

# **FURBOT-WEB: UMA PLATAFORMA ADAPTATIVA PARA O ENSINO DE PROGRAMAÇÃO**

Aluna: Heloisa Kaestner Kopsch

Orientador: Maurício Capobianco Lopes

# Roteiro

- Introdução
- Objetivos
- Fundamentação Teórica
- Trabalhos Correlatos
- Requisitos
- Especificação
- Implementação
- Operacionalidade da Implementação
- Resultados e Discussões
- Conclusões e Sugestões

# Introdução

- Contribuições no ensino e aprendizagem de programação
- Automatização do processo de aplicação de exercícios
- Acompanhamento de alunos que não evoluem nos conteúdos
- Aprendizagem personalizada, atrativa e com potencial de desenvolver melhor suas habilidades

# Objetivos

O objetivo deste trabalho é desenvolver uma plataforma web adaptativa para o ensino de programação, tendo como base o *framework* FURBOT.

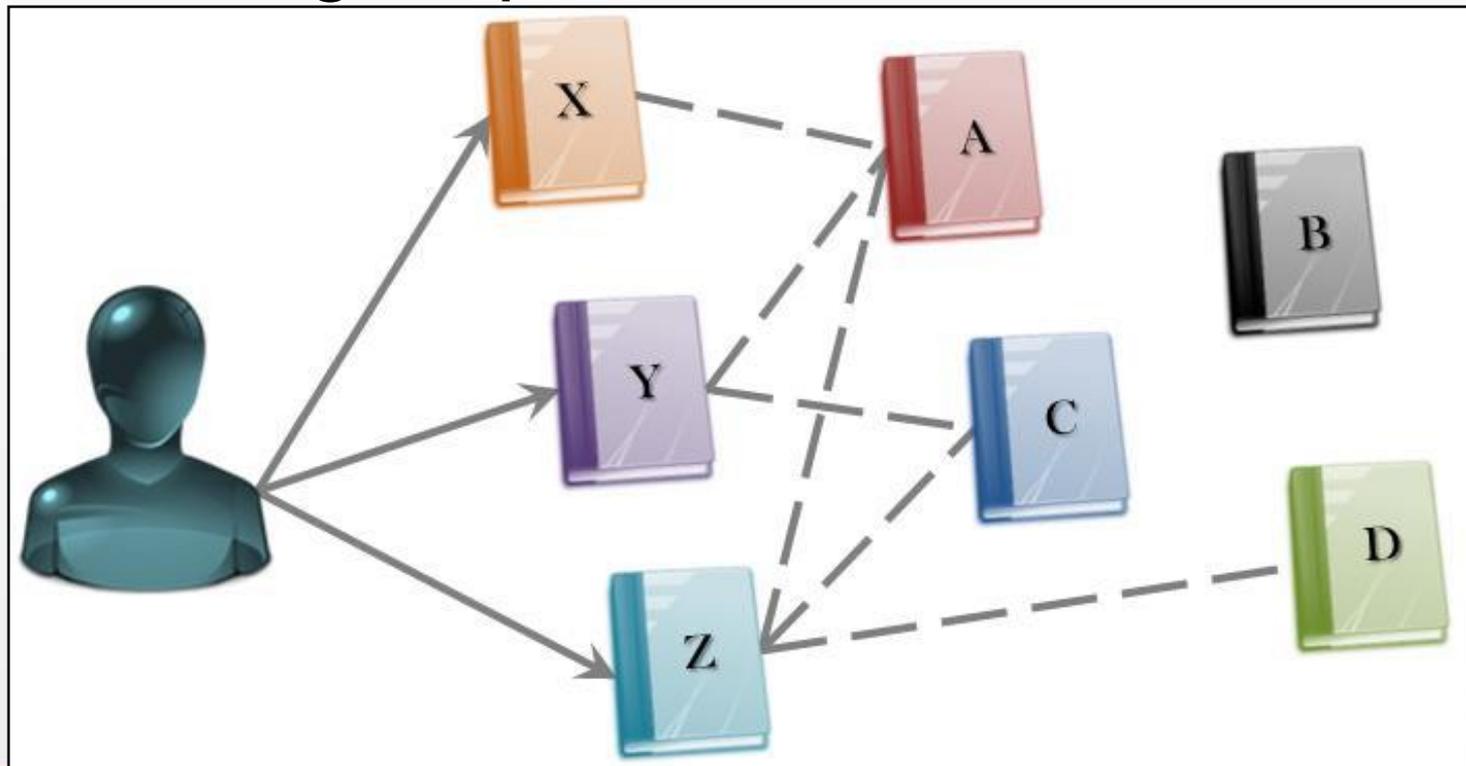
# Objetivos

Os objetivos específicos do trabalho são:

- disponibilizar o FURBOT em um ambiente web;
- criar uma plataforma personalizada de ensino para o FURBOT-WEB;
- analisar a usabilidade da plataforma desenvolvida.

# Fundamentação Teórica

- Personalização do Ensino
  - Sistemas de Recomendação
    - Filtragem por Conteúdo



# Fundamentação Teórica

- FURBOT
- Ensino de Lógica de Programação

Mundo do Furbot v 1.6

a) Faça o furbot andar ate a ultima posicao da linha  
OBSERVAÇÕES: 1  
-Lembre-se de que as coordenadas seguintes serao fornecidas como (x,y)  
-A primeira coluna e linha possuem valor zero.

3 Run 4 New 5 Stop

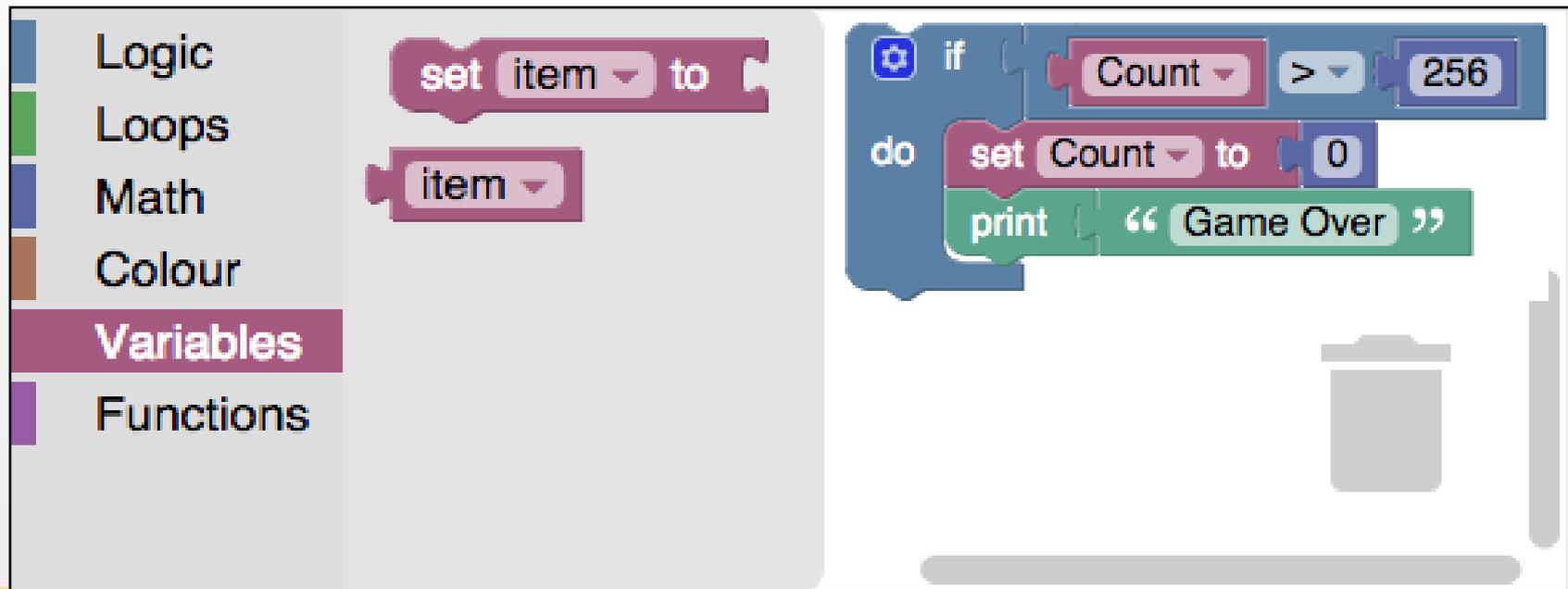
Velocidade : 6

2

7

# Fundamentação Teórica

- **BLOCKLY**
  - Biblioteca JS da Google
    - Programação com Blocos
      - Blocos customizados



The image shows a screenshot of the Blockly programming environment. On the left, there is a vertical menu with the following categories: Logic, Loops, Math, Colour, Variables (highlighted in purple), and Functions. In the center, there are two purple blocks: a 'set item to' block and an 'item' block. On the right, there is a blue 'if' block with a gear icon, containing a 'Count > 256' condition. Below the 'if' block is a 'do' block containing two sub-blocks: a purple 'set Count to 0' block and a green 'print "Game Over"' block. A trash can icon is visible at the bottom right of the workspace.

# Fundamentação Teórica

- **TECEDU**
  - Motor JS para renderização gráfica
    - Componentes

```
<script type="text/javascript">
  function buildGame() {
    Game.loadAPI();

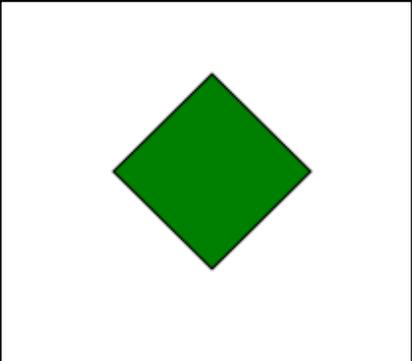
    var quadrado = new PolygonObject().initialize(500, 300, 0,
    [new Point2D().initialize(-50, 0),
    new Point2D().initialize(0, -50),
    new Point2D().initialize(50, 0),
    new Point2D().initialize(0, 50)], null,
    "green", "black");

    ComponentUtils.addComponent(quadrado, new RigidBodyComponent().initialize(0,1,false,false,0.2));

    var layer = new Layer().initialize();
    layer.setGravity(0);
    layer.addGameObject(quadrado);

    var scene = new Scene().initialize(-4000, -4000, 4000, 4000);
    scene.addLayer(layer);

    Game.init(document.getElementById("gameCanvas"), scene);
  }
</script>
```



# Trabalhos Correlatos

- **Scratch**
  - Programação de histórias iterativas, jogos e animações com blocos



# Trabalhos Correlatos

- Programação com Anna e Elsa
  - Programação com blocos utilizando conceitos de sequência e repetição



The screenshot displays a programming interface with a character on a blue background. The interface includes a 'Blocos' (Blocks) area on the left and an 'Área de trabalho: 2 / <> Mostrar código' (Workspace: 2 / Show code) area on the right. The 'Blocos' area contains three blocks: 'avance por 100 pixels', 'vire à direita por 90 graus', and 'vire à esquerda por 90 graus'. The 'Área de trabalho' area contains a 'quando executar' (when executed) block with an 'avance por 100 pixels' block attached to it. Below the workspace, there is a 'Recomeçar' (Restart) button, a progress bar, and a character icon with the text 'Oi! Eu sou Elsa de Arendelle. Ajude-me a criar uma reta.' (Hi! I am Elsa from Arendelle. Help me create a straight line.)

