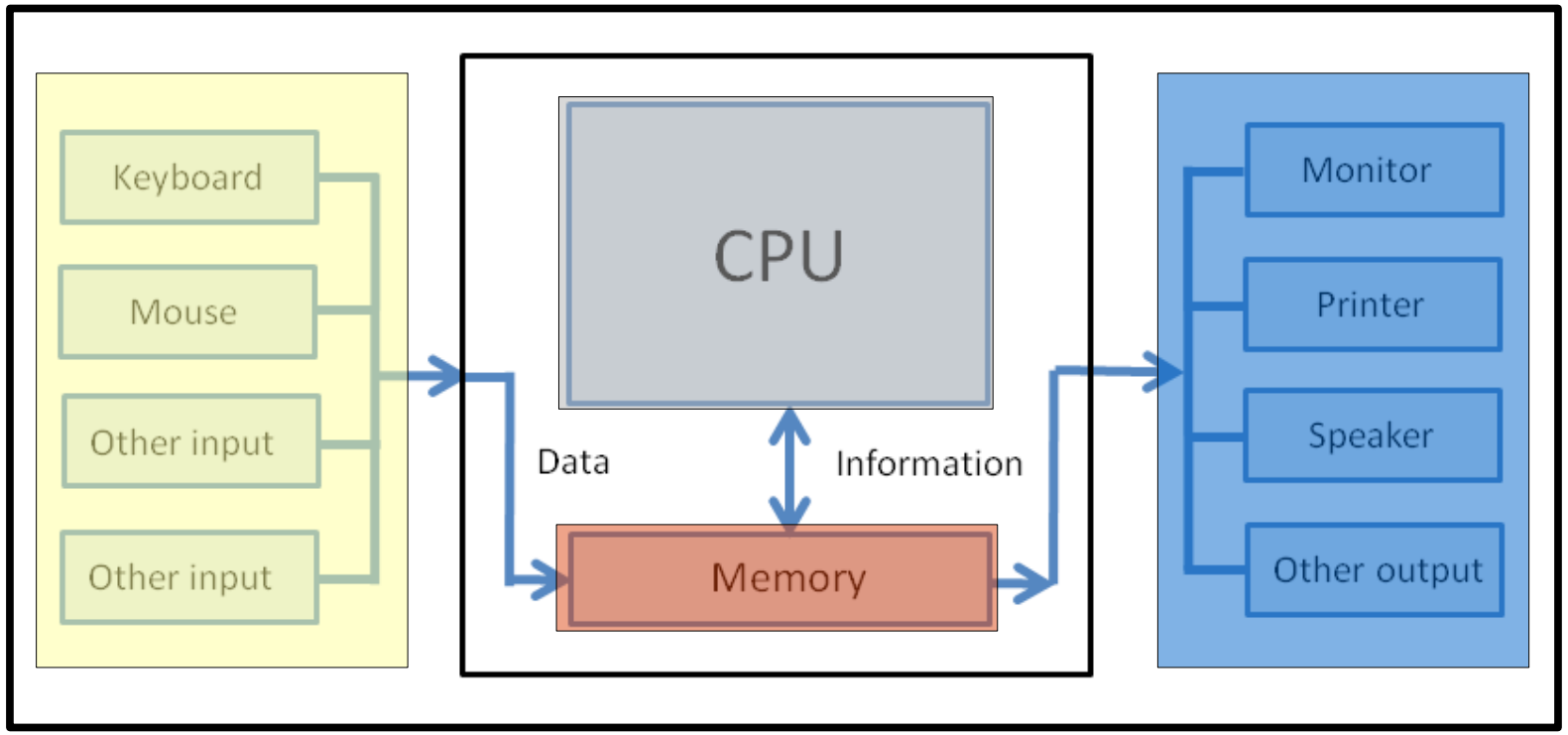
Escrita

x ←

```
var integer x ← 25;
var integer y ← 25;
emit GRAPHICS_DRAW_PIXEL(x,y);
loop do
    var integer key ← await KEY_PRESS;
    if key = KEY_LEFT then
        x ← x - 1;
    else/if key = <...> then
        <...>
    end
    emit GRAPHICS_DRAW_PIXEL(x,y);
end
```

