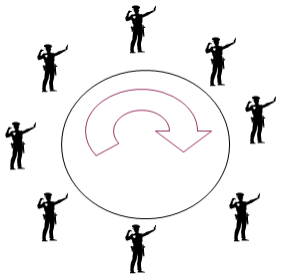
UNIVERSIDADE PRESBITERIANA MACKENZIE
Faculdade de Computação e Informática
Prof. Dr. Ivan Carlos Alcântara de Oliveira
Estruturas de Dados I

Atividade N2 (1) - em dupla ou três
Problema do Josephus

Nome	**TIA**
Erik Samuel Viana Hsu	32265921
Mateus Kenzo Iochimoto	32216289
Rodrigo Machado de Assis Oliveira de Lima	32234678
Thiago Shihan Cardoso Toma	32210744



Implementar o Problema de Josephus, com a lista encadeada circular e as operações necessárias. O seu programa deve permitir apresentar ao final o nome do ganhador!

Para isso, utilizar a lista simplesmente encadeada circular contendo as operações necessárias para o seu funcionamento fornecida em aula.

O nó da lista deve conter um dado do tipo inteiro e uma string. A lista pode conter, além do cabeça de lista, a quantidade de elementos.

Na sua implementação, gerar um menu de opções que permita simular o funcionamento do problema de Josephus, similar ao abaixo:

Problema do Josephus

1) Iniciar: cria uma lista vazia;
2) Inserir soldado: insere um soldado na lista (final ou começo, tanto faz, mas sempre da mesma forma). O soldado tem um código e um nome.
3) Sortear número (entre 1 e 100): sorteia um número aleatório e apresenta o seu resultado.
4) Retirar soldado: começar do cabeça (caso seja a primeira vez que sorteou um número, lembrar que deve ser informado o nome do soldado o a começar a percorrer) ou da posição do último soldado removido, percorrer a lista pelo número sorteado e nó contendo o nome do soldado (cada nó percorrido soma 1 e segue até o valor sorteado). Então, remover o soldado. Caso a lista restante tenha somente 1 elemento, apresente o nome do felizardo que vai pegar o cavalo.
5) Mostrar os soldados: apresenta a lista de soldados com a informação do seu código e nome, partindo do nó cabeça.

6) Fim.

Obs.: Será necessário alterar algumas operações das classes *CircleLinkedList* e *Node*.

No início do relatório incluir pelo menos 2 testes das operações realizadas.

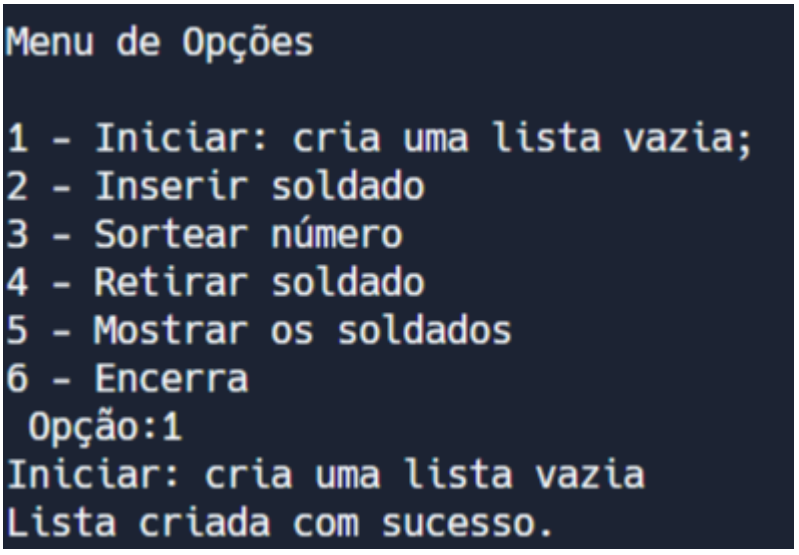
Colocar o código fonte completo em um Apêndice (no final do Relatório).