# Trabalhos Correlatos

- **Persona-Algo**

- Sistema web que recomenda exercícios de algoritmos de acordo com o nível de conhecimento do aluno
- Filtragem colaborativa e aspecto afetivo

$$dif = \frac{(num\_exercícios * dificuldade) + dif\_exerc\_atual}{(num\_exercícios + 1)}$$

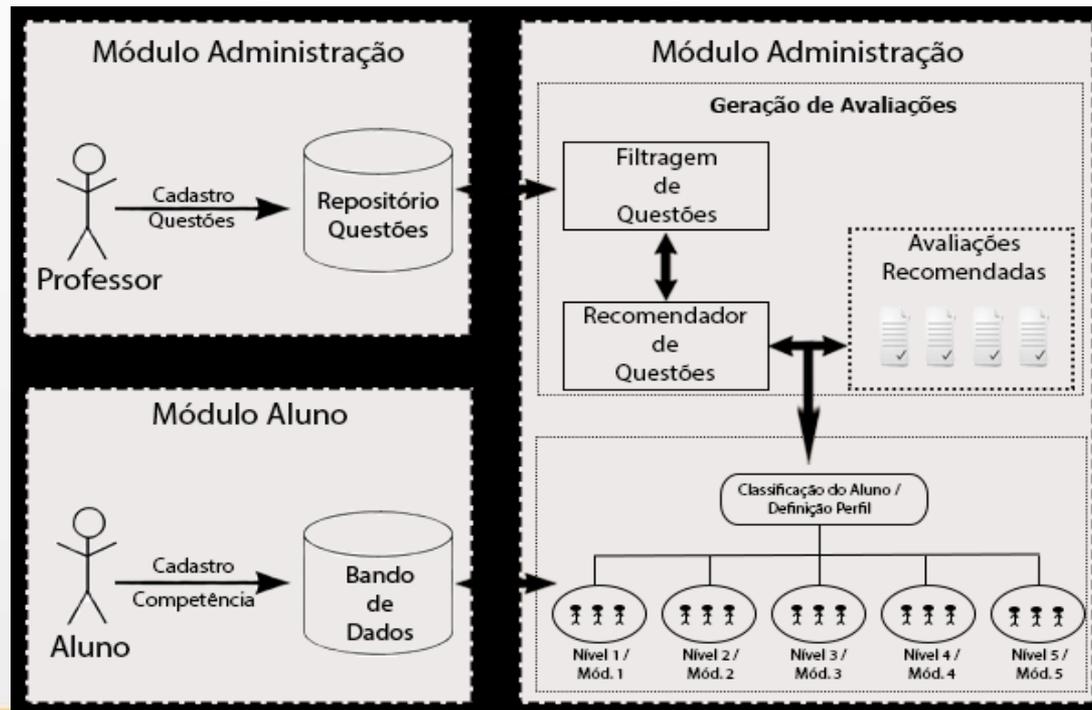
Onde:

- *num\_exercícios*: corresponde ao número de exercícios realizados pelo aluno;
- *dificuldade*: é o grau médio de dificuldade dos exercícios já realizados,
- *dif\_exerc\_atual*: é o grau de dificuldade informado pelo aluno com o exercício em questão.

# Trabalhos Correlatos

- **WebSQL**

- AVA que recomenda exercícios de SQL de acordo com a competência dos alunos
- Forma explícita e implícita



# Requisitos

O sistema web deve:

- permitir que usuários solicitem acesso ao sistema com o perfil de professor para o administrador (RF);
- sugerir exercícios para o aluno através do sistema de recomendação (RF);
- sugerir e-mails de colegas quando o aluno estiver com dificuldade em determinado conteúdo (RF);

# Requisitos

Permitir que o professor:

- cadastre e altere perguntas e respostas para os *quizzes* iniciais (RF);
- cadastre turmas e convide alunos para fazerem parte delas (RF);
- cadastre e altere exercícios (RF);
- crie conteúdos com exercícios associados e *links* para materiais complementares (RF);

# Requisitos

- tenha ciência da quantidade de exercícios pendentes de correção na tela inicial (RF);
- efetue a correção dos exercícios realizados pelos alunos (RF);
- acompanhe os exercícios realizados pelos alunos (RF);

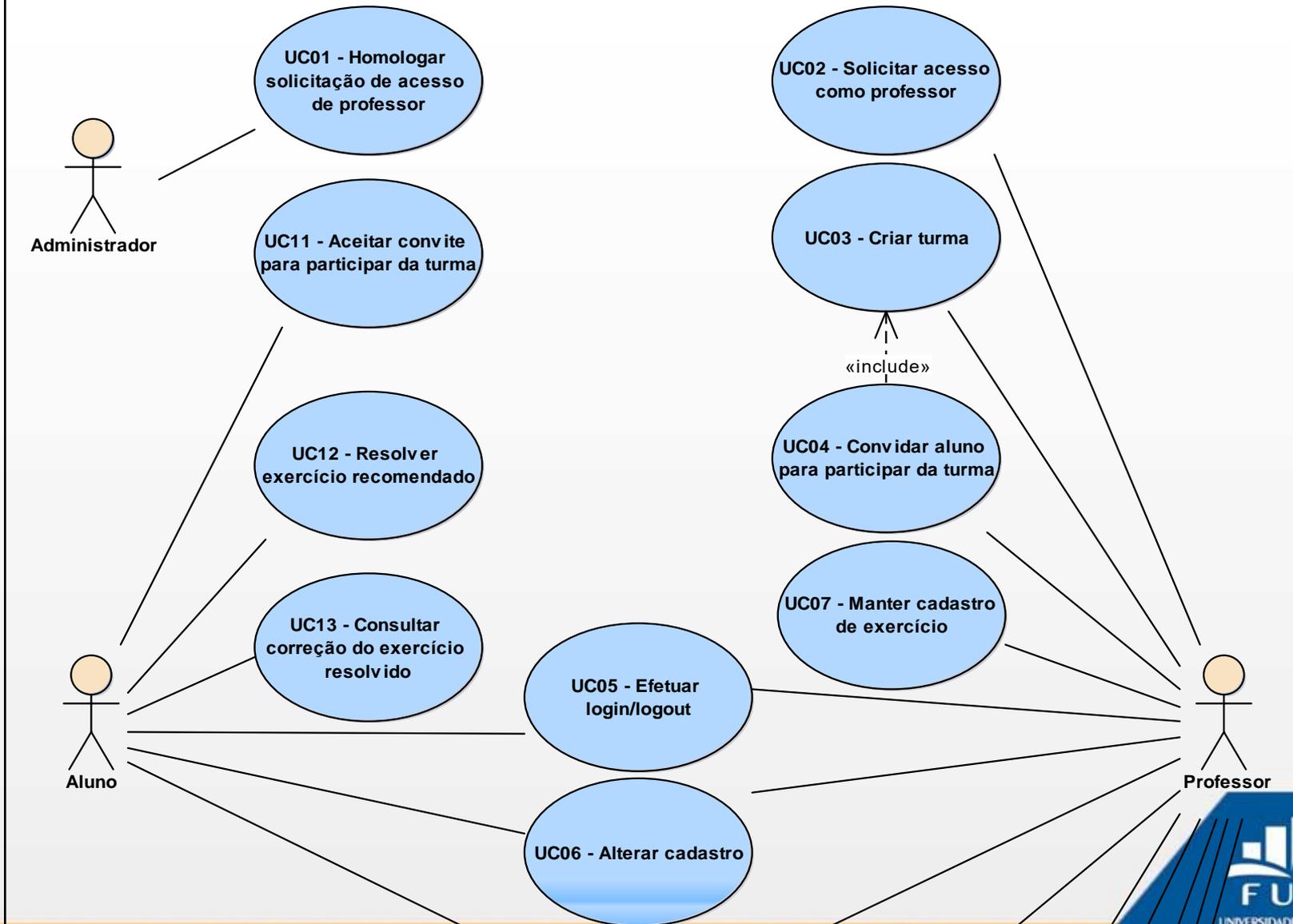
# Requisitos

Permitir que o aluno:

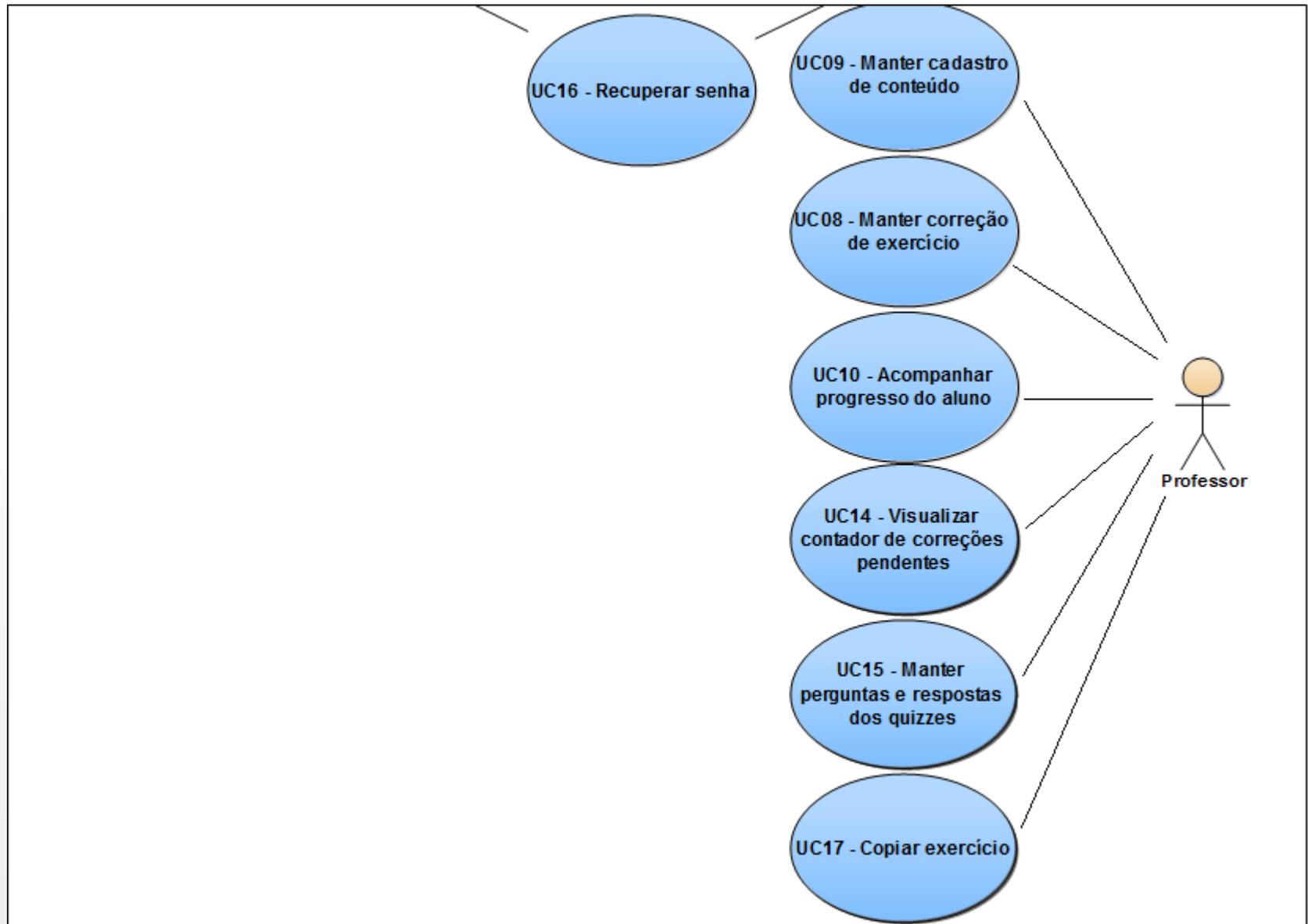
- responda um quiz no seu primeiro acesso ao sistema web (RF);
- resolva os exercícios recomendados, utilizando blocos de comandos arrastáveis baseados nos comandos do FURBOT (RF);
- responda um questionário autoavaliativo ao final de cada exercício (RF);

