Fundação CECIERJ - Vice Presidência de Educação Superior a Distância

Curso de Tecnologia em Sistemas de Computação
AD2 de Programação OO
1° semestre de 2018

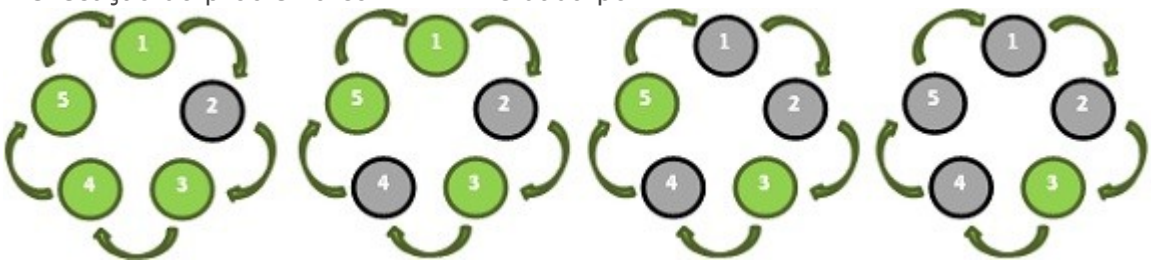
EXERCÍCIO (ENTREGAR OS ARQUIVOS EM MÍDIA, PARA FINS DE TESTE, JUNTAMENTE COM A AD IMPRESSA):

O problema de Josephus é assim conhecido por causa da lenda de Flavius Josephus, um historiador judeu que viveu no século 1. Segundo o relato de Josephus do cerco de Yodfat, ele e seus companheiros foram presos em uma caverna, cuja saída foi bloqueada pelos romanos. Eles preferiram suicidar-se a serem capturados, e decidiram que iriam formar um círculo e começar a matar-se pulando de **k** em **k**. Josephus afirma que, por sorte ou talvez pela mão de Deus, ele permaneceu por último e preferiu entregar-se aos romanos a suicidar-se. Seu programa deve resolver esse problema para qualquer **k** informado. O parâmetro de entrada é um arquivo com os nomes dos participantes do circulo e N entradas para **k (N** e **k** maiores que zero**)**. A resposta será infomada no arquivo **saida-<nome_arquivo_entrada>**, e conterá, para cada um dos **N** valores de **k**, o nome do sobrevivente.

Dados um arquivo de entrada de exemplo a seguir:

```
1
2
3
4
5
FIM   // MARCA DE FIM DOS NOMES
1     // O VALOR DE N
2     // O VALOR DO ÚNICO k DO ARQUIVO
```

A execução do problema com **k** = **2** é dada por:



O arquivo de saída conterá:

2 3

LEMBRE-SE: SEU PROGRAMA SÓ PODE LER O ARQUIVO DE ENTRADA UMA ÚNICA VEZ, E ELE DEVE EXECUTAR COM QUAISQUER DADOS INFORMADOS COMO PARÂMETROS DE ENTRADA. SE O SEU PROGRAMA RESOLVER SOMENTE O EXEMPLO SUPRACITADO, SUA QUESTÃO SERÁ TOTALMENTE DESCONTADA.

RESPOSTA:

```
import java.io.*;
import java.util.*;

class no{
  String nome;
  no prox;

  no(String nome){
    this.nome = nome;
    prox = null;
  }

  public String toString(){ return nome; }
}


class lista{
  no prim;

  lista(){ prim = null; }

  void insere(String nome){
    no novo = new no(nome);
    if(prim == null){
      prim = novo;
      novo.prox = novo;
    }
    else{
      no p = prim.prox;
      while(p.prox != prim){
        if(nome.compareToIgnoreCase(p.nome) == 0) return;
        p = p.prox;
      }
      novo.prox = prim;
      p.prox = novo;
    }
  }

  no busca(String nome){
    if(nome.compareToIgnoreCase(prim.nome) == 0) return prim;
    no p = prim.prox;
    while(p != prim){
      if (nome.compareToIgnoreCase(p.nome) == 0) return p;
      p = p.prox;
    }
    return null;
  }
```

```
  void retira(String nome){
     if(prim == null) return;
      if((nome.compareToIgnoreCase(prim.nome)  ==  0)  &&  (prim.prox  ==
prim)){
       prim = null;
       return;
     }
     no p = prim.prox, ant = prim;
     while((p != prim) && (nome.compareToIgnoreCase(p.nome) != 0)){
       ant = p;
       p = p.prox;
     }
     if((p == prim) && (nome.compareToIgnoreCase(prim.nome) != 0))
       return;
     if(p == prim) prim = prim.prox;
     ant.prox = p.prox;
  }

  public String toString(){
     if(prim == null) return null;
     String resp = prim.toString() + "\n";
     no p = prim.prox;
     while (p != prim){
       resp += p.toString() + "\n";
       p = p.prox;
     }
     return resp;
  }
}


public class AD2_POO_2018_1{
  public static void main(String[] args) throws IOException{
     BufferedReader in;
     in = new BufferedReader(new FileReader(args[0]));
     BufferedWriter out;
     out = new BufferedWriter(new FileWriter("saida-" + args[0]));
     try {
       lista l = new lista();
       String s = in.readLine();
       while(!s.equals("FIM")){
         l.insere(s);
         s = in.readLine();
       }
       String resp = "";
       s = in.readLine();
       int i, N = Integer.parseInt(s), k;
       for(i = 0; i < N; i++){
         s = in.readLine();
         k = Integer.parseInt(s);
         lista aux = Copia(l);
         resp += Josephus(l, aux, k);
       }
```

```
      in.close();
      out.write(resp);
      out.close();
    }
    catch (Exception e){
      System.out.println("Excecao\n");
    }
  }

  static lista Copia(lista origem){
    if(origem == null) return null;
    lista dest = new lista();
    dest.insere(origem.prim.nome);
    no p = origem.prim.prox;
    while(p != origem.prim){
      dest.insere(p.nome);
      p = p.prox;
    }
    return dest;
  }

  static String Josephus(lista original, lista l, int k){
    int j;
    no p = l.prim;
    while(l.prim.prox != l.prim){
      j = 1;
      while(j != k){
        j++;
        p = p.prox;
      }
      System.out.println(p.nome + " morreu...");
      no q = p.prox;
      l.retira(p.nome);
      p = q;
    }

    int num = 1;
    p = original.prim;
    do{
      if(p.nome.compareToIgnoreCase(l.prim.nome) == 0)
        return k + " " + num + "\n";
      num++;
      p = p.prox;
    } while(p != original.prim);
    return null;
  }
}
```