

## Exercícios:

0) Criar em java um projeto com uma Thread chamada ThreadVetor, que receba um valor numérico e vetor como parâmetros. Caso o valor numérico seja par, a thread deve percorrer o vetor utilizando uma estrutura for ( $\text{int } i = 0 ; i < \text{vet.length}; i++$ ) e contar o tempo para percorrer o vetor. Caso o valor numérico seja ímpar, a thread deve percorrer o vetor utilizando uma estrutura foreach e contar o tempo para percorrer o vetor. No final, ambas as possibilidades devem apresentar o tempo em segundos. A operação main deve gerar um vetor de 1000 posições com valores aleatórios de 1 a 100. Deve iniciar 2 ThreadVetor e para uma passar o número 1 e o vetor e para a outra, passar o número 2 e o mesmo vetor.

## Exercícios:

- 1) Fazer uma aplicação que rode 5 Threads que cada uma delas imprima no console o seu número.
- 2) Fazer uma aplicação que insira números aleatórios em uma matriz 3 x 5 e tenha 3 chamadas de Threads, onde cada uma calcula a soma dos valores de cada linha, imprimindo a identificação da linha e o resultado da soma.

## Exercícios:

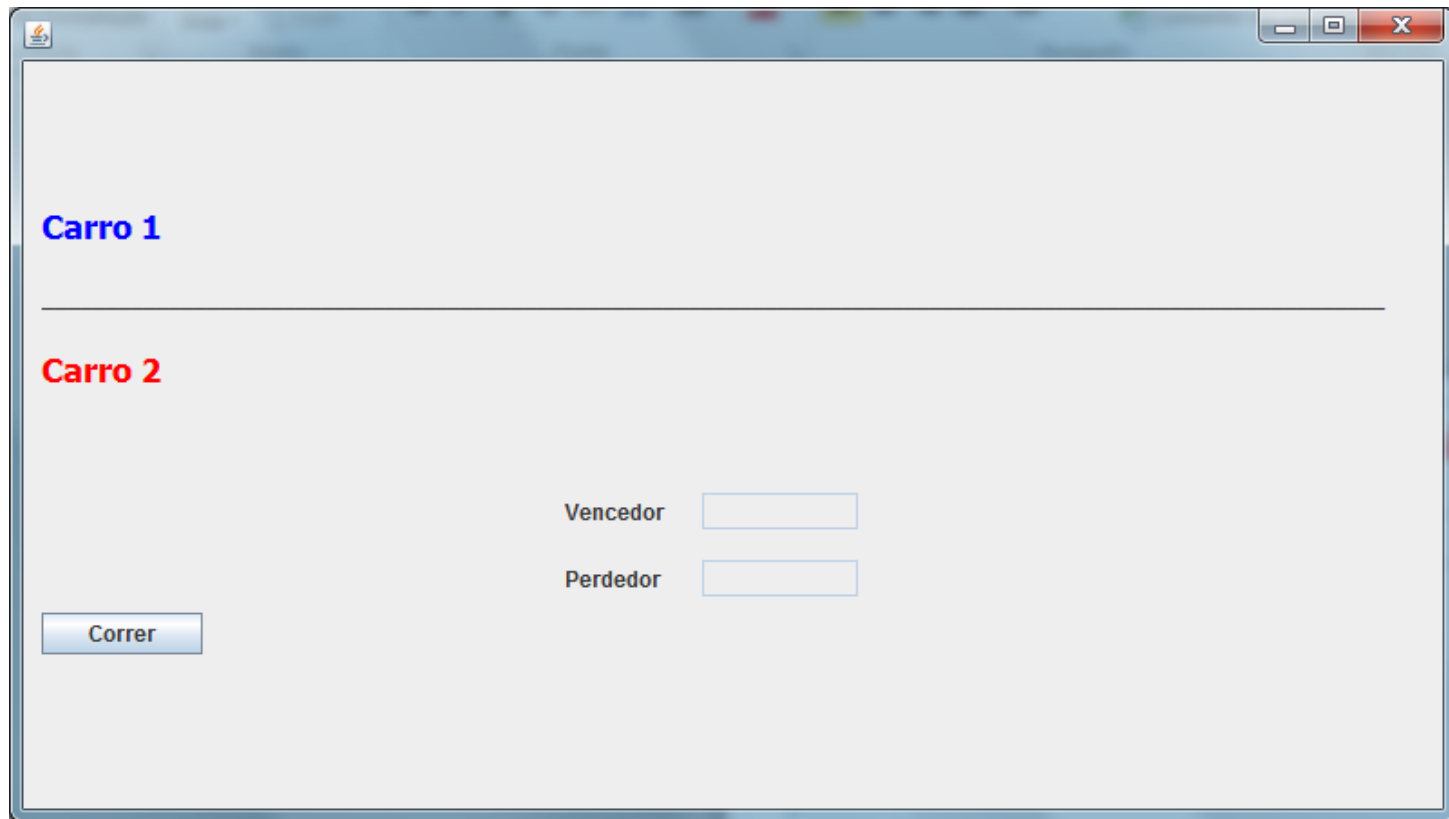
3) Fazer uma aplicação de uma corrida de sapos, com 5 Threads, cada Thread controlando 1 sapo. Deve haver um tamanho máximo para cada pulo do sapo (em metros) e a distância máxima para que os sapos percorram. A cada salto, um sapo pode dar um salto de 0 até o tamanho máximo do salto (valor aleatório). Após dar um salto, a Thread, para cada sapo, deve mostrar no console, qual foi o tamanho do salto e quanto o sapo percorreu. Assim que o sapo percorrer a distância máxima, a Thread deve apresentar que o sapo chegou e qual sua colocação.

## Exercícios:

4) Utilizando o Java SWING, criar uma tela, semelhante à tela abaixo, para criar uma corrida de carros, tipo *drag race*. A aplicação deve ter a distância que os carros devem correr e a velocidade máxima dos carros. Os carros (Jlabel) devem, a cada 100 mS, dar uma arrancada de velocidade que pode estar entre 0 e a velocidade máxima (definida aleatoriamente). Assim que o primeiro carro chegar, o JTextField Vencedor deve receber o nome deste e o JTextField Perdedor receberá o nome do outro carro. Assim que a corrida se inicia, o botão Correr deve sumir.

## Exercícios

4)



The screenshot shows a Java Swing window with a light gray background. The window has a title bar with standard OS controls (minimize, maximize, close). Inside the window, the text "Carro 1" is displayed in blue at the top left. A horizontal line separates this from the text "Carro 2" which is displayed in red below it. In the bottom left corner, there is a button labeled "Correr". In the bottom right area, there are two labels: "Vencedor" and "Perdedor", each followed by an empty text input field.

## Exercícios:

5) Fazer, com o Java SWING, uma aplicação de caça-níquel, conforme figura abaixo. O caça níquel tem 3 JTextFields, independentes, que giram, aleatoriamente, de 1 a 150 vezes e apresentará um número de 1 a 7. Quando iniciado, o botão Jogar deve desaparecer.

