PUC Minas

PONTIFÍCIA UNIVERSIDADE CATÓLICA DE MINAS GERAIS

CURSO: ENGENHARIA DE SOFTWARE

MATÉRIA: ALGORITMOS E ESTRUTURAS DE DADOS II

PROFESSOR: RODRIGO RICHARD

ALUNA: RAÍSSA CAROLINA VILELA DA SILA

BELO HORIZONTE, 25 DE SETEMBRO DE 2018

4) Crie a função CListaDup ConcatenaLD(CListaDup L1, CListaDup L2) **que concatena as listas L1 e L2 passadas por parâmetro, retornando uma lista duplamente encadeada.**

```
public static CListaDup concatena (CListaDup c1, CListaDup c2){
        CListaDup concatenada = new CListaDup( );
        int conc      = 0;
        int Clista1   = c1.quantidade( );
        int Clista2   = c2.quantidade( );
        int amount  = Clista1 + Clista2;

        while (conc != amount){
                if (! c1.vazia( )){
                        concatenada.insereFim(c1.removeRetornaComeco( ));
                        conc ++;
                } else {
                        if (! c2.vazia( )){
                                concatenada.insereFim(c2.removeRetornaComeco( ));
                                conc ++;
                        }//end if
                }//end else
        }//end while
   return concatenada;
}//end concatena( )
```

5) Crie a função CFila ConcatenaFila(CFila F1, Cfila F2) **que concatena as filas F1 e F2 passadas por parâmetro.**

```
public static CFila concatenaFila (CFila f1, CFila f2){
        CFila concatenada = new CFila ( );
        int conc     = 0;
        int q1       = f1.quantidade( );
        int q2       = f2.quantidade( );
        int amount = q1 + q2;

        while (conc != amount){
                if (! f1.vazia( )){
                        concatenada.enfileira(f1.desenfileira( ));
                        conc ++;
                } else {
                        if (! f2.vazia( )){
                                concatenada.enfileira(f2.desenfileira( ));
                                conc ++;
                        }//end if
                }//end else
        }//end while
   return concatenada;
}//end concatenaFila( )
```

6) Crie a função CPilha ConcatenaPilha(CPilha P1, CPilha P2) **que concatena as pilhas P1 e P2 passadas por parâmetro.**

```
public static CPilha concatenaPilha (CPilha p1, CPilha p2){
        CPilha concatenada = new CPilha( );
        int conc     = 0;
        int q1       = p1.quantidade( );
        int q2       = p2.quantidade( );
        int amount = q1 + q2;

        while (conc != amount){
                if (! p1.vazia( )){
                        concatenada.empilha(p1.desempilha( ));
                        conc ++;
                } else {
                        if (! p2.vazia ( )){
                                concatenada.empilha(p2.desempilha( ));
                                conc ++;
                        }//end if
                }//end else
        }//end while
   return concatenada;
}//end concatenaPilha( );
```

*** 7) A class** RandomQueue **é uma Fila que retorna elementos aleatórios ao invés de sempre retornar o primeiro elemento. Crie a classe RandomQueue com os seguintes métodos:**

```
public class RandomQueue {

	private CCelula front;
	private CCelula behind;
	private int amount;

	public RandomQueue ( ){
		front  = new CCelula( );
		behind = front;
	}//end RandomQueue( )

	public boolean isEmpty ( ){
		if ( front == behind ){
			return true;
		} else {
			return false;
		}//end else
	}//end isEmpty( )

	public void Enqueue (Object dado){
		behind.prox = new Ccelula(dado);
		behind = behind.prox;
		amount++;
	}//end Enqueue( )

	public Object Dequeue ( ){
		int random       = (int)(Math.random( )*(this.amount));
		int aux          = 0;
		int quantidade   = this.amount;
		Object dado      = null;
		Object item      = null;
		boolean found    = false;

		while(!found && quantidade > 0){
			if(aux == random){
				if (front != behind){
					front = front.prox;
					item = front.item;
					dado = item;
					amount--;
				}//end if
			}//end if

			if(quantidade > 0){
				if (front != behind) {
					front = front.prox;
					item = front.item;
				}//end if
				this.Enqueue(item);
				quantidade--;
				aux++;
				if(quantidade == 1){
					found = true;
				}//end if
			}//end if
		}//end while
	      return dado;
	}//end Dequeue( )

	public Object Sample ( ){
		int random       = (int)(Math.random( )*(this.amount));
		int aux          = 0;
		int quantidade   = this.amount;
		Object dado      = null;
		Object item      = null;
		boolean found    = false;
```

```
                while(!found && quantidade > 0){
                        if(aux == random){
                                if (front != behind){
                                        front = front.prox;
                                        item = front.item;
                                        dado = item;
                                        this.Enqueue(item);
                                }//end if
                        }//end if

                        if(quantidade > 0){
                                if (front != behind) {
                                        front = front.prox;
                                        item = front.item;
                                }//end if

                                this.Enqueue(item);
                                quantidade--;
                                aux++;
                                if(quantidade == 1){
                                        found = true;
                                }//end if
                        }//end if
                }//end while
            return dado;
        }//end Sample( )
}//end class
```

