Pontifícia Universidade Católica
do Rio de Janeiro

ENG1000 – Introdução à Engenharia

Aula 06 – Estruturas Condicionais e Interação

Edirlei Soares de Lima
<elima@inf.puc-rio.br>

visionlab
visualization laboratory

Tomada de Decisão

- Até o momento, todas as instruções dos nossos programas eram executadas sequencialmente.
  - Mesmo usando funções elas ainda eram executadas na ordem em que foram codificadas.

- Em geral, precisamos ter maior controle na sequência de instruções que devem ser executadas.

- É fundamental que seja possível tomar diferentes decisões baseado em **condições** que são avaliadas em **tempo de execução.**

Estruturas Condicionais

- **Estruturas condicionais** permitem a criação de programas que não são totalmente sequenciais.

- Com o uso de estruturas condicionais é possível criar **regras** que definem quando uma determinada parte do código deve ser executada.

Estruturas Condicionais

- Em Lua, a tomada de decisão é construída através do comando ***if***:

```
if expressão_lógica then
   -- bloco de comandos
end
```

- Exemplo:

```
if nota < 5.0 then
   io.write("Reprovado")
end
```

Os comandos do **bloco de comandos** somente são executados se a **expressão lógica** for verdadeira

Estruturas Condicionais

- Também é possível usar o comando **else** para executar algo quando a expressão lógica não é verdadeira:

```
if expressão_lógica then
   -- Bloco de comandos
else
   -- Bloco de comandos
end
```

Exemplo:

```
if nota < 5.0 then
   io.write("Reprovado")
else
   io.write("Aprovado")
end
```

Estruturas Condicionais

- Também é possível criar sequencias de comandos **if-else** para a verificação exclusiva de varias condições:

```
if  condição_1 then
   -- Bloco de comandos 1
elseif condição_2 then
   -- Bloco de comandos 2
elseif condição_3 then
   -- Bloco de comandos 3
end
```

Se a primeira condição resultar em *verdadeiro*, apenas o primeiro bloco de comandos é executado, e as outras condições não são sequer avaliadas.
Senão, se a segunda condição resultar em *verdadeiro*, apenas o segundo bloco de comandos é executado, e assim por diante.

Estruturas Condicionais

- **Exemplo:**

```
if nota < 3.0 then
  io.write("Reprovado")
elseif nota >= 5.0 then
  io.write("Aprovado")
else
  io.write("Em prova final")
end
```

Expressões Booleanas

- Uma expressão booleana é construída através da utilização de **operadores relacionais:**

Descrição	Símbolo
Igual a	==
Diferente de	~=
Maior que	>
Menor que	<
Maior ou igual a	>=
Menor ou igual a	<=

Exemplos:

X = 10 e Y = 5

Expressão	Resultado
`X == Y`	Falso
`X ~= Y`	Verdadeiro
`X > Y`	Verdadeiro
`X < Y`	Falso
`X >= Y`	Verdadeiro
`X <= Y`	Falso

Todos estes operadores comparam **dois operandos**, resultando no valor 0 (falso) ou 1 (verdadeiro).

Expressões Booleanas

- Expressões booleanas também podem ser combinadas através de **operadores lógicos**.

Operador	Significado	Símbolo em C
Conjunção	E	and
Disjunção	OU	or
Negação	NÃO	not

Exemplos:

Expressão	Resultado
`X > 0 and X == Y`	Falso
`X > 0 or X == Y`	Verdadeiro
`not Y < 10`	Falso

X = 10
Y = 5

Expressões Booleanas

- Operadores lógicos combinam expressões ou valores booleanos, resultando em um valor booleano.