Leitura

x

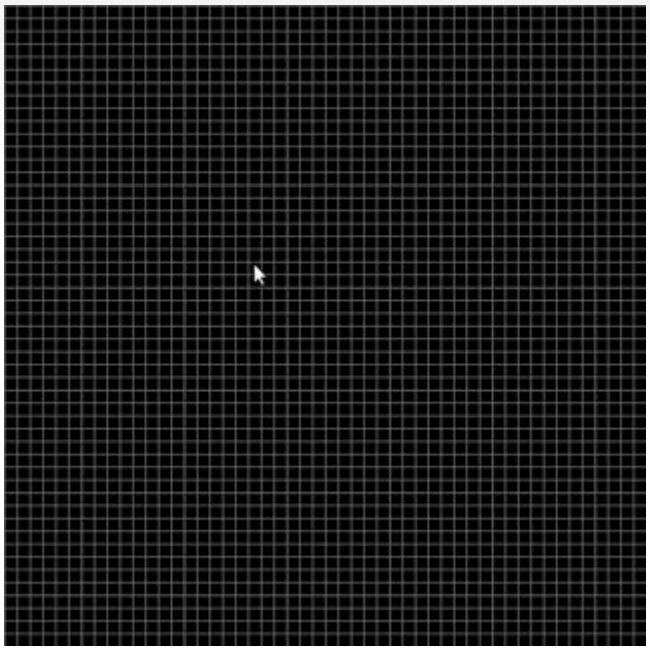


Exercises / Projects

- Add a new feature
  - erase the moving trail
  - use different colors
  - draw other geometric shapes (circles, figures)
  - write text
- Explore the API

Concurrency

- Drawing an **X** on the screen:



Concurrency

```
var integer p;
loop p in [0 -> 50[ do
   emit GRAPHICS_DRAW_PIXEL(p,p);
   emit GRAPHICS_DRAW_PIXEL(49-p,p);
   await 100ms;
end
```

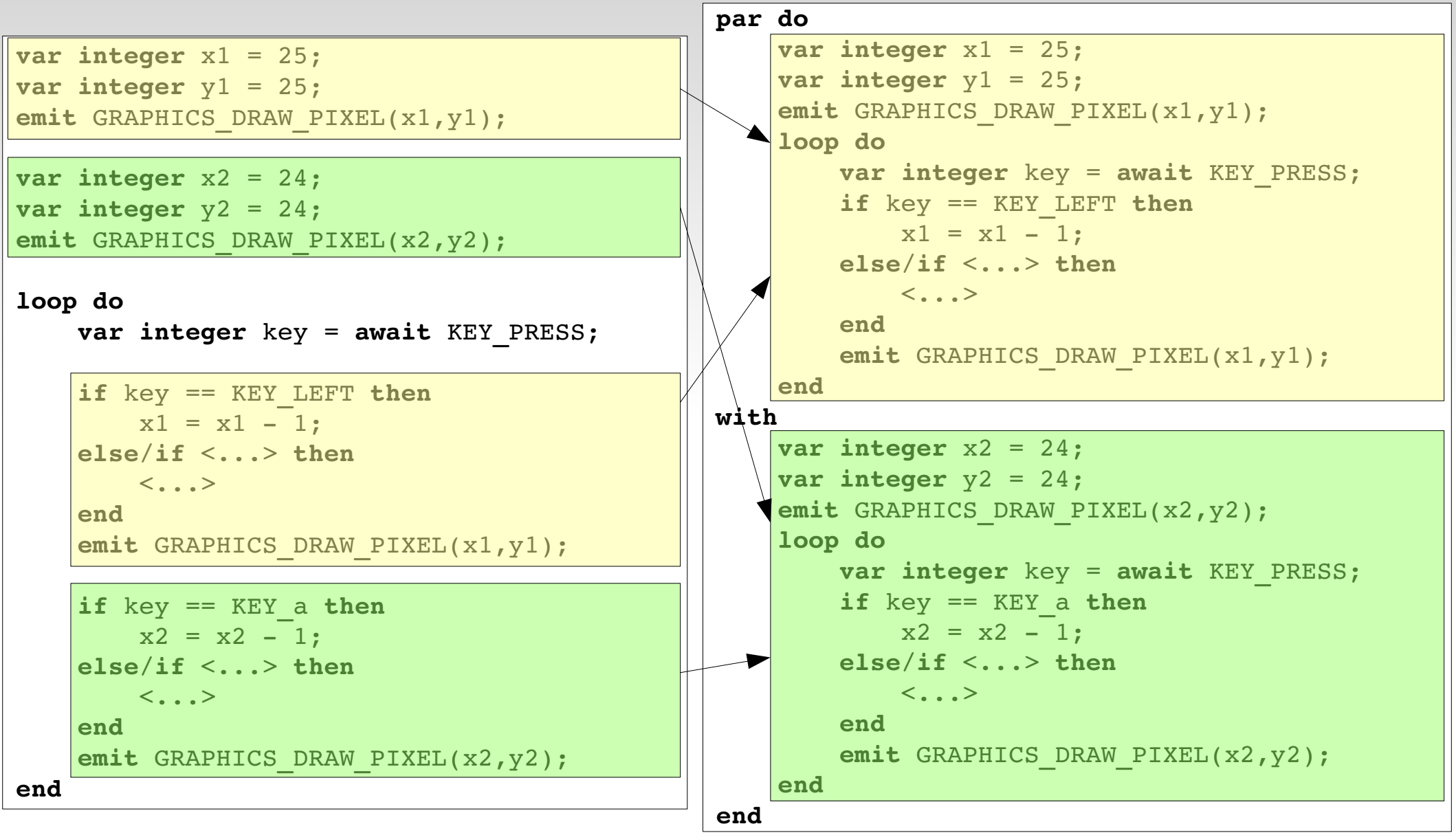
```
par do
   var integer p1;
   loop p1 in [0 -> 50[ do
      emit GRAPHICS_DRAW_PIXEL(p1,p1);
      await 100ms;
   end
with
   var integer p2 = 0;
   loop p2 in [0 -> 50[ do
      emit GRAPHICS_DRAW_PIXEL(49-p2,p2);
      await 100ms;
   end
end
```