10) Deque (Double-ended-queue) é um Tipo Abstrato de Dados (TAD) que funciona como uma fila e como uma pilha, permitindo que itens sejam adicionados em ambos os extremos. Implemente a classe Deque, usando duplo encadeamento, com os seguintes métodos:

```
class Deque {

   private CCelulaDup frente;
   private CCelulaDup tras;
   private int qtde;

        public Deque() {
                this.frente = new CCelulaDup();
                this.tras   = this.frente;
        }//end Deque ( )

        public boolean isEmpty() {
                return (this.frente == this.tras);
        }//end isEmpty( )

        public int size () {
                return (this.qtde);
        }//end size ( )

        public void pushLeft(Object valorItem) {
                tras.prox = new CCelulaDup(valorItem, tras, null);
                tras      = tras.prox;
                qtde++;
        }//end pushLeft( )

        public void pushRight(Object valorItem) {
                tras.prox = new CCelulaDup(valorItem, tras, null);
                tras      = tras.prox;
                qtde++;
        }//end pushRight

        public Object popLeft() {
                if(frente != tras) {
                        CCelulaDup aux = frente;

                        while (aux.prox != tras) {
                                aux = aux.prox;
                        }//end while
```

```
                CCelulaDup aux2 = aux.prox;
                tras            = aux;
                tras.prox       = null;
                qtde--;
                return aux2.item;
            }//end if
        return null;
    }//end popLeft ( )

    public Object popRight() {
            Object valorItem = null;

            if(frente != tras) {
                frente    = frente.prox;
                valorItem = frente.item;
                qtde--;
            }//end if
        return (valorItem);
    }//end popRigth ( )

    public void imprime() {
            CCelulaDup aux = frente.prox;

            while (aux != null) {
                System.out.println(“”+aux.item);
                aux = aux.prox;
            }//end while
    }//end imprime ( )

    public int quantidade() {
            return qtde;
    }//end quantidade( )
}//end class
```

13) Crie na Cfila o método int quantidadeOcorrencias (Object elemento) **a qual retorna a quantidade de vezes que o elemento passado como parâmetro está armazenado na CFila.**

```
    public int quantidadeOcorrencias (int i){
            int length       = this.quantidade( );
            int quantidade = 0;
            int dado         = 0;
            CCelula aux      = frente.prox;
            while (length > 0){
                dado = (int) aux.item;
                if (dado == i){
                        quantidade++;
                }//end if
                aux = aux.prox;
                length--;
            }//end while
        return quantidade;
    }//end quantidadeOcorrencias ( )
```

14) Crie na CPilha o método void inverte ( ) **que inverte a ordem dos elementos da Pilha.**

```
    public void inverte ( ){
            CPilha Paux = this;
            Object item = null;
            for (int i = this.quantidade( ); i > 0; i--){
                item = Paux.desempilha( );
                this.empilha(item);
            }//end for
    }//end inverte( )
```

16) Crie na CLista o método Object [ ] copiaParaVetor( ) **que copia todos os elementos da Lista para um vetor.**

```
public Object [ ] copiaParaVetor ( ){
        Object [ ] array = new Object [this.quantidade( )];
        for (int i = 0; i < array.length; i++){
                array[i] = this.removeRetornaComeco( );
        }//end for
    return array;
}//end copiaParaVetor
```