# Especificação

uc Diagrama de casos de uso do FURBOT- WEB

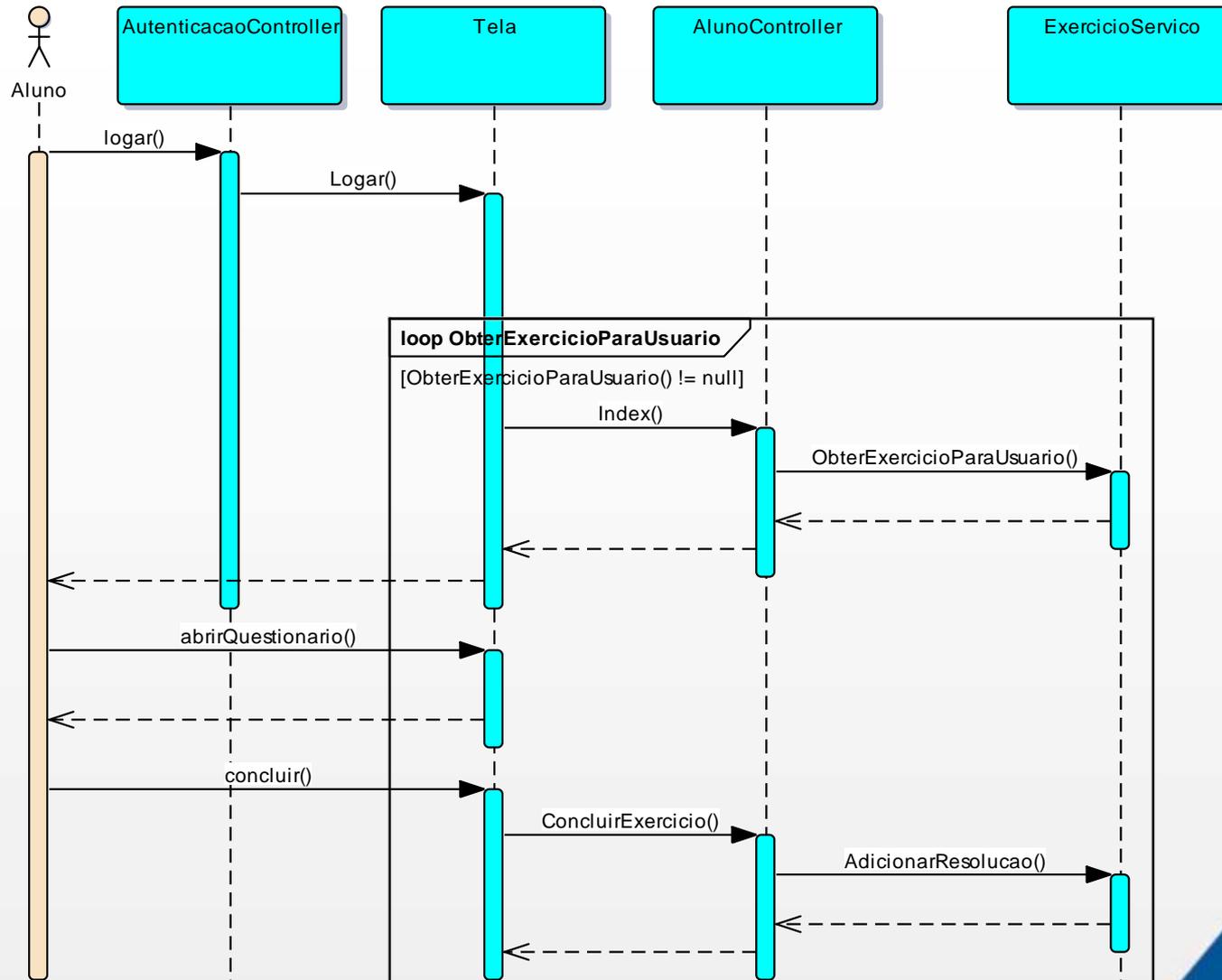


# Especificação



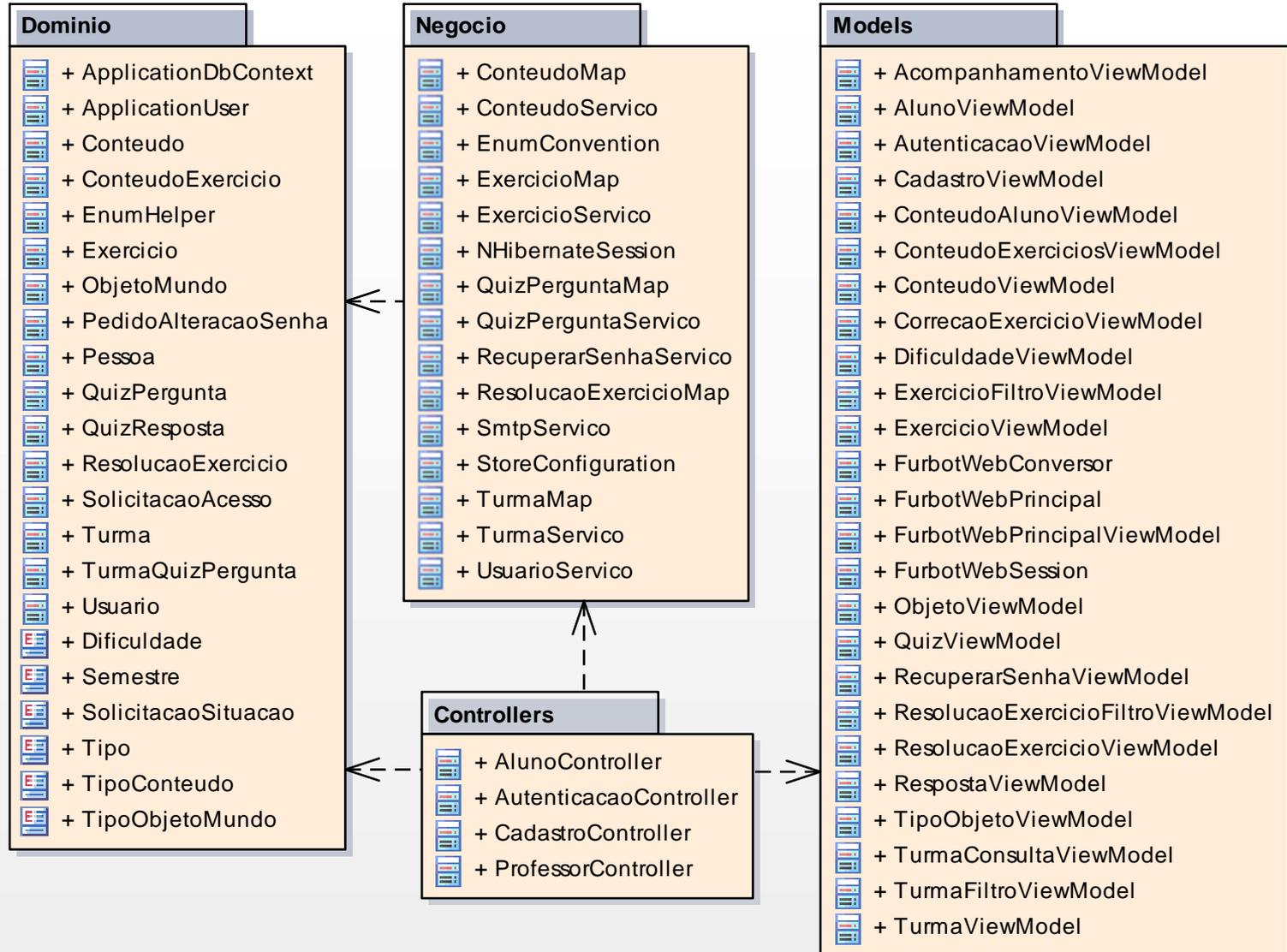
# Especificação

sd FURBOT-WEB Sequence Model

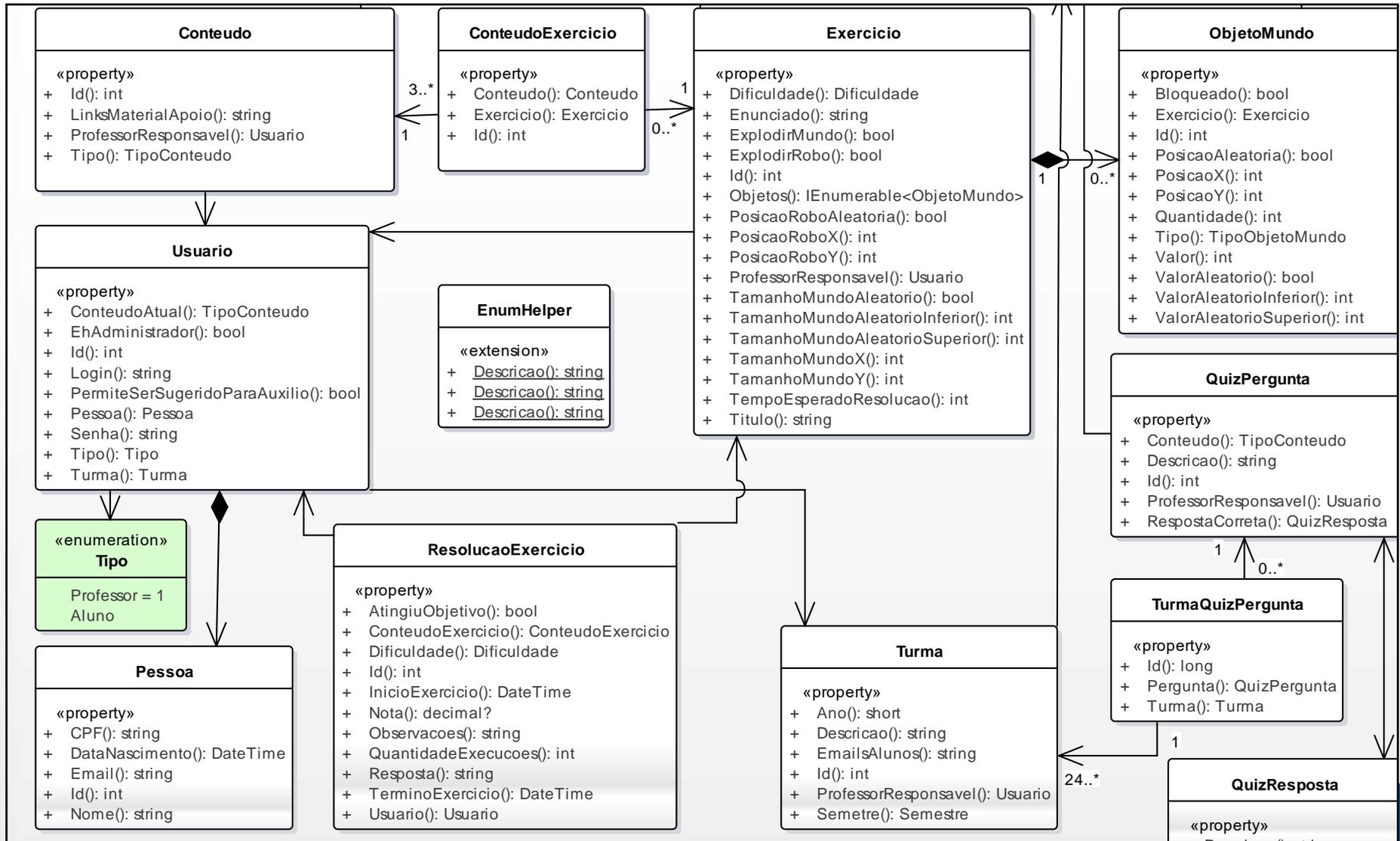


# Especificação

pkg FurbotWeb



# Especificação



# Especificação

## class Negocio

### ExercicioServico

- + Adicionar(Exercicio): void
- + AdicionarResolucao(ResolucaoExercicio): void
- + Atualizar(Exercicio): void
- + Atualizar(ResolucaoExercicio): void
- CalcularPontuacao(IQueryable<ResolucaoExercicio>, int): int
- + ClassificarAlunoNoProximoConteudo(IQueryable<ResolucaoExercicio>): bool
- + Excluir(int): void
- + Obter(int): Exercicio
- + Obter(int, string): List<Exercicio>
- + ObterConteudoExercicio(int): ConteudoExercicio
- + ObterConteudosExercicioParaUsuario(int, string, string): List<ConteudoExercicio>
- + ObterExercicioConteudo(int): ConteudoExercicio
- + ObterExercicioParaUsuario(int): ConteudoExercicio
- + ObterExerciciosCorrigidos(int, string, string, string): List<ResolucaoExercicio>
- + ObterExerciciosModal(string, bool, int): List<Exercicio>
- + ObterExerciciosNaoCorrigidos(int, string, string, string): List<ResolucaoExercicio>
- + ObterExerciciosResolvidos(int): List<ResolucaoExercicio>
- + ObterPerguntasModal(string, bool, int): List<QuizPergunta>
- + ObterQtdResolucoesPorConteudo(TipoConteudo): int
- + ObterQtdResolucoesPorConteudo(int, TipoConteudo): int
- + ObterQuantidadeExerciciosNaoCorrigidos(int): int
- + ObterResolucaoExercicio(long): ResolucaoExercicio
- + ObterTodos(): List<Exercicio>
- + VerificaSeEstaEmUso(int): bool

