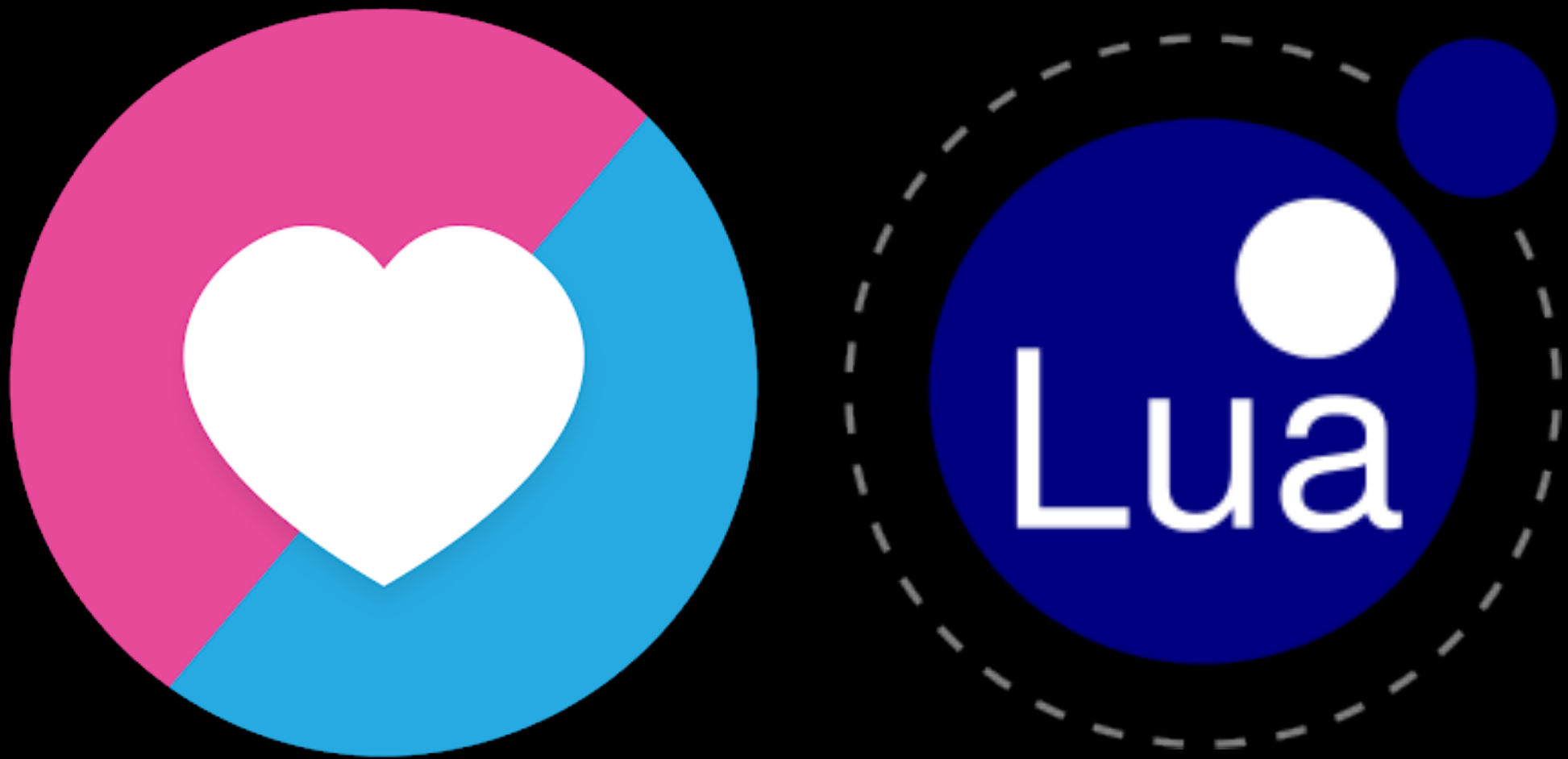