20) Cria o método void Limpar ( ) **para todas as classes** (CLista, CListaDup, Cfila e CPilha)**, o qual deve remover todos os itens da estrutura.**

```
/*
        Metodo responsavel por limpar uma PILHA
*/
        public void limpar( ){
                Object item = null;

                while(topo != null && this.qtde != 0){
                        item = this.desempilha( );
                }//end while

                if(this.vazia( )){
                    System.out.println("Nao ha' elementos nesta pilha.");
                }//end if
         }//end limparPilha( )

/*
        Metodo responsavel por limpar uma FILA
*/
        public void limpar ( ){
                Object item = null;

                while(this.frente != this.tras && this.qtde != 0){
                        item = this.desenfileira( );
                }//end while

                if(this.vazia( )){
                    System.out.println("Nao ha' elementos nesta fila.");
                }//end if
        }//end limpar ( )

/*
        Metodo responsavel por limpar uma LISTA
*/

         public void limpar( ){
                Object item = null;

                while (this.primeira != this.ultima && this.qtde != 0){
                        item = this.removeRetornaFim( );
                }//end while

                if(this.vazia( )){
                    System.out.println("A lista esta' vazia.");
                }//end if
         }//end Limpar ( )

/*
        Metodo responsavel por limpar uma LISTA DUPLA
*/

         public void limpar ( ){
                Object item = null;

                while(this.primeira != this.ultima && this.qtde != 0){
```

```
                item = this.removeRetornaFim( );
            }//end while

            if(this.vazia( )){
                System.out.println("Nao ha' elementos nesta fila dupla.");
            }//end if
    }//end limpar ( )
```

23) Crie a função construtora CFila (CFila F) **na classe CFila que crie a fila com todos os elementos da Fila F recebida como parâmetro.**

```
    public CFila (CFila F){
            frente = F.frente;
            tras = frente;
    }//end CFila
```

24) Crie na classe CLista o método void insereEspelhado(Object item), **o qual insere o elemento no início e no final da lista. Assim, as chamadas para inserir os elementos 1, 2 e 3 deveriam resultar na seguinte lista [3 2 1 1 2 3].**

```
    public void insereEspelhado (Object item){
            Object clone = item;
            primeira.prox = new CCelula(item, primeira.prox);

            if (primeira.prox.prox == null){
                ultima = primeira.prox;
                qtde++;
            }//end if
            ultima.prox = new CCelula(clone);
            ultima = ultima.prox;
            qtde++;
    }//end insereEspelhado ( )
```

26) Crie uma função construtora CPilha (CPilha P) **na classe CPilha que recebe a Pilha P passada como parâmetro e copia todos os seus elementos (sem destruí-la) para a nova pilha que está sendo criada.**

```
    public CPilha (CPilha P){
            topo = P.topo;
    }//end CPilha
```