### TurmaServico

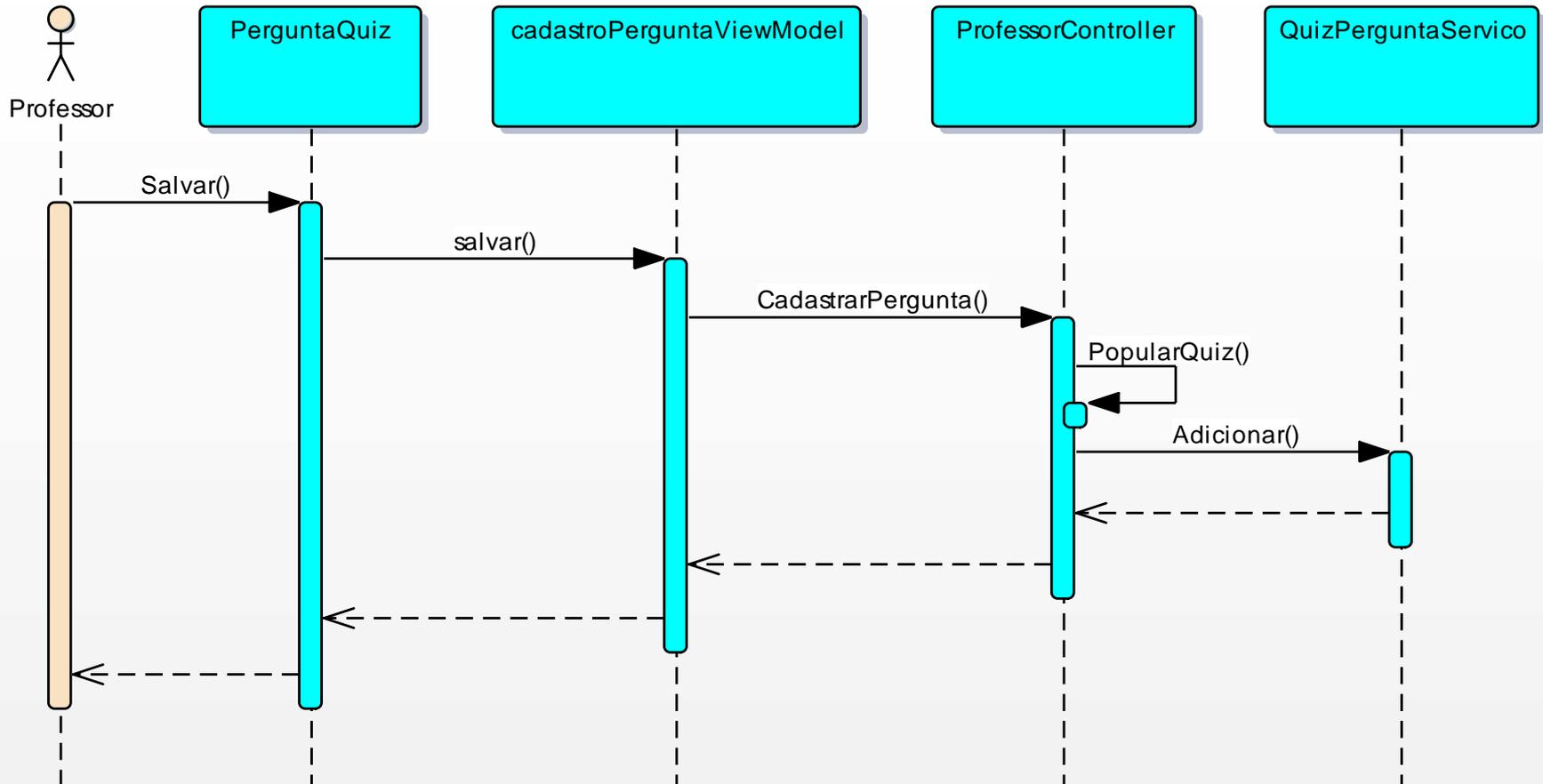
- + Adicionar(Turma, List<TurmaQuizPergunta>): int
- + Atualizar(Turma, List<TurmaQuizPergunta>): void
- + AtualizarEnderecosEmailAlunos(string, int): void
- + Excluir(int): void
- + Obter(int): Turma
- + ObterPorProfessor(ISession, int, string, string, string): List<Turma>
- + ObterQuiz(int): List<QuizResposta>
- + VerificaSeEstaEmUso(ISession, int): bool

### UsuarioServico

- + Adicionar(Usuario, Dictionary<int, string>): void
- + Atualizar(Usuario): void
- + AtualizarSolicitacao(SolicitacaoAcesso): void
- + Existe(string): bool
- + ExisteSolicitacao(string): bool
- + Obter(string): Usuario
- + Obter(int): Usuario
- + ObterAdmin(): Usuario
- + ObterAlunoPorTurma(Turma): List<Usuario>
- + ObterEmail(string): string
- + ObterEmailsParaAuxilio(TipoConteudo): IList<string>
- + ObterSolicitacao(string): SolicitacaoAcesso
- + Solicitar(SolicitacaoAcesso): void

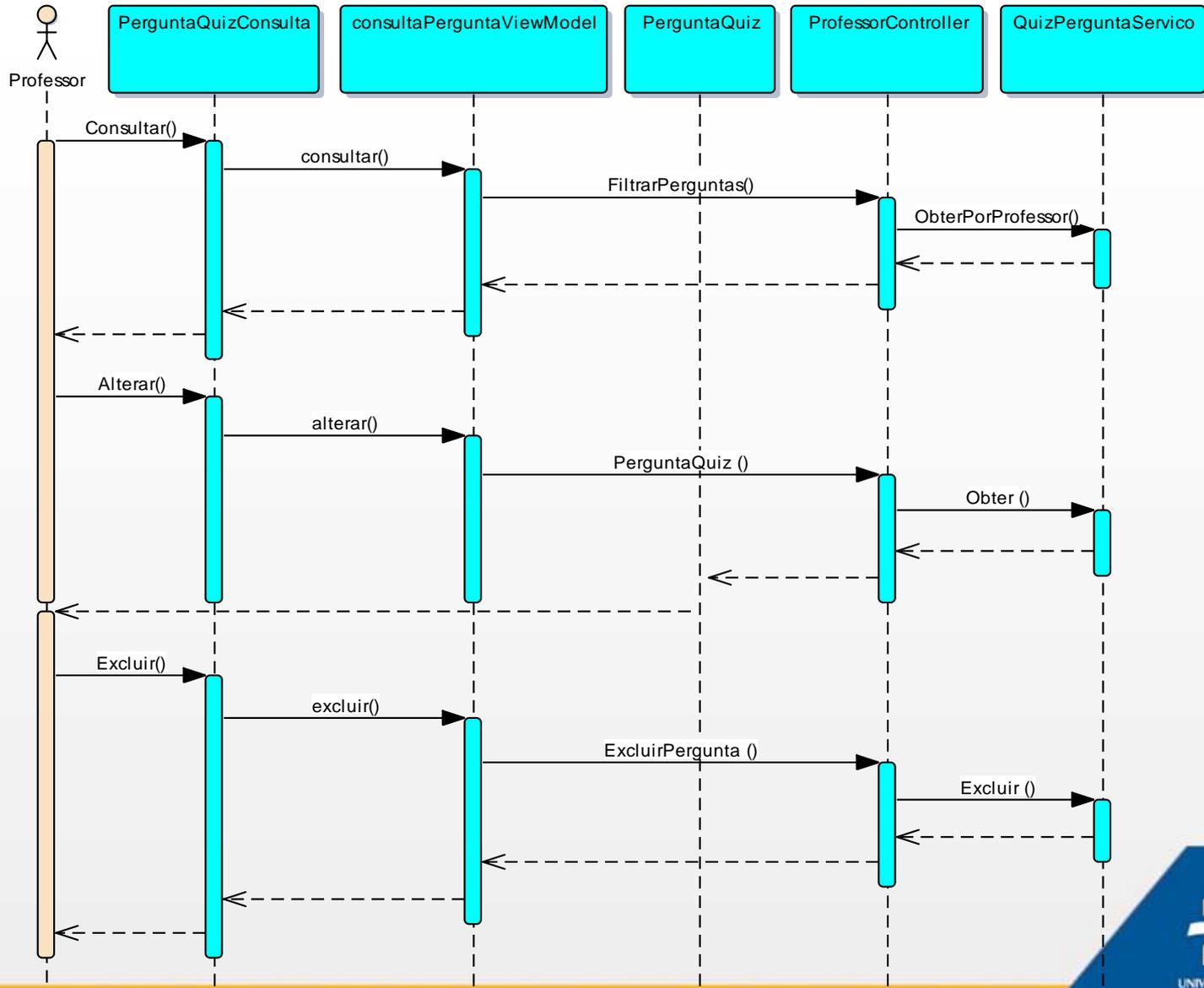
# Especificação

sd FURBOT-WEB Sequence Model Cadastro



# Especificação

sd FURBOT-WEB Sequence Model Consulta



# Implementação

- Arquitetura cliente-servidor
- MVC 5, C#, NHibernate 4.0.4.4000 e SQL Server 2014.
- JavaScript, JQuery 1.10.2, CSS 3, HTML 5, Razor MVC, Blockly, Js-Interpreter, Knockout e TecEdu
- Visual Studio 2015 e Team Foundation Server
- Navegador Google Chrome

# Implementação

- Cadastro e resolução de exercícios - TecEdu

```
124 //Inicializada a cena com posição fixa
125 var scene = new Scene().initialize(-400, -400, 400, 400);
126 layer = new Layer().initialize();
127 layer.setGravity(0);
128 //Se o robô possui posição aleatória, define primeiro os objetos com posição fixa
129 if (model.PosicaoRoboAleatoria()) {
130     objetosPosicaoFixa(model);
131     //Define a posição aleatória do robô, validando posições que já estejam sendo usadas
132     var randomX = Math.floor(Math.random() * tamanhoMapaX);
133     var randomY = Math.floor(Math.random() * tamanhoMapaY);
134     var existeObjeto = layer.listGameObjects.some(x => x.getCenterX() == (randomX * areaMapa) &&
135                                                       x.getCenterY() == (randomY * areaMapa));
136     while (existeObjeto) {
137         randomX = Math.floor(Math.random() * tamanhoMapaX);
138         randomY = Math.floor(Math.random() * tamanhoMapaY);
139         existeObjeto = layer.listGameObjects.some(x => x.getCenterX() == (randomX * areaMapa) &&
140                                                       x.getCenterY() == (randomY * areaMapa));
141     }
142     var posicaoRoboX = randomX;
143     var posicaoRoboY = randomY;
144     adicionarRobo(posicaoRoboX, posicaoRoboY, model);
145     //Define as posições dos objetos que tenham posição aleatória
146     objetosPosicaoAleatoria(model);
147 }
```

# Implementação

- Cadastro e resolução de exercícios - Blockly

```
5 Blockly.Blocks['furbot_andar_direita'] = {
6   init: function () {
7     this.appendDummyInput()
8       .appendField("andarDireita");
9     this.setPreviousStatement(true);
10    this.setNextStatement(true);
11    this.setColour(330);
12    this.setTooltip('faz o robô andar para a direita');
13  }
14 };
15 Blockly.JavaScript['furbot_andar_direita'] = function (block) {
16   var code = 'andarDireita(); \n';
17   return code;
18 };
```

# Implementação

- Sistema de recomendação de exercícios - resolução

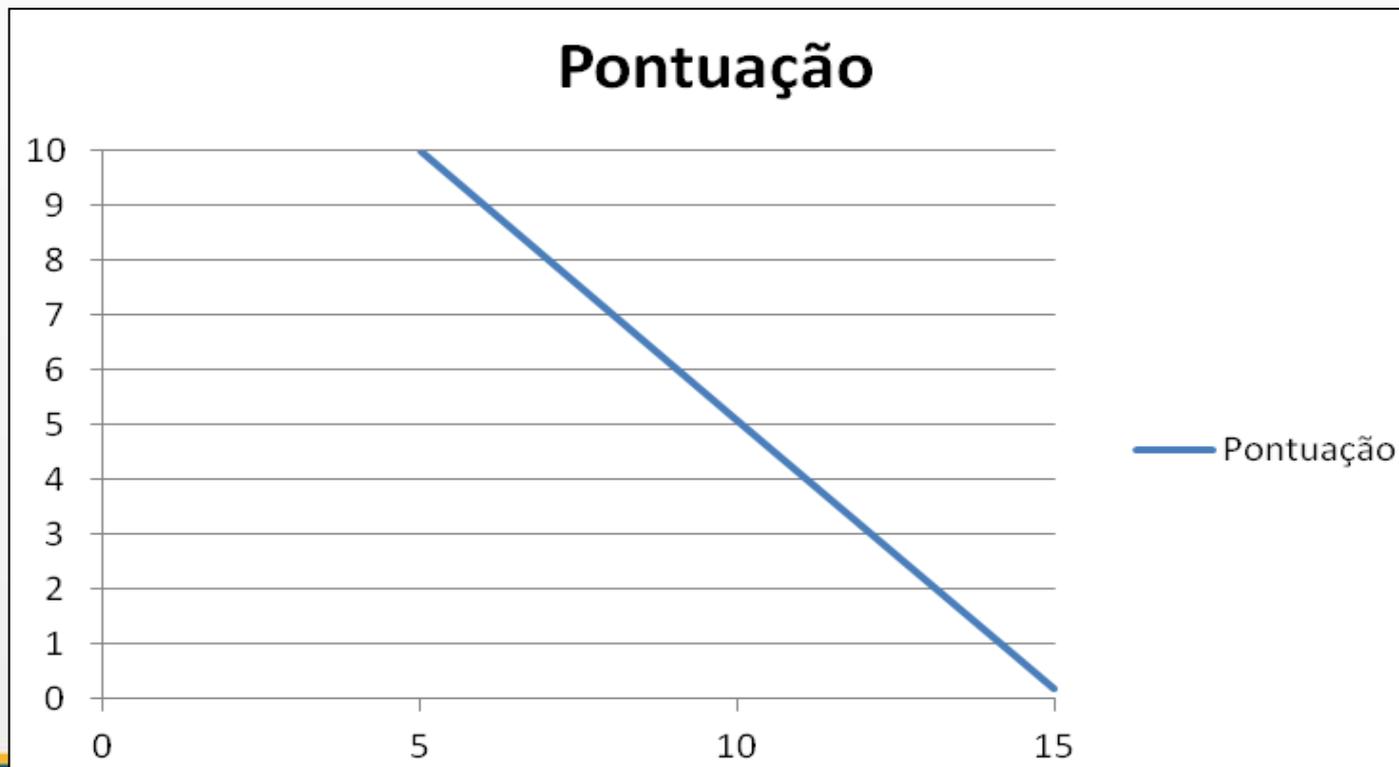
```
se quantidade de exercícios resolvidos do conteúdo atual >=3 então
    se atingiu o objetivo da metade dos exercícios resolvidos então
        se a média da correção dos exercícios resolvidos >= a 8 então
            reclassifica aluno
        fim se
        se não
            x = média da quantidade de execuções dos exercícios
            y = média do tempo de todas as resoluções dos exercícios
            se a média de x e y for maior ou igual a 8 então
                reclassifica aluno
            fim se
            se não
                não reclassifica aluno
            fim se
        fim se
    fim se
    se não
        não reclassifica aluno
    fim se
fim se
se não
    não reclassifica aluno
fim se
```