Conjunção (and)

Operando 1	Operando 2	Resultado
Falso	Falso	Falso
Falso	Verdadeiro	Falso
Verdadeiro	Falso	Falso
Verdadeiro	Verdadeiro	Verdadeiro

Disjunção (or)

Operando 1	Operando 2	Resultado
Falso	Falso	Falso
Falso	Verdadeiro	Verdadeiro
Verdadeiro	Falso	Verdadeiro
Verdadeiro	Verdadeiro	Verdadeiro

Negação (not)

Operando	Resultado
Falso	Verdadeiro
Verdadeiro	Falso

Expressões Booleanas

- Exemplo 1 (and):

```
...
if media >= 5.0 and nota1 >= 3.0 and nota2 >=3.0 and nota3 >= 3.0 then
  io.write("Aprovado")
end
...
```

- Exemplo 2 (or):

```
...
if media < 5.0 or nota1 < 3.0 or nota2 < 3.0 or nota3 < 3.0 then
  io.write("Em prova final")
end
...
```

- Exemplo 3 (not):

```
...
if not (media < 5.0 or nota1 < 3.0 or nota2 < 3.0 or nota3 < 3.0) then
  io.write("Aprovado")
end
...
```

Estruturas Condicionais - Exemplo

- Crie um programa que converta a nota de um aluno (que varia de 0 a 10) para um conceito (A, B, C, D, ou F). Assuma a seguinte equivalência entre a nota e o conceito:
  - A (9.0 a 10.0)
  - B (8.0 a 8.9)
  - C (7.0 a 7.9)
  - D (5.0 a 6.9)
  - F (menor que 5.0)

Estruturas Condicionais - Exemplo

```
local nota

io.write("Digite a nota: ")
nota = tonumber(io.read())

if nota >= 9.0 then
  io.write("A")
elseif nota >= 8.0 and nota < 9.0 then
  io.write("B")
elseif nota >= 7.0 and nota < 8.0 then
  io.write("C")
elseif nota >= 5.0 and nota < 7.0 then
  io.write("D")
else
  io.write("F")
end
```

De Volta ao “Hello World”

- Na ultima implementação do “Hello World” fizemos o texto se mover na tela
  - **Problema:** quando a posição do texto passava do limite da tela, o texto sumia (continuava se movendo para longe da tela)

- Com uma estrutura condicional podemos fazer o texto voltar para o inicio da tela.

- Como podemos fazer isso?

De Volta ao “Hello World”

```
local px    -- posição x do texto

function love.load()
  love.graphics.setColor(0, 0, 0)
  love.graphics.setBackgroundColor(255, 255, 255)
  px = 0
end

function love.update(dt)
  px = px + (100 * dt)
end

function love.draw()
  love.graphics.print("Hello World", px, 300)
end
```