Não esquecer de enviar em separado o código fonte completo desenvolvido em cada exercício.

Obs.: Para geração de números randômicos, se desejar, veja o descrito em: https://dicasdejava.com.br/como-gerar-um-numero-aleatorio-em-java/.

RELATÓRIO DE LABORATÓRIO

Printscreens de testes da opção 1 do menu:



Printscreens de testes da opção 2 do menu:
(quando é selecionado a opção 2 antes de criar a lista)

```
1 - Iniciar: cria uma lista vazia;
2 - Inserir soldado
3 - Sortear número
4 - Retirar soldado
5 - Mostrar os soldados
6 - Encerra
 Opção:2
Crie uma lista antes!
```

(situação normal de utilização da opção 2)

```
1 - Iniciar: cria uma lista vazia;
2 - Inserir soldado
3 - Sortear número
4 - Retirar soldado
5 - Mostrar os soldados
6 - Encerra
 Opção:2
Inserir soldado

Digite o código do soldado (digitar -1 vai parar o loop):
4
Digite o nome do soldado:
josé
Digite o código do soldado (digitar -1 vai parar o loop):
2
Digite o nome do soldado:
josephus
Digite o código do soldado (digitar -1 vai parar o loop):
6
Digite o nome do soldado:
roberto
Digite o código do soldado (digitar -1 vai parar o loop):
-1
```

Printscreens de testes da opção 3 do menu:
Teste 1:

```
Menu de Opções

1 - Iniciar: cria uma lista vazia;
2 - Inserir soldado
3 - Sortear número
4 - Retirar soldado
5 - Mostrar os soldados
6 - Encerra
 Opção:3
Sortear número
Número sorteado:77
```

Teste 2:

```
1 - Iniciar: cria uma lista vazia;
2 - Inserir soldado
3 - Sortear número
4 - Retirar soldado
5 - Mostrar os soldados
6 - Encerra
 Opção:3
Sortear número
Número sorteado:57
```

Printscreens de testes da opção 4 do menu:

Teste 1:

```
Menu de Opções

1 - Iniciar: cria uma lista vazia;
2 - Inserir soldado
3 - Sortear número
4 - Retirar soldado
5 - Mostrar os soldados
6 - Encerra
 Opção:4
Retirar soldado
Soldado eliminado: Pedro
```

Teste 2:

```
Menu de Opções

1 - Iniciar: cria uma lista vazia;
2 - Inserir soldado
3 - Sortear número
4 - Retirar soldado
5 - Mostrar os soldados
6 - Encerra
 Opção:4
Retirar soldado
Soldado eliminado: Felipe
```

Printscreens de testes da opção 5 do menu:
Teste 1:

```
Menu de Opções

1 - Iniciar: cria uma lista vazia;
2 - Inserir soldado
3 - Sortear número
4 - Retirar soldado
5 - Mostrar os soldados
6 - Encerra
 Opção:5
Mostrar os soldados
Lista de soldados: [ 12 Felipe, 510 João, 32 André ]
```

Teste 2:

```
Menu de Opções

1 - Iniciar: cria uma lista vazia;
2 - Inserir soldado
3 - Sortear número
4 - Retirar soldado
5 - Mostrar os soldados
6 - Encerra
 Opção:5
Mostrar os soldados
Lista de soldados: [ 510 João, 32 André ]
```

Printscreens de testes da opção 6 do menu:

```
Menu de Opções

1 - Iniciar: cria uma lista vazia;
2 - Inserir soldado
3 - Sortear número
4 - Retirar soldado
5 - Mostrar os soldados
6 - Encerra
 Opção:6
Programa encerrado
```

Código fonte desenvolvido:

- **Código fonte Main.java:**

```java
import java.util.Scanner;
import java.util.Random;

class Main {
  public static void main(String[] args) {
    String opcoes = "\nMenu de Opções\n\n1 - Iniciar: cria uma lista vazia;\n2 - Inserir soldado\n3 - Sortear número\n4 - Retirar soldado\n5 - Mostrar os soldados\n6 - Encerra\n Opção:";
    Scanner ent = new Scanner(System.in);
    int num_sorteado = -1;// Se num_sorteado == -1 --> erro
    CircleLinkedlist lista = null;
    int opcao = 0;
    Node head = null;
    boolean flag = false;
    boolean flag2 = false;
    do{
      System.out.print(opcoes);
      System.out.print(opcoes);
      opcao = ent.nextInt();
      switch(opcao){
        case 1:
          System.out.println("Iniciar: cria uma lista vazia");
          lista = new CircleLinkedlist();
          if (lista != null) System.out.print("Lista criada com sucesso.");
          else System.out.print("Problema na criação da lista. ");
          break;
        case 2:
          if (lista == null)
          {
            System.out.print("Crie uma lista antes!\n");
            break;
          }
          else
          {
          System.out.println("Inserir soldado\n");
          Scanner inputStr = new Scanner(System.in);
          Scanner inputInt = new Scanner(System.in);
          int codigo;
          String nome;
          while (true)
          {
          System.out.println("Digite o código do soldado (digitar -1 vai parar o loop): ");
          codigo = inputInt.nextInt();
          if (codigo == -1) break;
          System.out.println("Digite o nome do soldado: ");
          nome = inputStr.nextLine();

          lista.insertTail(codigo, nome);
          }}
          break;

```

```
49          case 3:
50            System.out.println("Sortear número");
51            Random r = new Random();
52            int low = 1;
53            int high = 101;
54            num_sorteado = r.nextInt(high-low) + low;
55            System.out.println("Número sorteado:"+num_sorteado);
56            break;
57
58
59          case 4:
60            System.out.println("Retirar soldado");
61
62            if (lista.getCount() == 1) {
63              System.out.print("O soldado ganhador é: " + head.getNome());
64              break;
65            }
66
67            if (flag2 == false){
68              head = lista.getHead();
69              flag2 = true;
70            }
71
72            int i;
73            for (i = 0; i < num_sorteado; i++){
74              head = head.getProx();
75            }
76
77              int id = head.getId();
78              lista.remove(id);
79              String nm = head.getNome();
80              System.out.println("Soldado eliminado: " + nm);
81
82              head = head.getProx();
83
84              break;
85
86            case 5:
87              System.out.println("Mostrar os soldados");
88              lista.print();
89              break;
90
91            case 6:
92              System.out.println("Programa encerrado");
93              break;
94          }
95        }while (opcao != 6);
96      }
97    }
98
```

- **Código fonte CircleLinkedlist.java:**

```java
public class CircleLinkedlist {
  private Node head;
  private Node tail;
  private int count;

  public CircleLinkedlist() {
    head = tail = null;
    count = 0;
  }

  public boolean isEmpty() {
    return head == null;
  }

  public boolean isFull() {
    Node aux = new Node();
    return aux == null;
  }

  public int getCount() {
    return count;
  }
```

```java

public boolean insertTail(int id, String nome){
  Node aux;
    if (!isFull()){ // Não há espaço de memória
      aux = new Node(id, nome, null);
      if (isEmpty()){ // Lista está vazia insere no cabeça
      aux.setProx(head);
        head = tail = aux;
      }else { // Insere no final e atualiza os ponteiros
      tail.setProx(aux);
      aux.setProx(head);
      tail = aux;
      }
      count++;
    return true;
    }
    else return false;
};

public boolean insertHead(int id, String nome){
  Node aux;
    if (!isFull()){
      aux = new Node(id, nome, null);
      if (isEmpty()){ // Lista está vazia
      aux.setProx(head);
      head = tail = aux;
      }else { // Insere no começo e atualiza os ponteiros
        aux.setProx(head);
        head = aux;
        tail.setProx(head);
      }
      count++;
      return true;
    }
    else return false;
};