# Implementação

- Sistema de recomendação de exercícios - resolução

$$y = -0,98x + 14,9$$

Onde x é a média da quantidade de execuções dos exercícios

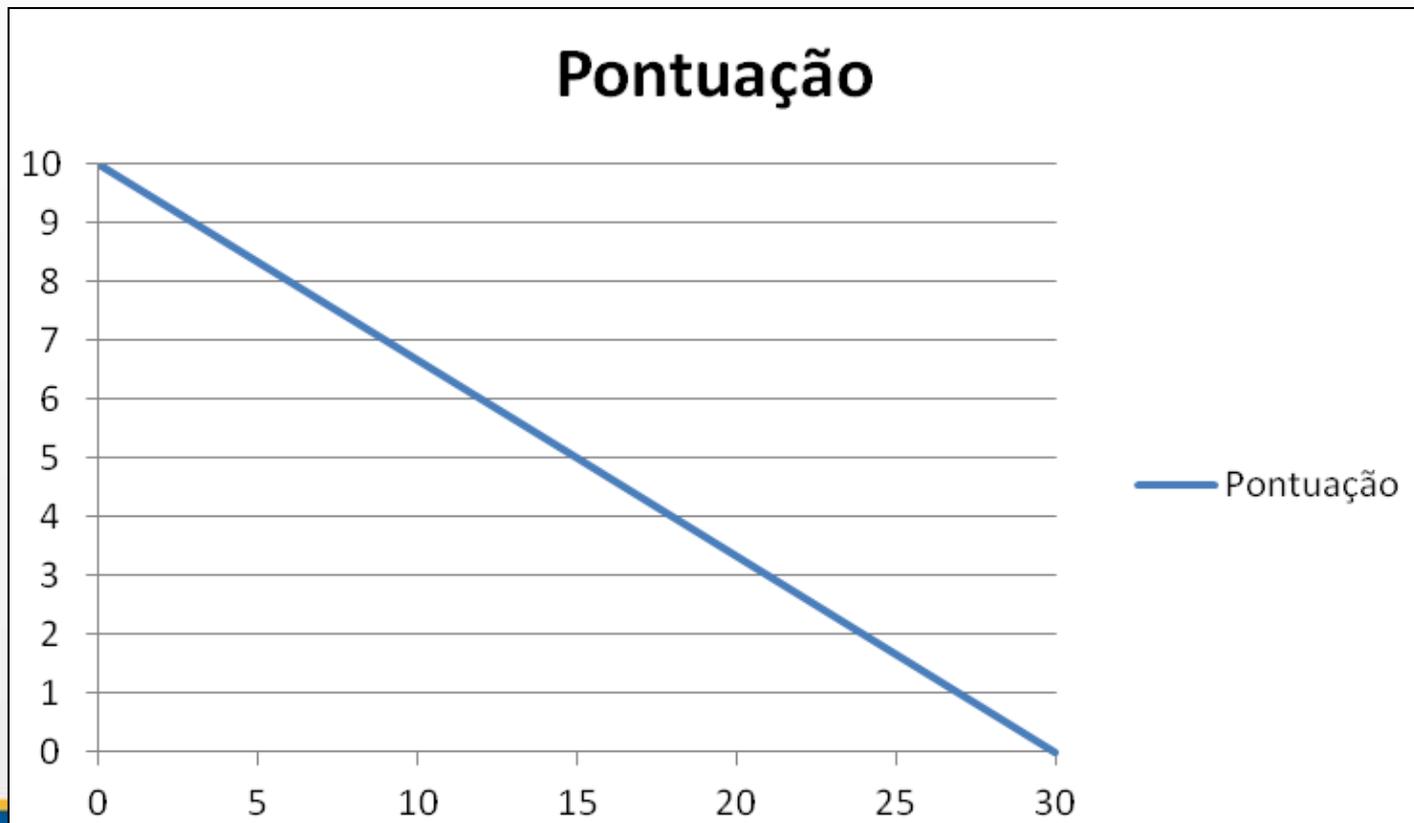


# Implementação

- Sistema de recomendação de exercícios - resolução

$$y = (-10/\text{dobro do tempo esperado})x + 10$$

Onde x é a diferença do tempo da resolução em relação ao tempo esperado



# Implementação

- Sistema de recomendação de exercícios - resolução

```
62 //Se resolveu ao menos 3 exercícios do conteúdo atual
63 if (qtdExerciciosResolvidos >= 3)
64 {
65     var qtdExerciciosResolvidosQueAtingiramObjetivo = exerciciosResolvidos.Where(x => x.AtingiuObjetivo)
66                                     .Count();
67     //Se atingiu o objetivo de pelo menos a metade dos exercícios resolvidos do conteúdo atual
68     if (qtdExerciciosResolvidosQueAtingiramObjetivo > 0 &&
69         (qtdExerciciosResolvidos / qtdExerciciosResolvidosQueAtingiramObjetivo) <= 2)
70     {
71         var exerciciosComNota = exerciciosResolvidos.Where(x => x.Nota.HasValue);
72         var qtdExerciciosComNota = exerciciosComNota.Count();
73         if (qtdExerciciosComNota > 0)
74         {
75             var mediaNotas = exerciciosComNota.Sum(x => x.Nota) / qtdExerciciosComNota;
76             //Se a média da correção dos exercícios resolvidos do conteúdo que foram corrigidos
77             //for no mínimo 8
78             if (mediaNotas >= 8)
79                 return true;
80         }
81         int mediaPontuacao = CalcularPontuacao(exerciciosResolvidos, qtdExerciciosResolvidos);
82         if (mediaPontuacao >= 8)
83             return true;
84     }
85 }
```

# Operacionalidade da Implementação

FURBOT WEB - Ciência da Computação - FURB Maurício Capobianco Lopes | Sair

[Exercícios](#) [Conteúdos](#) [Perguntas para quiz](#) [Turmas](#) [Cadastro](#)

## CADASTRO DE EXERCÍCIO

Título  Dificuldade

Tempo esperado de resolução (minutos)

Enunciado

Tamanho do mundo aleatório <input type="checkbox"/> 5	Tamanho do mundo X <input type="text" value="0"/> 6	Tamanho do mundo Y <input type="text" value="0"/> 7	Limite inferior do mundo aleatório <input type="text" value="1"/> 8
Limite superior do mundo aleatório <input type="text" value="9"/> 9	Explodir mundo <input type="checkbox"/> 10	Posição do robô aleatória no mundo <input type="text" value="11"/> 11	Posição X do robô no mundo <input type="text" value="0"/> 12
Posição Y do robô no mundo <input type="text" value="0"/> 13	Explodir robô <input type="checkbox"/> 14		

# Operacionalidade da Implementação

OBJETOS

Tipo	Posição aleatória no mundo	Posição X no mundo	Posição Y no mundo
Selecione um tipo... 15 ▾	16	0 17	0 18
Quantidade	Bloqueado	Valor aleatório	Valor
1 19	20	21	0 22
Limite inferior do valor aleatório	Limite superior do valor aleatório		
0 23	20 24		

Adicionar

Tipo	Posição X	Posição Y	Posição aleatória	Quantidade	Bloqueado	Valor	ValorAleatorio
------	-----------	-----------	-------------------	------------	-----------	-------	----------------

Página:

25

# Operacionalidade da Implementação

FURBOT WEB - Ciência da Computação - FURB

[Retornar para o login](#)

CADASTRO

Cadastrar

Nome

CPF

E-mail

Data de nascimento

Autorizo meu e-mail ser compartilhado com os alunos que precisarem de auxílio

Login

Senha

QUIZ

Qual o propósito do algoritmo a seguir?

inicio

ler X

ler Y

K  $\leftarrow$  X

X  $\leftarrow$  Y

Y  $\leftarrow$  K

escrever X

escrever Y

fim

escrever x e y  multiplicar x e y  nenhum pois deveria ter escrito a variável k  trocar os valores x e y de variável  colocar os valores de x e y em k

# Operacionalidade da Implementação

## RESOLVER EXERCÍCIO

- Solicitar auxílio para colegas **1** Visualizar material de apoio **2** Visualizar código gerado **3** Executar **4** Parar execução **5** Reiniciar mundo **6** Encaminhar para correção **7**

O robô passeia pelo modo tradicional. Em cada linha ele soma os valores dos números, sempre iniciando em 0. Sempre que ele chegar ao final da linha deve informar o valor total. No final ele precisa falar qual foi a linha que teve a maior soma. Atenção: não podem ser utilizados atributos. Somente variáveis locais.

**9**

<b>10</b>	1	1		1		1	1
1	1	1		1			1
	1	1	1			1	1
1		1	1		1	1	1
	1	1	1	1	1		1
		1	1	1	1		1
1			1	1			1
		1	1	1	1		

### Comandos FURBOT

- Movimentação
- Comunicação **11**
- Avaliação do mundo
- Obtenção de coordenadas
- Manipulação de objetos
- Outros

### Comandos Básicos

- Laço
- Texto
- Lógica
- Matemática
- Listas
- Funções
- Variáveis

```

definir maiorLinha para 0
definir maiorVITotal para 0
repeita enquanto não ehFim ABAIXO
faça
  movimentarRobo com: DIRECAO DIREITA
  andarAbaixo
  movimentarRobo com: DIRECAO ESQUERDA
  se não ehFim ABAIXO
  faça andarAbaixo
diga criar texto com " maior linha: " maiorLinha
para movimentarRobo com: DIRECAO
  definir viTotal para 0
  repeita enquanto não ehFim DIRECAO
  faça
    verificarValor
    se DIRECAO = DIREITA
    faça andarDireita
    senão andarEsquerda
  verificarValor
  
```

**12**

# Operacionalidade da Implementação

FURBOT WEB - Ciência da Computação - FURB

Maurício Capobianco Lopes | Sair

Exercícios

Conteúdos

Perguntas para quiz

Turmas

Cadastro

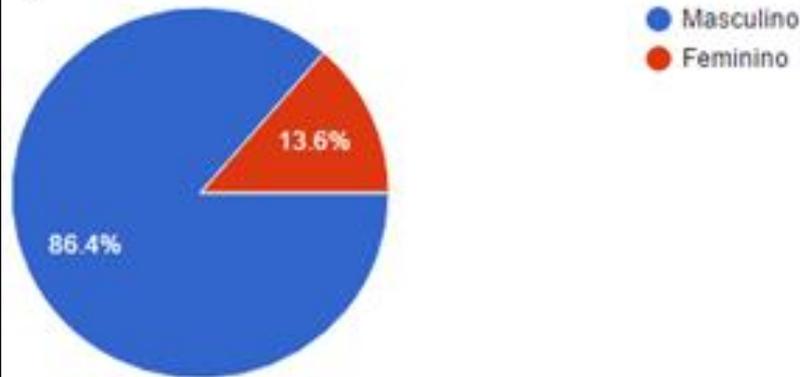
ACOMPANHAMENTO DE ALUNOS - INTRODUÇÃO A PROGRAMAÇÃO - 2016/I

	Básicos	Repetição (ENQUANTO)	Seleção (SE)	Subprogramas	Repetição (PARA)
Fulano da Silva	■	■	■	■	■
Fulano da Costa	■	■	■	■	■
Fulano dos Santos	■	■	■	■	■
Fulano Pereira	■	■	■	■	■
Fulano Junior	■	■	■	■	■
Fulano Abreu	■	■	■	■	■
Fulano Filho	■	■	■	■	■
Fulano Neto	■	■	■	■	■
Fulano Bisneto	■	■	■	■	■