*** 30) Crie as classes** CcelulaDicionario **e** Cdicionario **conforme a interface abaixo:**

```
class CCelulaDicionario {
    public Object key, value;
    public CCelulaDicionario prox;

    public CCelulaDicionario ( ){
        key = null; value = null; prox = null;
    }//end CCelulaDicionario( )

    public CCelulaDicionario (Object chave, Object valor){
        this.key   = chave;
        this.value = valor;
    }//end CCelulaDicionario ( )

    public CCelulaDicionario (Object chave, Object valor, CCelulaDicionario proxima){
        this.key   = chave;
        this.value = valor;
        this.prox  = proxima;
    }//end CCelulaDicionario ( )
}//end class
```

```
class CDicionario {
        private CCelulaDicionario first, last;

        public CDicionario ( ){
                this.first = new CCelulaDicionario ( );
                this.last  = first;
        }//end CDicionario ( )

        public boolean empty ( ){
                 if (this.first == this.last){
                      return true;
                } else {
                      return false;
                }//end if
        }//end empty ( )

        public void add (Object chave, Object valor){
                this.last.prox = new CCelulaDicionario (chave, valor);
                this.last = last.prox;
        }//end add ( )

        public Object recebaValor (Object chave){

                boolean found = false, achou;
                Object  item  = null;
                CCelulaDicionario aux = this.first.prox;

                while (aux != null && !found){
                        achou = aux.key.equals(chave);

                        if (achou == true){
                                found  = true;
                                item   = aux.value;
                        } else {
                                if (aux.prox == null){
                                        found = true;
                                }//end if
                        }//end else
                        aux = aux.prox;
                }//end while
            return item;
        }//end recebaValor
}//end class

public class Dicionario extends CCelulaDicionario {

   public static Scanner in = new Scanner (System.in);

   public static void main (String [ ] args){
      CDicionario dic = new CDicionario ( );
      int quantidade  = 0;
      String url          = "";
      String ip           = "";
      String dado         = "";

        url = "www.google.com";
        ip  = "74.125.234.81";
        dic.add(url,ip);
        url = "www.pucminas.br";
        ip  = "200.229.32.27";
        dic.add(url,ip);
        url = "www.youtube.com";
        ip  = "216.58.194.206";
        dic.add(url,ip);
        url = "www.capes.gov.br";
        ip  = "200.130.18.222";
        dic.add(url,ip);
        url = "www.yahoo.com";
        ip  = "98.138.219.232";
        dic.add(url,ip);
        url = "www.microsoft.com";
```

```
        ip  = "104.81.49.171";
        dic.add(url,ip);
        url = "www.twitter.com";
        ip  = "104.244.42.65";
        dic.add(url,ip);
        url = "www.brasil.gov.br";
        ip  = "170.246.252.243";
        dic.add(url,ip);
        url = "www.wikipedia.com";
        ip  = "198.35.26.96";
        dic.add(url,ip);
        url = "www.amazon.com";
        ip  = "216.137.36.124";
        dic.add(url,ip);
        url = "research.microsoft.com";
        ip  = "13.67.218.189";
        dic.add(url,ip);
        url = "www.facebook.com";
        ip  = "157.240.22.39";
        dic.add(url,ip);
        url = "www.whitehouse.gov";
        ip  = "104.91.190.179";
        dic.add(url,ip);
        url = "www.answers.com";
        ip  = "151.101.40.203";
        dic.add(url,ip);
        url = "www.uol.com.br";
        ip  = "54.230.147.3";
        dic.add(url,ip);
        url = "www.hotmail.com";
        ip  = "204.79.197.212";
        dic.add(url,ip);
        url = "www.cplusplus.com";
        ip  = "167.114.170.15";
        dic.add(url,ip);
        url = "www.nyt.com";
        ip  = "151.101.41.164";
        dic.add(url,ip);
        url = "www.apple.com";
        ip  = "172.230.107.90";
        dic.add(url,ip);
        url = "www.dropbox.com";
        ip  = "162.125.4.1";
        dic.add(url,ip);
        url = "www.submarino.com.br";
        ip  = "23.53.253.85";
        dic.add(url,ip);
        url = "www.americanas.com";
        ip  = "23.53.253.85";
        dic.add(url,ip);
        url = "www.casasbahia.com.br";
        ip  = "104.80.206.167";
        dic.add(url,ip);
        url = "www.pontofrio.com.br";
        ip  = "104.92.127.204";
        dic.add(url,ip);

      System.out.println ("Deseja o IP de qual site?: ");
      dado = in.next( );
      System.out.println ( "O IP do site '"+dado+"' é: " +dic.recebaValor(dado));
   }//end main
}//end class
```