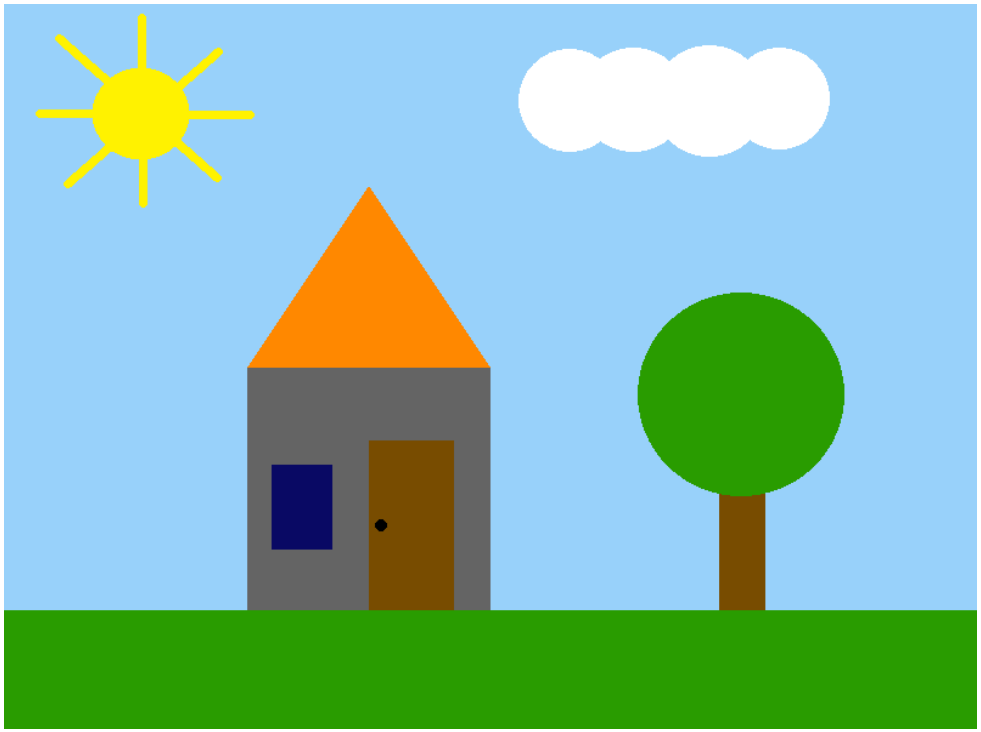
```
local px    -- posição x do texto

function love.load()
  love.graphics.setColor(0, 0, 0)
  love.graphics.setBackgroundColor(255, 255, 255)
  px = 0
end

function love.update(dt)
  px = px + (100 * dt)

  if px > 800 then -- a largura da janela é 800
    px = 0
  end

end

function love.draw()
    love.graphics.print("Hello World", px, 300)
end
```

```
local px    -- posição x do texto

function love.load()
  love.graphics.setColor(0, 0, 0)
  love.graphics.setBackgroundColor(255, 255, 255)
  px = 0
end

function love.update(dt)
  px = px + (100 * dt)

  if px > love.window.getWidth() then
    px = 0
  end

end

function love.draw()
  love.graphics.print("Hello World", px, 300)
end
```

Uma forma mais geral de acessar a largura da janela.

Exercício 1

1) Estenda o programa do exercício da aula anterior criando uma animação que faça com que a nuvem se mova de um lado para o outro da tela. Se a nuvem estiver tampando o sol, então a cor de fundo do cenário deve ficar mais escura.



Módulo `love.keyboard`

- O módulo `love.keyboard` contem funções dedicadas a interação pelo teclado.

- É possível verificar se um determinada tecla foi pressionada usando o comando `love.keyboard.isDown`

```
love.keyboard.isDown(key)
```

- A função retorna verdadeiro se a tecla passada como parâmetro estiver pressionada.

Módulo `love.keyboard`

- É necessário utilizar uma **estrutura condicional** para verificar se a tecla foi pressionada.

- **Exemplo:**

```
if love.keyboard.isDown("right") then
  px = px + (100 * dt)
end
```

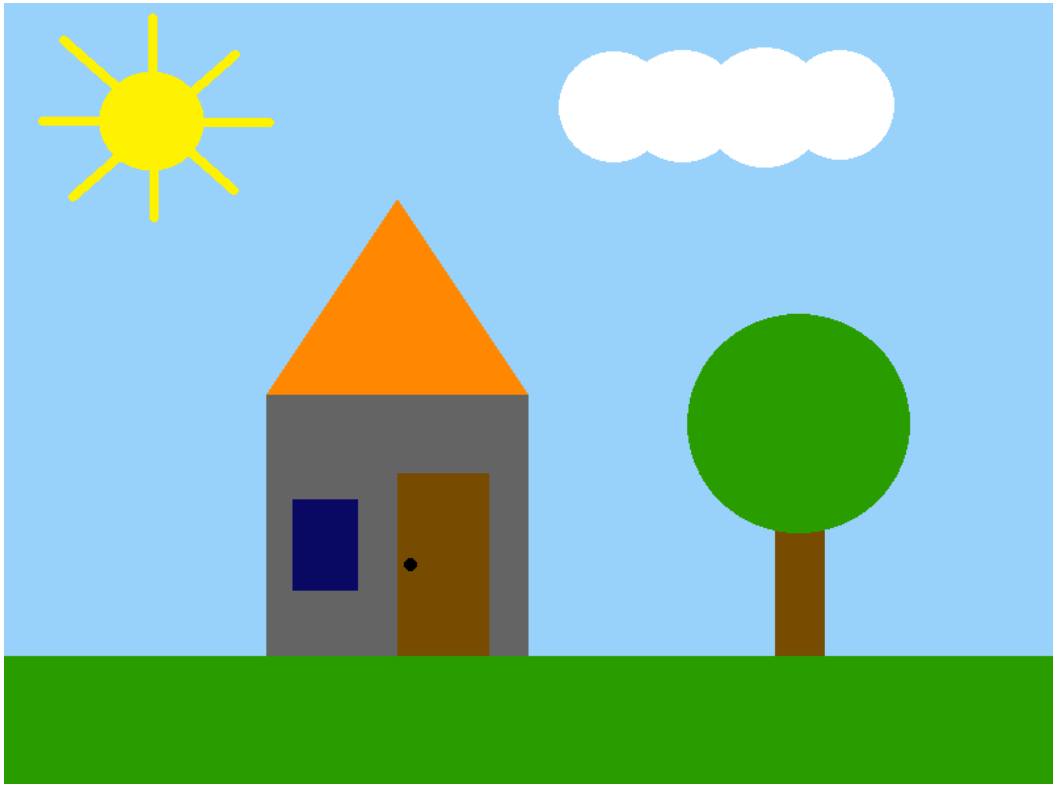
- Lista de teclas: http://www.love2d.org/wiki/KeyConstant