```

```java
public Node search(int id){
  Node pAnda;

   if (isEmpty()) {
   return null; // Lista vazia
   }else{
     pAnda = head;
     // procura a posição do elemento na lista
     while ((pAnda.getProx() != head) && (pAnda.getId() != id))
       pAnda = pAnda.getProx();
     if (pAnda.getId() == id)
       return pAnda; // Retorna a referência para o No
     return null; // elemento não encontrado
   }
 }

```

```java
public boolean remove(int id){
  Node pAnt = null, pAnda;
    if (isEmpty()) return false; // Lista vazia
    else{
      pAnda = head;
      int contador = 0;
      // procura a posição do elemento na lista
      while ((contador != count) && (pAnda.getId() != id)){
        pAnt = pAnda;
        pAnda = pAnda.getProx();
      contador++;
      }
      if ((contador == count) && (pAnda.getId() != id))
        return false; // Se não encontrou o elemento
      // Se elemento encontrado remove da lista
      else {
        // se o elemento encontrado está na cabeça da lista
      // e tem somente um elemento
        if ((head == pAnda && count == 1)) {
      head = null;
      tail = null;
     // está no cabeça e tem mais elementos
      } else if ((head == pAnda && count != 1)){
        head = head.getProx();
        tail.setProx(head);
      }else {// remove elemento do meio/fim
        // Se o elemento estiver no fim
        if (pAnda == tail)
          tail = pAnt; // Atualiza o fim
        pAnt.setProx(pAnda.getProx());
    }
        count--;
        return true;
      }
    }
 }

```

```java
public void print(){
  Node pAnda;
    System.out.print("Lista de soldados: [ ");
    if (!isEmpty()) {
      pAnda = head;
      while (pAnda.getProx() != head) {
        System.out.print(pAnda.getId()+" ");
        System.out.print(pAnda.getNome() + ", ");
        pAnda = pAnda.getProx();

      }
      System.out.print(pAnda.getId()+" ");
      System.out.print(pAnda.getNome()+" ");
    }
    System.out.print("]\n");
}

public Node getHead(){
  return head;
}

```

```java
public void clear(){
  Node pAnt, pAnda = head;

  while(pAnda.getProx() != head){
    pAnt = pAnda;  // Libera o nó
    pAnda = pAnda.getProx();
    pAnt.setProx(null);
    pAnt = null;
  }
  count = 0;
  tail = head = null;
}

@Override
public String toString() {

```

```
150
151     StringBuilder sb = new StringBuilder();
152     int qtde = 0;
153     sb.append("\n[Lista]\n");
154
155       sb.append("L: [ ");
156       Node pAnda = head;
157       while (qtde != count) {
158         sb.append(pAnda.getId()+" ");
159         qtde++;
160         pAnda = pAnda.getProx();
161       }
162       sb.append("]\n");
163
164       sb.append("Qtde.: " + count);
165       if (count != 0) {
166         sb.append("\nPrimeiro: " + head.getId() +
167             ", Ultimo: " + tail.getId());
168       }
169
170       sb.append("\n");
171       return sb.toString();
172   }
173
174 }
```

- **Código fonte Node.java:**

```
1  /* NOME: ERIK SAMUEL VIANA HSU    TIA: 32265921
2     NOME: MATEUS KENZO IOCHIMOTO   TIA: 32216289
3     NOME: RODRIGO MACHADO DE ASSIS OLIVEIRA DE LIMA   TIA: 32234678
4     NOME: THIAGO SHIHAN CARDOSO TOMA: TIA: 32210744
5   */
6
7  public class Node {
8      private int id;
9    private String nome;
10     private Node prox;
11
12     public Node() {
13         this(0, null ,null);
14     }
15
16     public Node(int id, String nome, Node prox) {
17         this.id = id;
18     this.nome = nome;
19         this.prox = prox;
20     }
21
22     public Node getProx() { return prox; };
23     public int getId(){ return id; };
24   public String getNome(){ return nome; }
25
26     public void setProx(Node prox) { this.prox = prox; };
27     public void setId(int id) { this.id = id;   };
28   public void setNome(String nome){ this.nome = nome;}
29
30 }
```