*** 31) Um biólogo precisa de um programa que traduza uma trinca de nucleotídeos em seu aminoácido correspondente. Por exemplo, a trinca de aminoácidos ACG é traduzida como o aminoácido Treonina, e GCA em Alanina. Crie um programa em Java que use a sua classe CDicionario para criar um dicionário do código genético. O usuário deve digitar uma trinca (chave) e seu programa deve mostrar o nome (valor) do aminoácido correspondente. Use a tabela a seguir para cadastrar todas as trincas/aminoácidos.**

```
public class Nucleotideos extends CCelulaDicionario{

   public static Scanner in = new Scanner (System.in);

        public static void main (String [ ]args){
                CDicionario dic      = new CDicionario ( );
                String trinca        = "";
                String aminoacido = "";
                String dado          = "";
                String max           = "";

                //montar tabela
                //Fenilalanina
                trinca     = "UUU";
                aminoacido = "Fenilalanina";
                dic.add(trinca,aminoacido);
                trinca     = "UUC";
                aminoacido = "Fenilalanina";
                dic.add(trinca,aminoacido);
                //Leucina
                trinca     = "UUA";
                aminoacido = "Leucina";
                dic.add(trinca,aminoacido);
                trinca     = "UUG";
                aminoacido = "Leucina";
                dic.add(trinca,aminoacido);
                trinca     = "CUU";
                aminoacido = "Leucina";
                dic.add(trinca,aminoacido);
                trinca     = "CUC";
                aminoacido = "Leucina";
                dic.add(trinca,aminoacido);
                trinca     = "CUA";
                aminoacido = "Leucina";
                dic.add(trinca,aminoacido);
                trinca     = "CUG";
                aminoacido = "Leucina";
                dic.add(trinca,aminoacido);
                //Isoleucina
                trinca     = "AUU";
                aminoacido = "Isoleucina";
                dic.add(trinca,aminoacido);
                trinca     = "AUC";
                aminoacido = "Isoleucina";
                dic.add(trinca,aminoacido);
                trinca     = "AUA";
                aminoacido = "Isoleucina";
                dic.add(trinca,aminoacido);
                //Metionina
                trinca     = "AUG";
                aminoacido = "Valina";
                dic.add(trinca,aminoacido);
                //Valina
                trinca     = "GUU";
                aminoacido = "Valina";
                dic.add(trinca,aminoacido);
                trinca     = "GUC";
                aminoacido = "Valina";
                dic.add(trinca,aminoacido);
                trinca     = "GUA";
                aminoacido = "Valina";
                dic.add(trinca,aminoacido);
                trinca     = "GUG";
                aminoacido = "Valina";
                dic.add(trinca,aminoacido);
                //Serina
                trinca     = "UCU";
                aminoacido = "Serina";
                dic.add(trinca,aminoacido);
                trinca     = "UCC";
                aminoacido = "Serina";
```

```
dic.add(trinca,aminoacido);
trinca     = "UCA";
aminoacido = "Serina";
dic.add(trinca,aminoacido);
trinca     = "UCG";
aminoacido = "Serina";
dic.add(trinca,aminoacido);
trinca     = "AGU";
aminoacido = "Serina";
dic.add(trinca,aminoacido);
trinca     = "AGC";
aminoacido = "Serina";
dic.add(trinca,aminoacido);
//Prolina
trinca     = "CCU";
aminoacido = "Prolina";
dic.add(trinca,aminoacido);
trinca     = "CCC";
aminoacido = "Prolina";
dic.add(trinca,aminoacido);
trinca     = "CCA";
aminoacido = "Prolina";
dic.add(trinca,aminoacido);
trinca     = "CCG";
aminoacido = "Prolina";
dic.add(trinca,aminoacido);
//Treonina
trinca     = "ACU";
aminoacido = "Treonina";
dic.add(trinca,aminoacido);
trinca     = "ACC";
aminoacido = "Treonina";
dic.add(trinca,aminoacido);
trinca     = "ACA";
aminoacido = "Treonina";
dic.add(trinca,aminoacido);
trinca     = "ACG";
aminoacido = "Treonina";
dic.add(trinca,aminoacido);
//Alanina
trinca     = "GCU";
aminoacido = "Alanina";
dic.add(trinca,aminoacido);
trinca     = "GCC";
aminoacido = "Alanina";
dic.add(trinca,aminoacido);
trinca     = "GCA";
aminoacido = "Alanina";
dic.add(trinca,aminoacido);
trinca     = "GCG";
aminoacido = "Alanina";
dic.add(trinca,aminoacido);
//Tirosina
trinca     = "UAU";
aminoacido = "Tirosina";
dic.add(trinca,aminoacido);
trinca     = "UAC";
aminoacido = "Tirosina";
dic.add(trinca,aminoacido);
//Histidina
trinca     = "CAU";
aminoacido = "Histidina";
dic.add(trinca,aminoacido);
trinca     = "CAC";
aminoacido = "Histidina";
dic.add(trinca,aminoacido);
//Glutamina
trinca     = "CAA";
aminoacido = "Glutamina";
dic.add(trinca,aminoacido);
trinca     = "CAG";
```

```
aminoacido = "Glutamina";
dic.add(trinca,aminoacido);
//Asparagina
trinca     = "AAU";
aminoacido = "Asparagina";
dic.add(trinca,aminoacido);
trinca     = "AAC";
aminoacido = "Asparagina";
dic.add(trinca,aminoacido);
//Lisina
trinca     = "AAA";
aminoacido = "Lisina";
dic.add(trinca,aminoacido);
trinca     = "AAG";
aminoacido = "Lisina";
dic.add(trinca,aminoacido);
//Aspartato
trinca     = "GAU";
aminoacido = "Aspartato";
dic.add(trinca,aminoacido);
trinca     = "GAC";
aminoacido = "Aspartato";
dic.add(trinca,aminoacido);
//Glutamato
trinca     = "GAA";
aminoacido = "Glutamato";
dic.add(trinca,aminoacido);
trinca     = "GAG";
aminoacido = "Glutamato";
dic.add(trinca,aminoacido);
//Cisteina
trinca     = "UGU";
aminoacido = "Cisteina";
dic.add(trinca,aminoacido);
trinca     = "UGC";
aminoacido = "Cisteina";
dic.add(trinca,aminoacido);
//Parada
trinca     = "UAA";
aminoacido = "Parada";
dic.add(trinca,aminoacido);
trinca     = "UAG";
aminoacido = "Parada";
dic.add(trinca,aminoacido);
trinca     = "UGA";
aminoacido = "Parada";
dic.add(trinca,aminoacido);
//Triptofano
trinca     = "UGG";
aminoacido = "Triptofano";
dic.add(trinca,aminoacido);
//Glicina
trinca     = "GGU";
aminoacido = "Glicina";
dic.add(trinca,aminoacido);
trinca     = "GGC";
aminoacido = "Glicina";
dic.add(trinca,aminoacido);
trinca     = "GGA";
aminoacido = "Glicina";
dic.add(trinca,aminoacido);
trinca     = "GGG";
aminoacido = "Glicina";
dic.add(trinca,aminoacido);
//Arginina
trinca     = "CGU";
aminoacido = "Arginina";
dic.add(trinca,aminoacido);
trinca     = "CGC";
aminoacido = "Arginina";
dic.add(trinca,aminoacido);
```

```
            trinca     = "CGA";
            aminoacido = "Arginina";
            dic.add(trinca,aminoacido);
            trinca     = "CGG";
            aminoacido = "Arginina";
            dic.add(trinca,aminoacido);
            trinca     = "AGA";
            aminoacido = "Arginina";
            dic.add(trinca,aminoacido);
            trinca     = "AGG";
            aminoacido = "Arginina";
            dic.add(trinca,aminoacido);

            System.out.println ("Digite a 'trinca' para o aminoacido desejado: ");
            dado = in.next( );
            System.out.println (""+dado.toUpperCase( )+" = "+
                                dic.recebaValor(dado.toUpperCase( )));

        }//end main ( )
}//end class
```