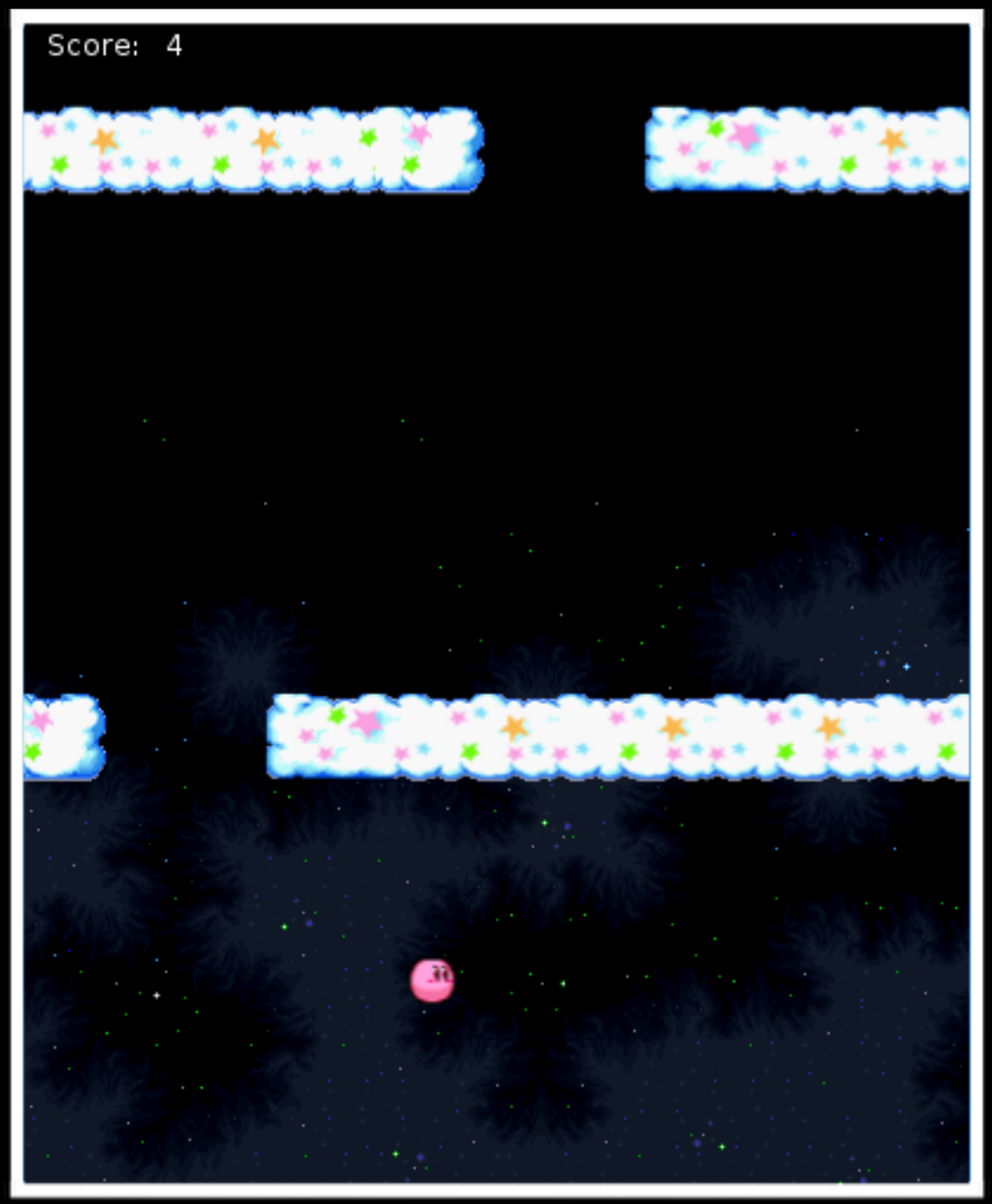