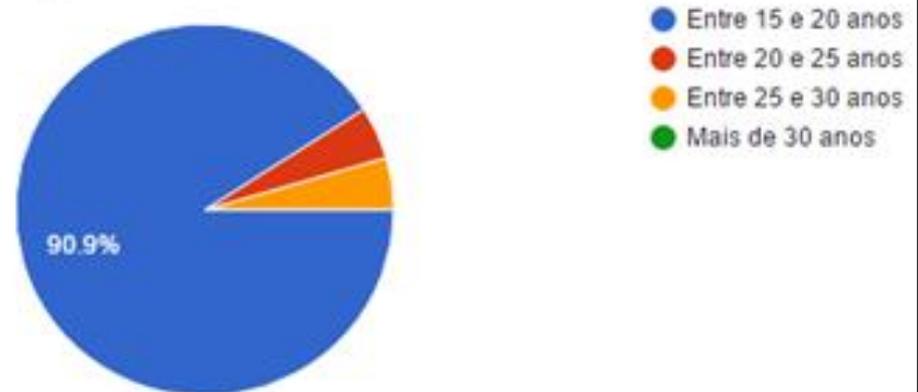
# Resultados e Discussões

- Experimento de usabilidade com 22 alunos de BCC (A)

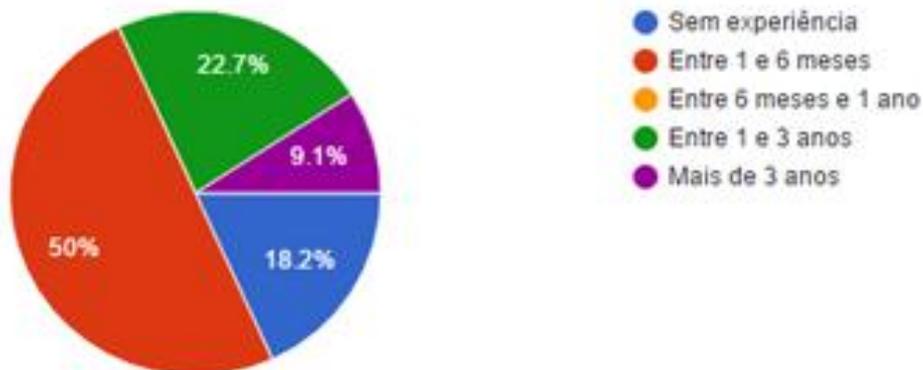
a) sexo



b) idade



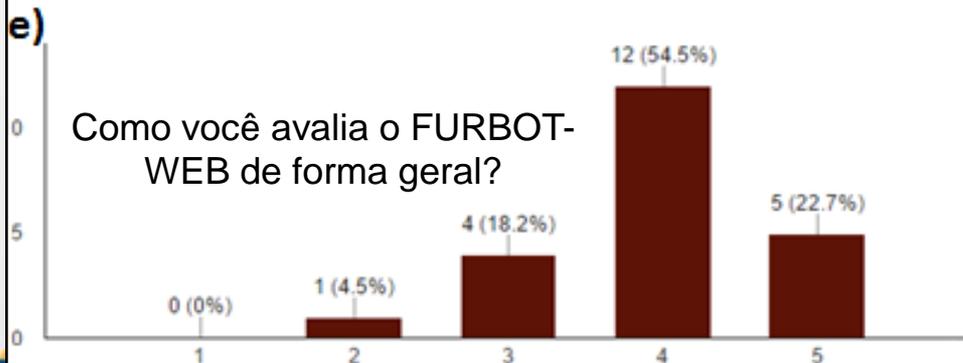
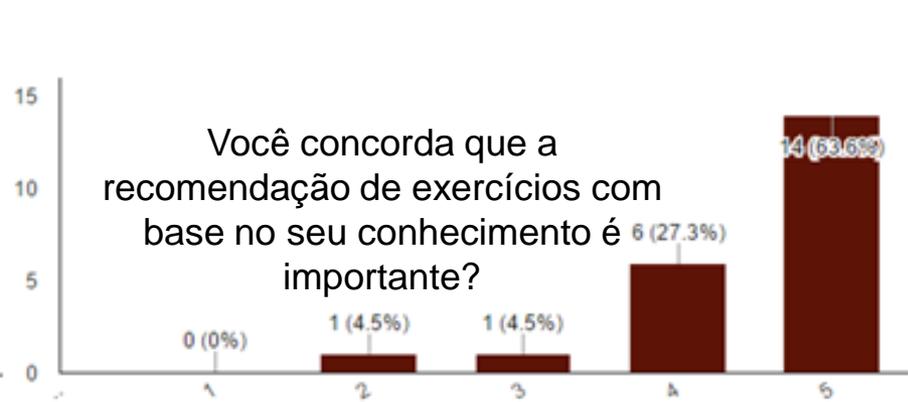
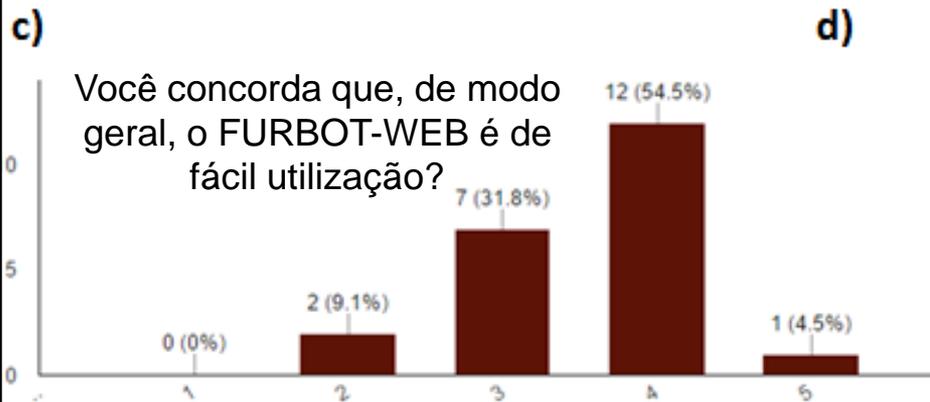
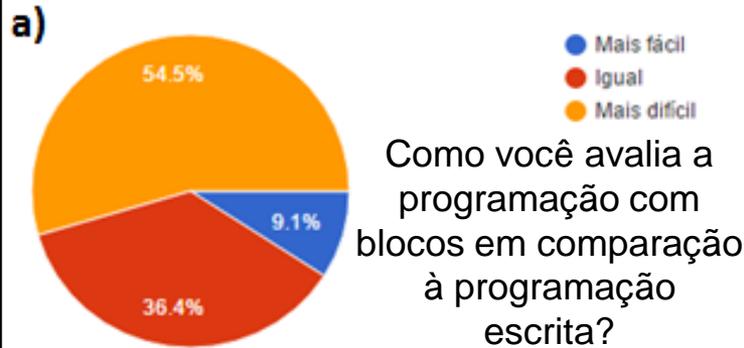
c) tempo de experiência com programação



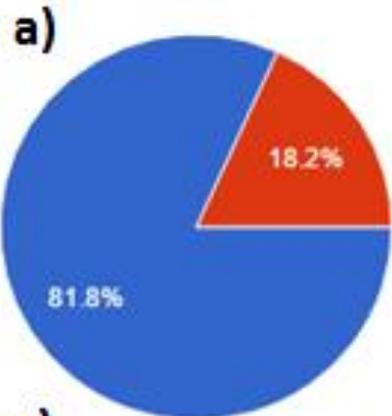
d) possui acesso à internet em casa?



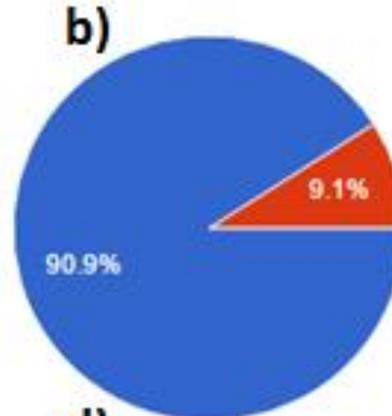
# Resultados e Discussões



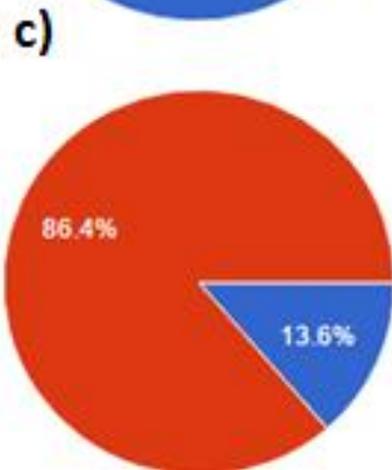
# Resultados e Discussões



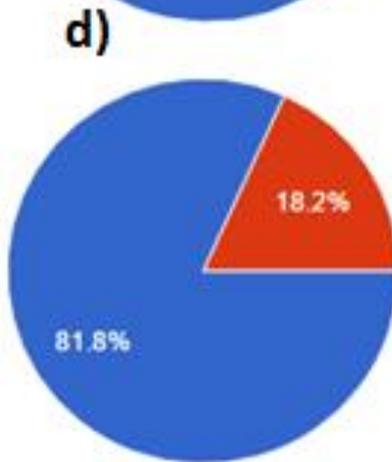
Qual das versões facilita mais a programação?



Qual das versões contribui mais para o seu conhecimento?



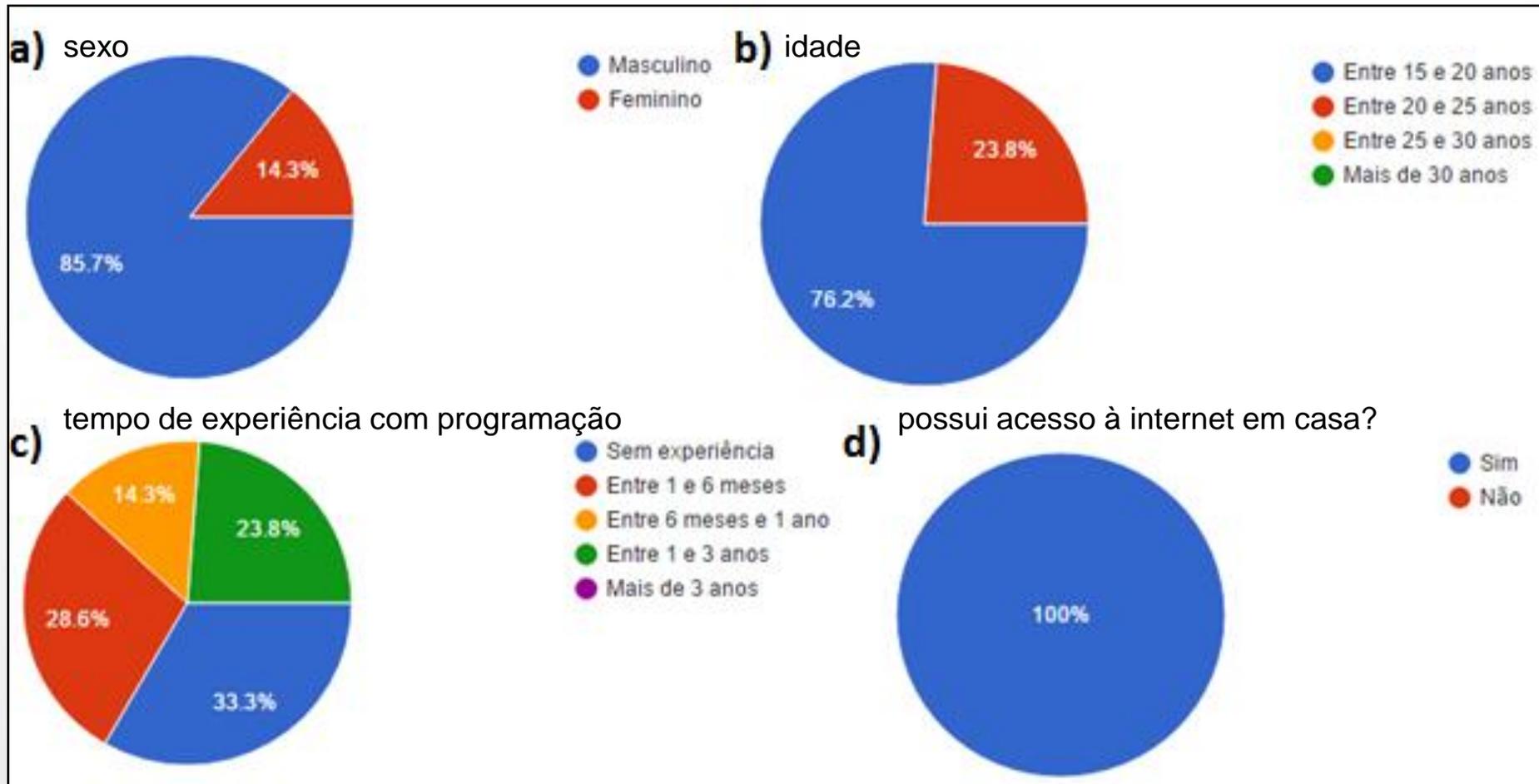
Qual das versões possui usabilidade mais amigável/atrativa?