32) Crie a classe ClistaSimples **que é uma lista simples encadeada sem célula cabeça e que possui apenas os métodos definidos na interface abaixo. Atenção: não podem ser acrescentados novos atributos ou métodos às classes CListaSimples e/ou CCelula abaixo:**

```
class CListaSimples {

        private CCelulaSimples primeira, ultima;

        public CListaSimples() {
                this.primeira = null;
                this.ultima = null;
        }//end ClistaSimples ( )

        public boolean vazia() {
                return (this.primeira == null);
        }//end vazia ( )

        public void insereComeco(Object valorItem) {
                if (primeira == null) {
                        primeira = new CCelulaSimples();
                        primeira.item = (int) valorItem;
                        primeira.prox = null;
                        ultima = primeira;
                } else {
                        CCelulaSimples aux = new CCelulaSimples();
                        aux = primeira;
                        primeira = new CCelulaSimples();
                        primeira.item = (int) valorItem;
                        primeira.prox = aux;
                }//end else
        }//end insereComeco( )

        public Object RemoveComeco() {
                if (primeira == ultima) {
                        primeira = null;
                        ultima = null;
                } else {
                        CCelulaSimples aux = primeira;
                        primeira = primeira.prox;
                        return (aux.item);
                }//end else
           return null;
        }//end RemoveComeco( )

        public void insereFim(Object valorItem) {
                if (primeira == null) {
                        primeira = new CCelulaSimples();
                        primeira.item = (int) valorItem;
                        primeira.prox = null;
                        ultima = primeira;
```

```
            } else {
                    ultima.prox = new CCelulaSimples();
                    ultima = ultima.prox;
                    ultima.item = (int) valorItem;
            }//end else
        }//end insereFim ( )

        public Object removeFim() {
            if (primeira == ultima) {
                    primeira = null;
                    ultima = null;
            } else {
                    CCelulaSimples aux = primeira;

                    while (aux.prox != ultima) {
                            aux = aux.prox;
                    }//end while

                    CCelulaSimples aux2 = new CCelulaSimples();
                    aux2 = aux.prox;
                    ultima = aux;
                    ultima.prox = null;
                    return (aux2.item);
            }//end else
           return (null);
        }//end removeFim( )

        public void imprime() {
            CCelulaSimples aux = new CCelulaSimples();
            aux = primeira;

            while (aux != null) {
                    System.out.println("" + aux.item);
                    aux = aux.prox;
            }//end while
        }//end imprime ( )

        public boolean contem(Object valorItem) {
            CCelulaSimples aux = primeira;
            boolean achou = false;

            while (aux != null) {
                    if (aux.item == (int) valorItem) {
                            achou = true;
                            return achou;
                    }//end if
                    aux = aux.prox;
            }//end while
           return achou;
        }//end contem
}//end class
```