MiniProjeto II - Jogo em LÖVE/Lua

INF1805 - Sistemas Reativos

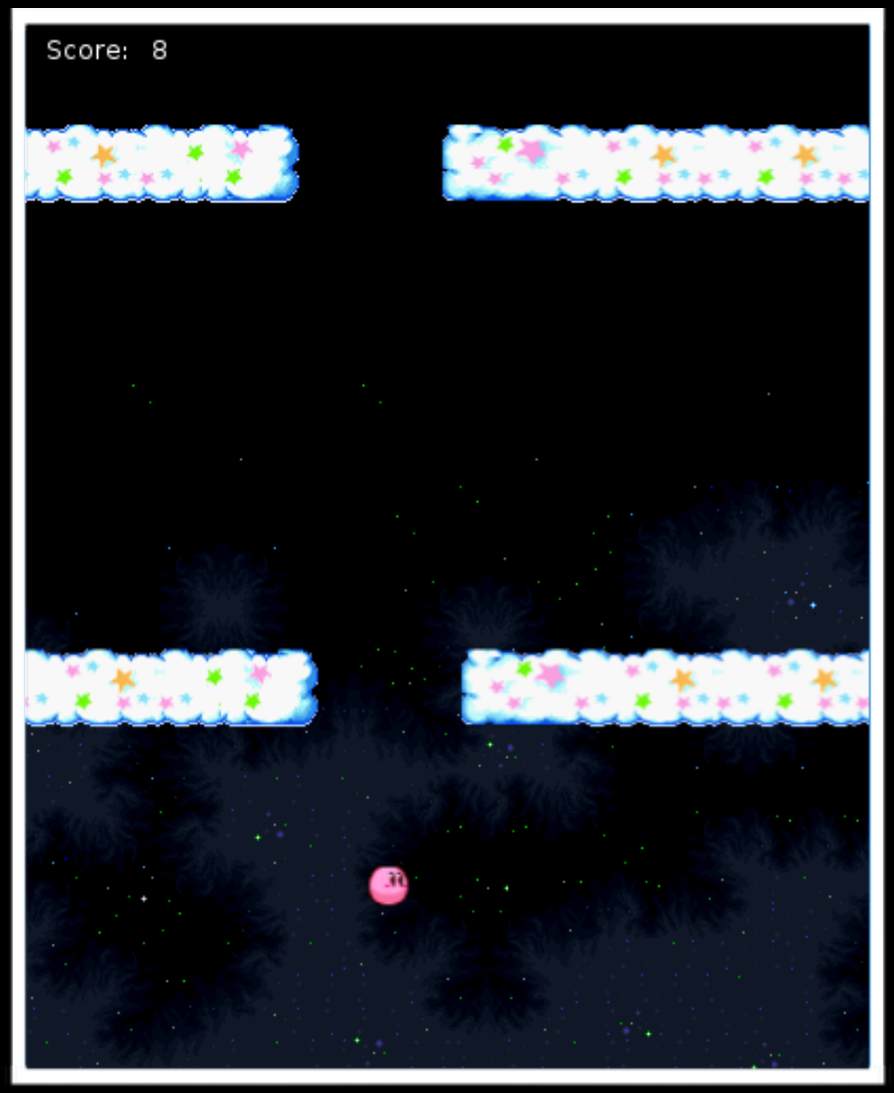
Matheus Cunha
Victor Meira

Objetivo

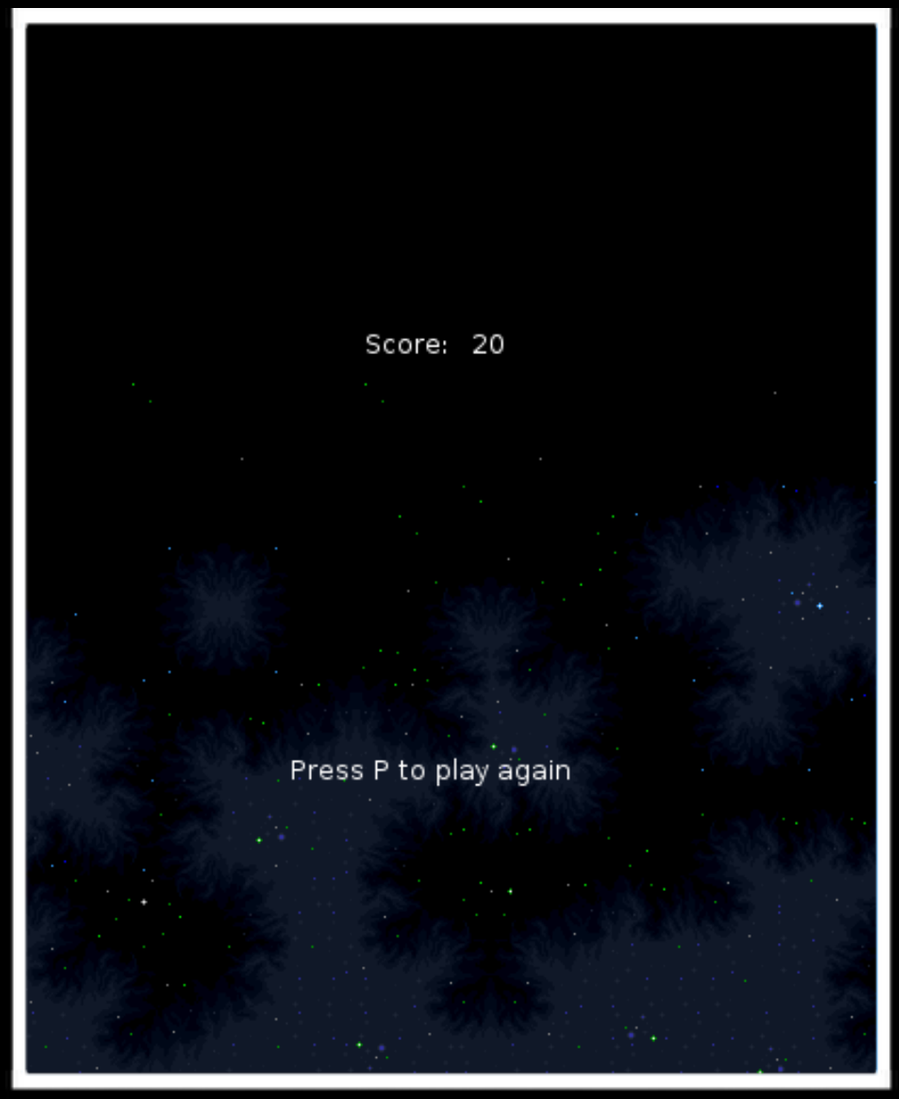


Funcionamento

Movimento Kirby



Funcionamento



Implementação

```
function new_player(x, y)

  return {

    kirby = love.graphics.newImage("kirby.png"),
    posx = x,
    posy = y,
    kirby_width = love.graphics.newImage("kirby.png"):getWidth(),
    kirby_height = love.graphics.newImage("kirby.png"):getHeight(),
    score = 0,

    dtMult_player = 400,

    draw = function (self)
            love.graphics.draw(self.kirby, self.posx, self.posy)
    end,

    ...
```

Implementação

```
...

   keyDown = function (self, dt)
      if love.keyboard.isDown("down") then
        self.posy = self.posy + self.dtMult_player*dt
        if self.posy >= HEIGHT - self.kirby_height then
          self.posy = HEIGHT - self.kirby_height
        end
      end
      if love.keyboard.isDown("up") then
        ...
      end
      if love.keyboard.isDown("right") then
        ...
      end
      if love.keyboard.isDown("left") then
        self.posx = self.posx - self.dtMult_player*dt
        if self.posx <= 0 then
          self.posx = 0
        end
      end
    end
```

Implementação

```
function new_enemy(x, y, w, h)

  return {

  cloud = love.graphics.newImage("enemy.png"),

  spaceBetweenBlocks = WIDTH/6,
  dtMult_enemy = 250,


  posx = x,
  posy = y,
  width = w,
  height = love.graphics.newImage("enemy.png"):getHeight(),
  score_flag = true,

   draw = function (self)
          love.graphics.draw(self.cloud, self.posx - WIDTH + self.width,
self.posy)
          love.graphics.draw(self.cloud, self.posx + self.width +
self.spaceBetweenBlocks, self.posy)
    end,

...
```

Implementação

```
...

   randomWidth = function (self)