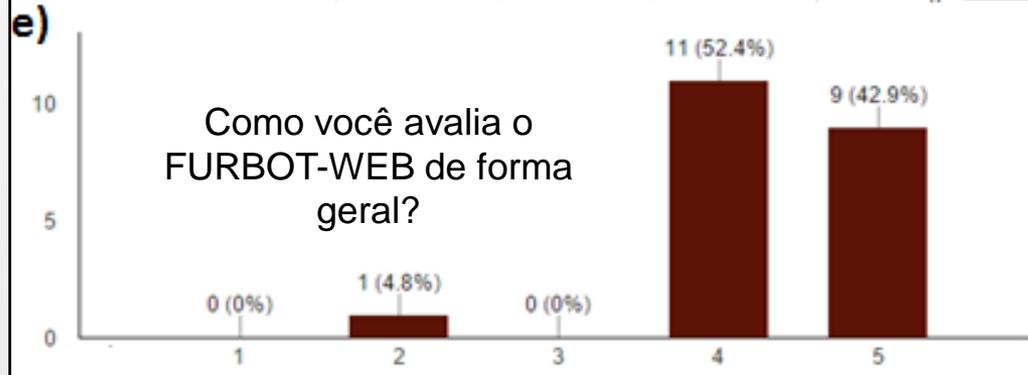
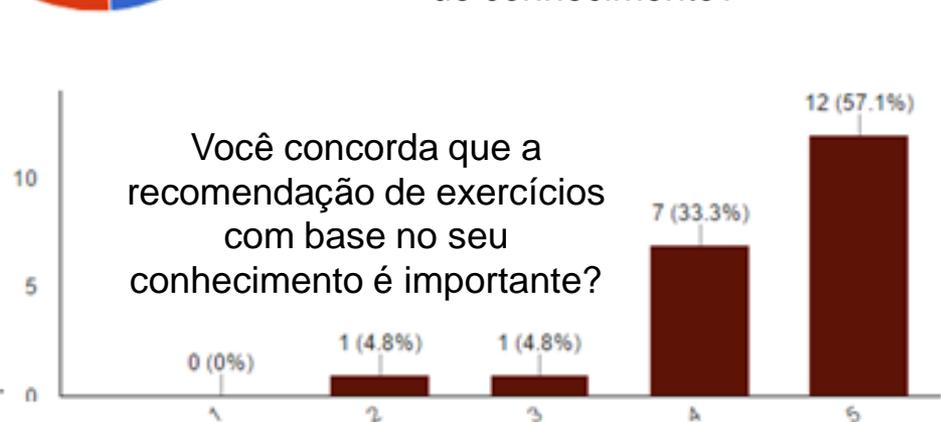
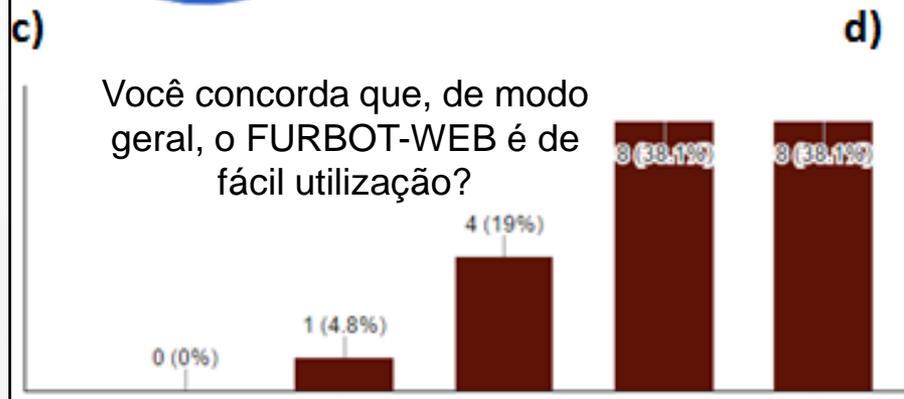
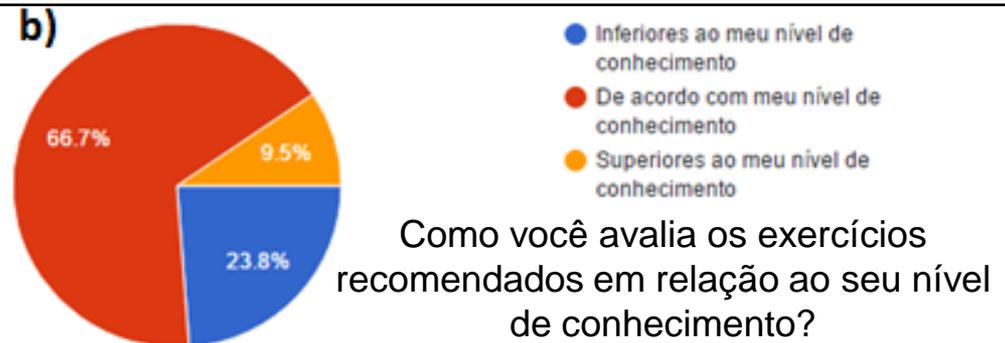
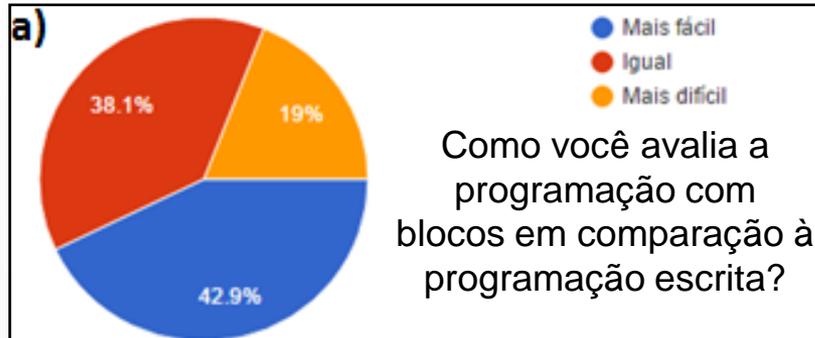
Se você pudesse optar por usar uma das versões para realizar os exercícios de uma disciplina, por qual versão optaria?

# Resultados e Discussões

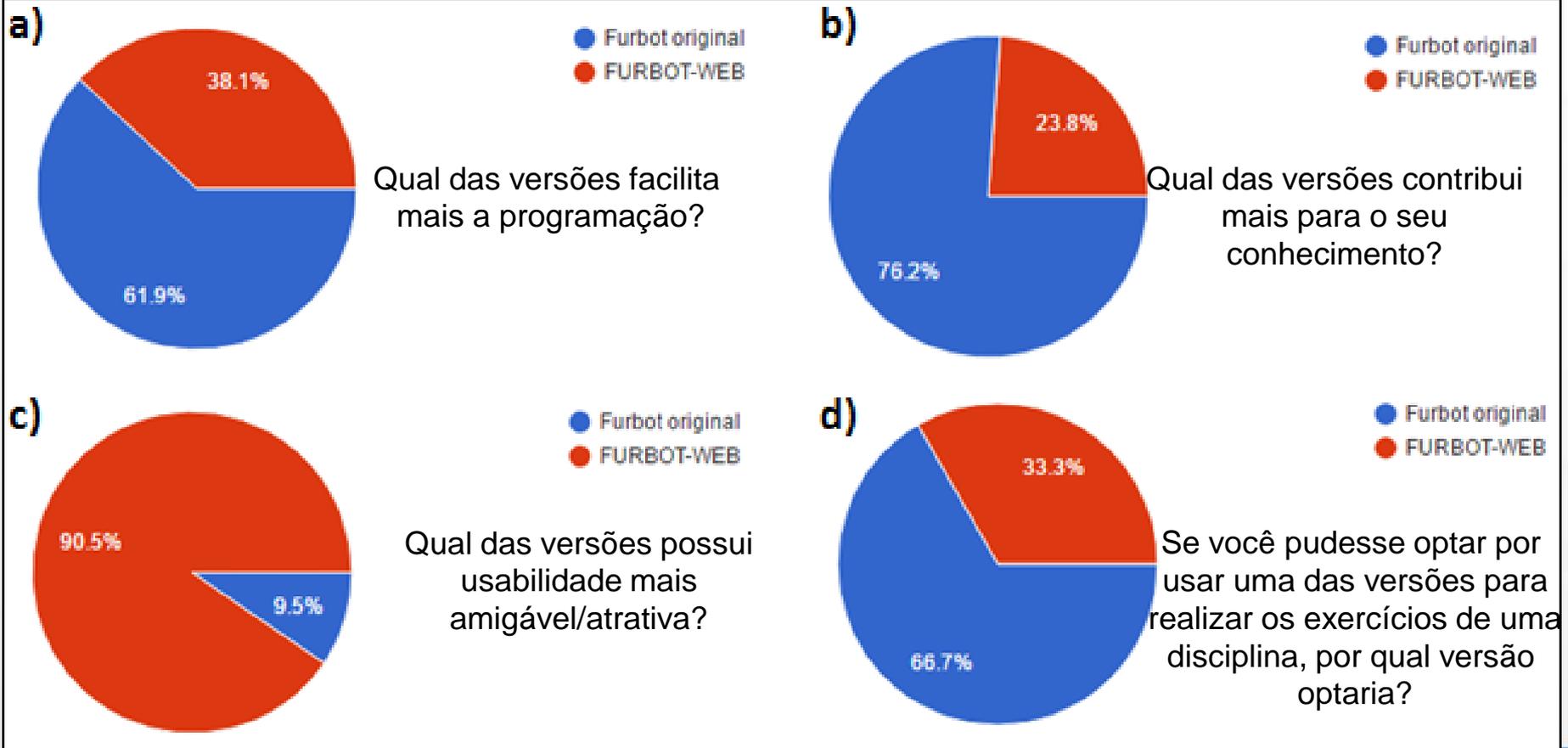
- Experimento de usabilidade com 21 alunos de BCC (B)



