

SIMULADOR JOSEPHUS

Alunos:

- Luan Bonasorte Capella RA00332061
- Manoela Macchion Martedi RA00331711
- Kauã Cordeiro Cavalheiro RA00331660

Introdução

Este relatório descreve o desenvolvimento e a funcionalidade do “Simulador Josephus”, um programa em Java criado como parte do projeto para o Laboratório de Estruturas Dinâmicas. O objetivo do simulador é proporcionar uma compreensão interativa do problema de Josephus, utilizando estruturas de dados dinâmicas.

Interface do Programa

A interface gráfica do programa foi projetada para ser simples e intuitiva. O usuário insere o número de participantes no campo de texto e clica no botão para iniciar a simulação. Os resultados são exibidos em uma área de texto, mostrando claramente quais participantes foram eliminados e quem é o sobrevivente final. Os elementos interativos são claramente definidos, facilitando a interação do usuário com o programa.

Individuos:	Passos:	Velocidade:		
1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20
21	22	23	24	25
26	27	28	29	30
31	32	33	34	35
36	37	38	39	40
41	42	43	44	45
46	47	48	49	50

Buttons: Iniciar, Pausar, Reiniciar, Aplicar, Relatorio, Sair

Espaços Interativos:

- **Indivíduos:** Um campo onde o usuário pode inserir o número de indivíduos participantes na simulação.
- **Passos:** Um campo para definir o número de passos entre as eliminações no processo de simulação.
- **Velocidade:** Um menu suspenso que permite ao usuário ajustar a velocidade da simulação.

Indivíduos:	<input type="text"/>	Passos:	<input type="text"/>	Velocidade:	0.5		
-------------	----------------------	---------	----------------------	-------------	-----	--	--

Controles do Programa: Na parte inferior da interface, encontram-se os botões de controle que oferecem as seguintes funcionalidades:

- **Iniciar:** Começa a simulação com as configurações definidas.
- **Pausar:** Interrompe temporariamente a simulação.
- **Reiniciar:** Reinicia a simulação do início.
- **Aplicar:** Confirma as configurações atuais e prepara a simulação para ser iniciada.
- **Relatório:** Gera um relatório dos resultados da simulação, expondo a ordem de eliminação e o último sobrevivente.

Iniciar	Pausar	Reiniciar	Aplicar	Relatorio	Sair
---------	--------	-----------	---------	-----------	------

Visualização da Simulação: A interface também inclui uma grade numérica que representa os indivíduos na simulação, organizados em colunas e linhas, facilitando o acompanhamento visual do processo de eliminação, sendo assim, quando um indivíduo é eliminado, sua cor fica vermelha e o último sobrevivente terá sua cor verde.

Simulador Josephus

Indivíduos:	20	Passos:	3	Velocidade:	0.75		
-------------	----	---------	---	-------------	------	--	--

1	2
3	4
5	6
7	8
9	10
11	12
13	14
15	16
17	18
19	20

Iniciar	Pausar	Reiniciar	Aplicar	Relatorio	Sair
---------	--------	-----------	---------	-----------	------

Desenvolvimento Do Programa

A implementação apresentada utiliza a biblioteca Swing para criar uma interface gráfica, onde a simulação pode ser visualizada. Além disso, há funcionalidades para pausar e retomar a simulação.

Pacotes a serem destacados:

pacote app

Aplicacao: contém método main. Responsável por inicializar o programa.

GUI: utiliza JFrame e faz a interface gráfica do programa. Também possui início da lógica principal do programa.

pacote controller

Armazenador: contém a lista duplamente ligada circular utilizada para armazenar os indivíduos. Contém o vetor dinâmico utilizado para armazenar mortos. Também possui métodos usados para manipulação das estruturas.

Controller: contém métodos simular e getVetor, usados para iniciar a simulação e retornar o vetor dinâmico usado na simulação.

pacote entities

Utils: verifica o tipo de valor (entrada como string) e se pode ser numérico inteiro, tipo flutuante, não numérico ou vazio.

Pessoa: instância de uma pessoa/indivíduo com status de vivo ou morto.

pacote listaLigada

Contém classes ListaDuplamenteLigada e VetDin, juntamente com suas interfaces, que possuem a lógica das estruturas de dados.

pacote gui

Contém classe Saida e ISaida (interface), para saídas de modo gráfico.