      self.width = math.random(0,5*WIDTH/6)
   end,

   update = function (self, dt)

      self.posy = self.posy + self.dtMult_enemy*dt

      if self.posy >= HEIGHT then
        self:randomWidth()
        self.posy = 0 - self.height
        self.score_flag = true
      end

    end,


...
```

Implementação

```
...

checkScoreOrCollision = function (self, _player)

      --score
      if self.posy - self.height >=  _player.posy - 2*_player.kirby_width then
        if self.score_flag == true then
          _player.score = _player.score + 1
          Walldt = Walldt + 0.001
          self.score_flag = false
        end

      --collision
      elseif (_player.posy <= self.posy + self.height) and (_player.posx <=
self.posx + self.width or _player.posx >= self.posx + self.width +
self.spaceBetweenBlocks - _player.kirby_width ) then
        screen = 1
      end
    end
```

Implementação

```
function love.update(dt)

  if screen == 0 then
    player:keyDown(dt)

    for i =1, #enemy_array do
      enemy_array[i]:update(Walldt)
      enemy_array[i]:checkScoreOrCollision(player)
    end
  end
end
```

Implementação

```
function love.draw()

  love.graphics.draw(background, 0, 0)

  if screen == 0 then
    player:draw()
    for i = 1, #enemy_array do
      enemy_array[i]:draw()
    end
    love.graphics.print("Score: ", 10, 10)
    love.graphics.print(tostring(player.score), 60, 10)

  elseif screen == 1 then
    love.graphics.print("Score: ", WIDTH/2 -40 , HEIGHT/2 - 100)
    love.graphics.print(tostring(player.score), WIDTH/2 + 10, HEIGHT/2 - 100)
    love.graphics.print("Press P to play again", WIDTH/2 - 75 , HEIGHT/2 + 100)
  end

end
```

Principais Dificuldades

- Espaço entre as paredes só aparecia na metade esquerda da tela
- Implementação do coroutine.wrap( )
- Implementação de inimigos
- Implementação da destruição das paredes