# Resultados e Discussões

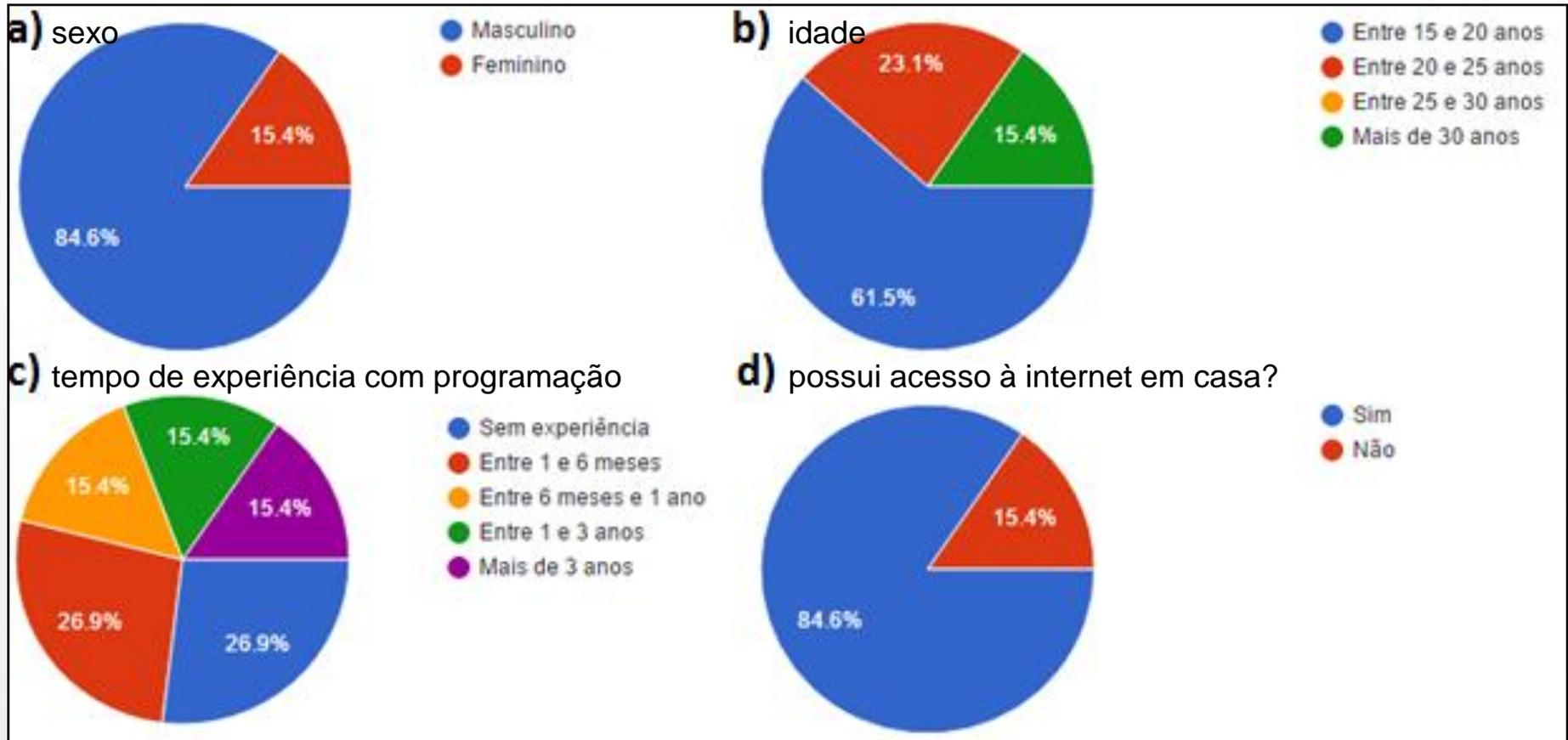


# Resultados e Discussões

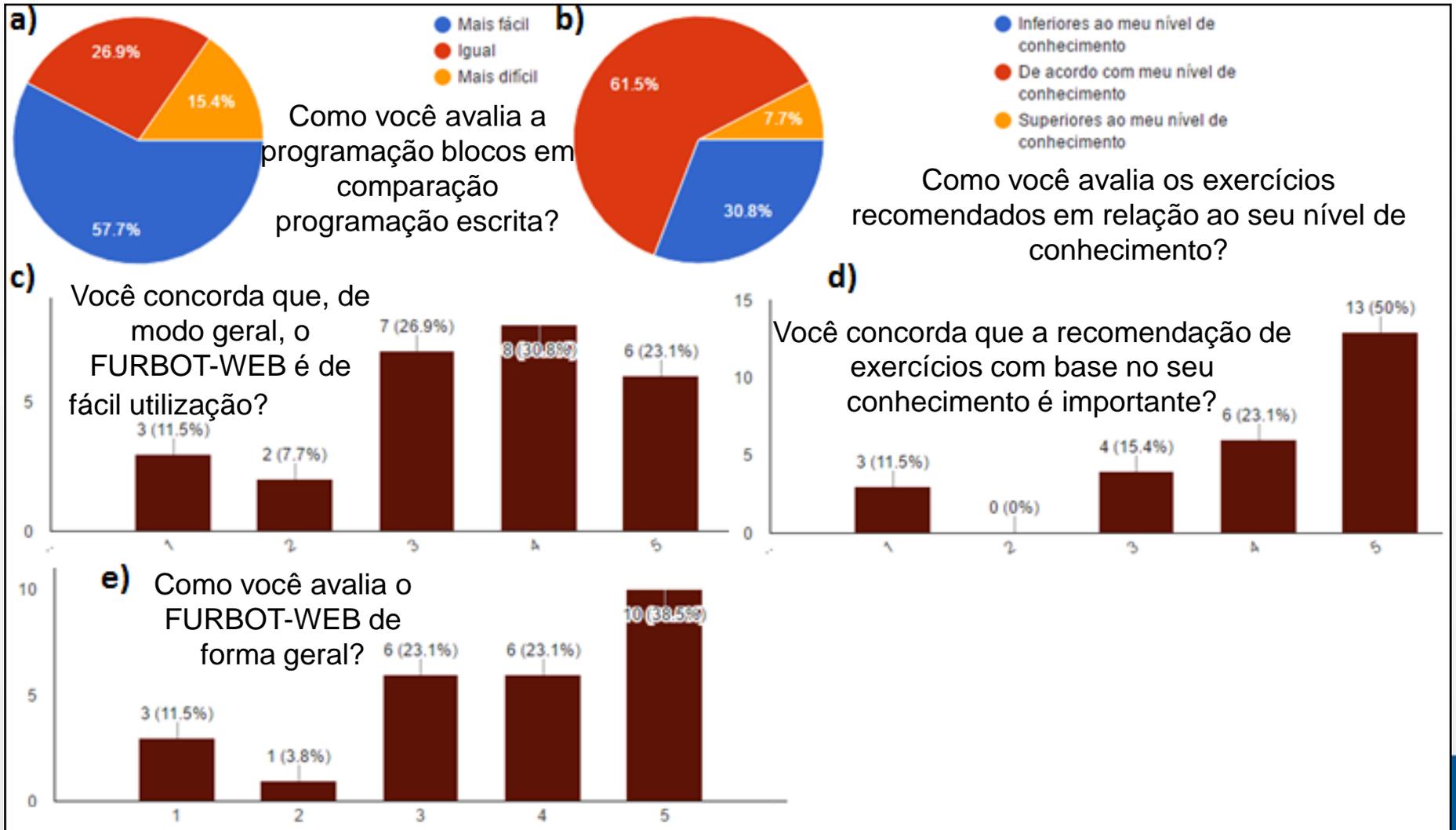


# Resultados e Discussões

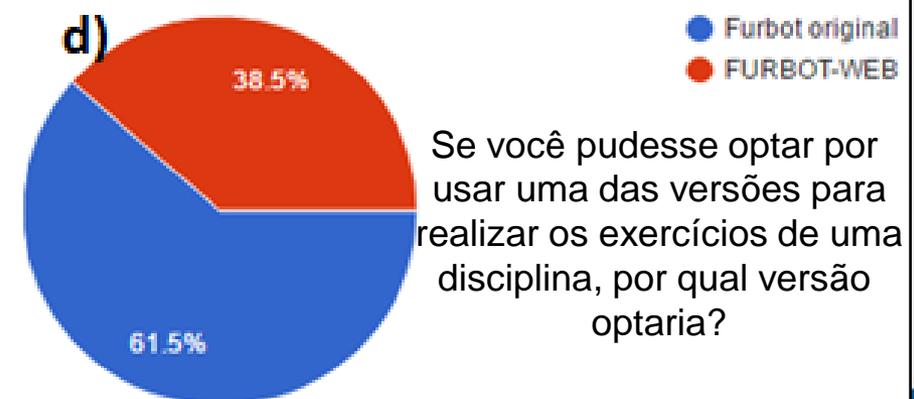
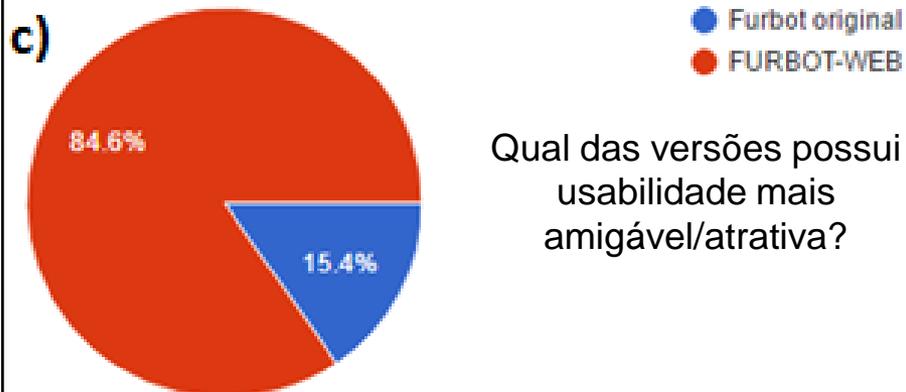
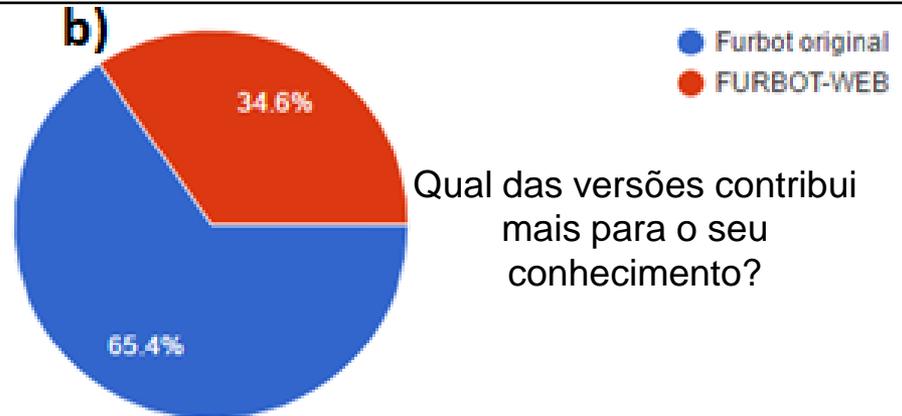
- Experimento de usabilidade com 26 alunos de SIS



# Resultados e Discussões



# Resultados e Discussões



# Resultados e Discussões

- **Comparativo dos trabalhos correlatos**

características/ trabalhos	FURBOT-WEB	Scratch	Programação com Anna e Elsa	Persona-Algo	WebSQL
programação com blocos	X	X	X		
personalização do ensino	X			X	X
aplicação web	X	X	X	X	X
possui fins educativos	X	X	X	X	X

# Conclusões e Sugestões

- Objetivos atendidos
  - FURBOT em ambiente web:
    - Usabilidade aberta e prática sem dependência de adicionar bibliotecas ou de configurar arquivos
    - Facilidade em relação ao ambiente, pois é acessível via navegador web
    - Programação com blocos mais fácil que a programação escrita

# Conclusões e Sugestões

- Objetivos atendidos
  - Plataforma de ensino personalizado
    - Sistema de recomendação com filtragem por conteúdo
    - FURBOT original não permite por não possuir controle de seção de usuários
    - Importância e assertividade para os alunos

# Conclusões e Sugestões

- Objetivos atendidos
  - Análise da usabilidade
    - Usabilidade amigável
    - Desinteressante para alunos que possuem contato com a programação escrita (não familiariedade com programação com blocos)
    - Útil no início de disciplinas que introduzam a lógica de programação

# Conclusões e Sugestões

- Ferramentas utilizadas
  - Blockly: tem muito a oferecer
  - TecEdu: utilidade dos componentes
  - Js-Interpreter: limitação na execução
- Limitação na criação do mundo

# Conclusões e Sugestões

- Relevância no âmbito acadêmico
  - Professores: automatização do processo de aplicação de exercícios e foco no aluno
  - Alunos: aprendizagem personalizada, atrativa e com potencial de desenvolvimento
- Relevância no âmbito tecnológico
  - Conjunto de recursos: C#, Sistema de Recomendação, Blockly e TecEdu
  - Desenvolvimento rápido e facilidade em futuras expansões

# Conclusões e Sugestões

- Sugestões de extensões
  - Implementar o FURBOT social, uma extensão do FURBOT-WEB com recursos de interação entre os alunos e professores e que influenciam a personalização
  - Implementar a correção automática dos exercícios resolvidos pelos alunos, utilizando agentes inteligentes
  - Permitir que o professor configure as variáveis do ambiente de personalização do ensino

# Conclusões e Sugestões

- Sugestões de extensões
  - Implementar uma IDE online para a linguagem Java, permitindo que também seja possível resolver os exercícios através da digitação do código fonte
  - Permitir que o professor execute o código fonte da resolução na correção de exercícios
  - Permitir que os alunos salvem a resolução do exercício sem precisarem enviar para correção
  - Permitir que os XMLs do FURBOT original sejam importados para o FURBOT-WEB