De Volta ao “Hello World”

```
local px    -- posição x do texto

function love.load()
  love.graphics.setColor(0, 0, 0)
  love.graphics.setBackgroundColor(255, 255, 255)
  px = 0
end

function love.update(dt)

  if love.keyboard.isDown("right") then
    px = px + (100 * dt)
  end

end

function love.draw()
  love.graphics.print("Hello World", px, 300)
end
```

Exercício 2

2) Estenda o programa do exercício anterior para permitir que o usuário possa movimentar o sol no eixo X e Y usando as setas direcionais do teclado.



Módulo `love.mouse`

- O módulo `love.mouse` contem funções dedicadas a interação pelo mouse.

- É possível verificar se um determinado botão do mouse foi pressionada usando o comando `love.mouse.isDown`

```
love.mouse.isDown(button)
```

- A função retorna verdadeiro se o botão passada como parâmetro estiver pressionado.

Módulo `love.mouse`

- É necessário utilizar uma **estrutura condicional** para verificar se o botão foi pressionado.

- **Exemplo:**

```
if love.mouse.isDown("l") then
  texto = "Left! :)"
end
```

- Lista de teclas: http://www.love2d.org/wiki/MouseConstant

Módulo `love.mouse`

- O módulo `love.mouse` também permite acessa a posição do através dos comandos `love.mouse.getX` e `love.mouse.getY`

```
love.mouse.getX()
```

```
love.mouse.getY()
```

- As funções retornam a posição do mouse relativa a janela do programa nos eixos X e Y.

```
mousex = love.mouse.getX()
mousey = love.mouse.getY()
```

De Volta ao “Hello World”

- Utilizando as funções de interação pelo mouse, podemos modificar o “Hello World” para permitir que:
  - O usuário possa movimentar o texto com o mouse;
  - O texto seja modificado quando o usuário clicar com o mouse:
    - Botão da direita: “Right! :)”
    - Botão da esquerda: “Left! :)”
    - Nenhum botão: “Hello World!”
- Como podemos fazer isso?

De Volta ao “Hello World”

```
local px    -- posição x do texto
local py    -- posição y do texto
local texto = "Hello World!"

function love.update(dt)
  if love.mouse.isDown("l") then
    texto = "Left! :)"
  elseif love.mouse.isDown("r") then
    texto = "Right! :)"
  else
    texto = "Hello World!"
  end
  px = love.mouse.getX()
  py = love.mouse.getY()
end

function love.draw()
  love.graphics.print(texto, px, py)
end
```

Exercício 3

3) Modifique o programa do exercício anterior para permitir que o usuário possa movimentar o sol utilizando o mouse.