Concurrency and Parallelism

- Early approach to the theme
- What means “at the same time”?
  - to the application user
  - to the application programmer
  - to the computer
  - experience, specification, implementation
  - concurrency, logical parallelism, real parallelism

Moving two pixels

- Early approach to the theme
- What means “at the same time"?
  - to the application user
  - to the application programmer
  - to the computer
  - experience, specification, implementation
  - concurrency, logical parallelism, real parallelism


var integer x1 = 25;
var integer y1 = 25;
emit GRAPHICS_DRAW_PIXEL(x1,y1);
var integer x2 = 24;
var integer y2 = 24;
emit GRAPHICS_DRAW_PIXEL(x2,y2);
loop do
var integer key = await KEY_PRESS;
if key == KEY_LEFT then
x1 = x1 - 1;
else/if <...> then
<...>
end
emit GRAPHICS_DRAW_PIXEL(x1,y1);
if key == KEY_a then
x2 = x2 - 1;
else/if <...> then
<...>
end
emit GRAPHICS_DRAW_PIXEL(x2,y2);
end
par do
var integer x1 = 25;
var integer y1 = 25;
emit GRAPHICS_DRAW_PIXEL(x1,y1);
loop do
var integer key = await KEY_PRESS;
if key == KEY_LEFT then
x1 = x1 - 1;
else/if <...> then
<...>
end
emit GRAPHICS_DRAW_PIXEL(x1,y1);
end
with
var integer x2 = 24;
var integer y2 = 24;
emit GRAPHICS_DRAW_PIXEL(x2,y2);
loop do
var integer key = await KEY_PRESS;
if key == KEY_a then
x2 = x2 - 1;
else/if <...> then
<...>
end
emit GRAPHICS_DRAW_PIXEL(x2,y2);
end
end

Exercises / Projects

- Create a simple game or application
  - how to track the moving trails?
  - how to detect collisions?

Structured/Imperative Programming

- Input and Output
- Assignment
- Control Flow
  - Sequence
  - Choice
  - Repetition
  - ***Parallelism (logical)***
- Abstractions
  - Code
  - Data

Abstractions

```
par do
    var integer x1 = 25;
    var integer y1 = 25;
    emit GRAPHICS_DRAW_PIXEL(x1,y1);
    loop do
        var integer key = await KEY_PRESS;
        if key == KEY_LEFT then
            x1 = x1 - 1;
        else/if <...> then
            <...>
        end
        emit GRAPHICS_DRAW_PIXEL(x1,y1);
    end
with
    var integer x2 = 24;
    var integer y2 = 24;
    emit GRAPHICS_DRAW_PIXEL(x2,y2);
    loop do
        var integer key = await KEY_PRESS;
        if key == KEY_a then
            x2 = x2 - 1;
        else/if <...> then
            <...>
        end
        emit GRAPHICS_DRAW_PIXEL(x2,y2);
    end
end
```

```
code Pix (var integer x, y,
          var integer key_left, key_right,
                      key_up, key_down)
              -> FOREVER
do
    emit GRAPHICS_DRAW_PIXEL(x,y);
    loop do
        var integer key = await KEY_PRESS;
        if key == key_left then
            x = x - 1;
        else/if <...> then
            <...>
        end
        emit GRAPHICS_DRAW_PIXEL(x,y);
    end
end

par do
    await Pix(25,25,
              KEY_LEFT,KEY_RIGHT,
              KEY_UP,KEY_DOWN);
with
    await Pix(24,24,
              KEY_a,KEY_d,
              KEY_w,KEY_s);
end
```

Network

- API for unreliable broadcast
  - uses UDP

```
par do
    var integer n;
    var byte&&  buf;
    every (n,buf) in NET_RECEIVE do
        emit GRAPHICS_WRITE("recv: “);
        emit GRAPHICS_WRITELN(buf);
    end
with
    loop do
        await 1s;
        emit NET_SEND(4, "ola");
        emit NET_SEND(6, "mundo");
    end
end
```

Other Subjects

- Classical Problems
  - dinning philosophers
  - shortest path
    - traveling salesman
  - two generals
    - three-way handshake
- “Physical computing”
  - Arduino, RPi
- Céu is “scalable”
  - sufficiently general purpose and integrates with C
  - other domains (SDL, Arduino, libuv, multimedia, WSNs, ...)

Arduino

- Ambiente ainda mais concreto

```
input  onoff PIN_02;
output onoff PIN_13;

emit PIN_13(on);

loop do
    var onoff v = await PIN_02;
    emit PIN_13(v);
end
```

TODO

- Documentação em Português
- Linguagem em Português?
- Debugger