PONTIFÍCIA UNIVERSIDADE CATÓLICA DE MINAS GERAIS
INSTITUTO DE CIÊNCIAS EXATAS E INFORMÁTICA
UNIDADE EDUCACIONAL PRAÇA DA LIBERDADE
Bacharelado em Engenharia de Software

Ravi Antônio Gonçalves de Assis

LISTA DE EXERCÍCIOS DE ALGORITMOS E ESTRUTURAS DE DADOS II

Belo Horizonte
2018

1 – Crie na CLista o método void InsereAntesDe(Object ElementoAInserir, Object Elemento) que insere o ElementoAInserir na posição anterior ao Elemento passado por parâmetro.

```
public void InsereAntesDe(Object ElementoAInserir, Object Elemento) {
	CCelula aux = primeira;
	while (aux != null) {
		if (aux.prox.item.equals(Elemento)) {
			aux.prox = new CCelula(ElementoAInserir, aux.prox);
			this.qtde++;
			return;
		}
		else aux = aux.prox;
	}
}
```

2 – Crie na CLista o método void InsereDepoisDe(Object ElementoAInserir, Object Elemento) que insere o ElementoAInserir na posição anterior ao Elemento passado por parâmetro.

```
public void InsereDepoisDe(Object ElementoAInserir, Object Elemento) {
		CCelula atual = primeira.prox;
		while (atual != null) {
			if ( atual.item.equals(Elemento) ) {
				atual.prox = new
CCelula(ElementoAInserir,atual.prox);
				if(atual.prox.prox == null) {
					this.ultima = atual.prox;
				}
				this.qtde++;
				return;
			}
			else atual = atual.prox;
		}
   }
```

3 – Crie na CLista o método void InsereOrdenado(int ElementoAInserir) que insere ElementoAInserir em ordem crescente (perceba que para funcionar corretamente, todos os elementos precisarão, necessariamente, ser inseridos através desse método).

```
public void InsereOrdenado(int elemento) {
        CCelula atual = primeira.prox;
        while (atual != null) {
              if ( (int) atual.item < elemento) {
                    atual.prox = new CCelula(elemento, atual.prox);
                    if (atual.prox == null) {
                          ultima = atual;
                    }
                    this.qtde++;
                    return;
              }
              else atual = atual.prox;
        }
        insereComeco(elemento);
   }
```

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

```
private static CListaDup ConcatenaArrayList(CListaDup a, CListaDup b)
{
            // TODO Auto-generated method stub
            CListaDup x = new CListaDup();
            for (int i = 1; i <= a.quantidade(); i++)
                  x.insereFim(a.retornaIndice(i));
            for (int i = 1; i <= b.quantidade(); i++)
                  x.insereFim(b.retornaIndice(i));
            return x;
}
```

5 – Crie na CLista um método recursivo para fazer a impressão dos elementos em ordem invertida.

```
public void imprimeInversoRecursivo () {
        imprimeInversoRecursivo(this.primeira.prox);
}
private void imprimeInversoRecursivo(CCelula aux) {
        if (aux != null) {
              imprimeInversoRecursivo(aux.prox);
              System.out.println(aux.item);
```

```
        }
        return;
}
```

6 – Crie na CLista um método recursivo para retornar um vetor contendo os elementos em ordem invertida.

```
/*Ex6*/
public int [] retornaVetorInvertidoRecursivo() {
      String str = retornaVetorInvertidoRecursivo(this.primeira.prox);
      char [] c = str.toCharArray();
      int [] vetor = new int[c.length];
      for (int i = 0; i < vetor.length; i++) {
            vetor[i] = Character.getNumericValue(c[i]);
            System.out.println(vetor[i]);
      }
      return vetor;
}
private String retornaVetorInvertidoRecursivo(CCelula aux) {
      // TODO Auto-generated method stub
      if (aux != null) {
            return retornaVetorInvertidoRecursivo(aux.prox) + aux.item;
      }
      return "";
}
```

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

class RandomQueue { RandomQueue() { } // Construtora – cria uma RandomQueue vazia bool IsEmpty() { } // Retorna true se a RandomQueue estiver vazia void Enqueue(Object item) { } // Adiciona um item Object Dequeue() { } // Remove e retorna um elemento aleatório da RandomQueue Object Sample() { } // Retorna um elemento aleatório sem removê-lo da RandomQueue }

Exemplo de uso da classe RandomQueue:

RandomQueue RQ = new RandomQueue(); for(int i = 1; i <= 5; i++) RQ.Enqueue(i); System.out.print("Remove e retorna um elemento qualquer = ", RQ.Dequeue()); System.out.print("\nRetorna um elemento sem remover = ", RQ.Sample());

```
public class RandomQueue {
	private CCelula primeira;  // Referencia a célula cabeça
	private CCelula ultima; // Referencia a última célula da lista
	private int qtde;
	RandomQueue(){
		this.ultima = this.primeira = new CCelula();
		this.qtde = 0;
	}

	public boolean isEmpty() {
		// TODO Auto-generated method stub
		return this.primeira == this.ultima;
	}

	public void enqueue(Object item) {
		// TODO Auto-generated method stub
		this.ultima.prox = new CCelula(item);
		this.ultima = ultima.prox;
		this.qtde++;
	}

	public Object sample() {
		// TODO Auto-generated method stub
		if ( this.isEmpty() ) return null;
		else if ( this.qtde == 1) return ultima.item;
		else {
			int index = this.indexRandom();
			CCelula atual = this.primeira.prox;
			CCelula anterior = this.primeira;
			for (int i = 0; i < index; i++) {
				atual = atual.prox;
				anterior = anterior.prox;
			}
			return atual.item;
		}
	}

	public Object dequeue() {
		// TODO Auto-generated method stub
		if( this.isEmpty() ) return null;
		else if ( this.qtde == 1) {
```

```
                CCelula aux = ultima;
                ultima = primeira;
                this.qtde--;
                aux.prox = null;
                return aux.item;
          }
          else {
                int index = this.indexRandom();
                CCelula atual = this.primeira.prox;
                CCelula anterior = this.primeira;
                for (int i = 0; i < index; i++) {
                      atual = atual.prox;
                      anterior = anterior.prox;
                }
                anterior.prox = atual.prox;
                if (atual.prox == ultima.prox) ultima = anterior;
                atual.prox = null;
                this.qtde--;
                return atual.item;
          }
    }

    public int indexRandom() {
          if (this.qtde == 0) return 0;
          else {
                return 1 + (int)(Math.random() * (this.qtde -1));
          }
    }
}
```

8 – Crie na CListaDup o método int primeiraOcorrenciaDe(Object elemento) que busca e retorna o índice da primeira ocorrência do elemento passado por parâmetro. Caso o elemento não exista, sua função deve retornar um valor negativo. Obs: considere que o primeiro elemento está na posição 1.

```
public int primeiraOcorrenciaDe(Object item) {
    // TODO Auto-generated method stub
    CCelulaDup aux = this.primeira.prox;
    int i = 1;
    while (aux != null) {
          if (aux.item.equals(item)) return i;
          aux = aux.prox;
          i++;
```

```
	}
	return -1;
}
```

9 – Crie na CListaDup o método int ultimaOcorrenciaDe(Object elemento) que busca e retorna o índice da ultima ocorrência do elemento passado por parâmetro. Caso o elemento não exista, sua função deve retornar um valor negativo. Obs: considere que o primeiro elemento está na posição 1.

```
public int ultimaOcorrenciaDe(Object item) {
	// TODO Auto-generated method stub
	CCelulaDup aux = this.ultima;
	int i = this.quantidade();
	while (aux != null) {
		if (aux.item.equals(item)) return i;
		aux = aux.ant;
		i--;
	}
	return -1;
}
```

* 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 { Deque() { } // Construtora – cria uma Deque vazia boolean isEmpty() { } // Retorna true se a Deque estiver vazia int size() { } // Retorna a quantidade de itens da Deque void pushLeft(Object item) { } // Adiciona um item no lado esquerdo da Deque void pushRight(Object item) { } // Adiciona um item no lado direito da Deque Object popLeft() { } // Remove e retorna um item do lado esquerdo da Deque Object popRight() { } // Remove e retorna um item do lado direito da Deque

```
public class Deque {
	private CCelulaDup left;
	private CCelulaDup right;
	private int qntd;
	Deque(){
		this.left = this.right = new CCelulaDup();
		this.qntd = 0;
	}
	public boolean isEmpty() {
		// TODO Auto-generated method stub
		return this.left == this.right;
```

```
	}
	public void pushLeft(Object item) {
		// TODO Auto-generated method stub
		if ( this.isEmpty() ) this.pushRight(item);
		else {
			this.left.prox = new CCelulaDup(item, this.left,
this.left.prox);
			this.left.prox.prox.ant = this.left.prox;
			this.qntd++;
		}
	}
	public void pushRight(Object item) {
		this.right.prox = new CCelulaDup(item, right, null);
		this.right = this.right.prox;
		this.qntd++;
	}
	public int size() {
		// TODO Auto-generated method stub
		return this.qntd;
	}
	public Object popLeft() {
		// TODO Auto-generated method stub
		if (this.isEmpty()) {
			return null;
		}
		else if ( this.left.prox.prox == null) return
this.popRight();
		else {
			CCelulaDup cel = this.left.prox;
			this.left.prox = this.left.prox.prox;
			this.left.prox.ant = this.left;
			cel.prox = cel.ant = null;
			this.qntd--;
			return cel.item;
		}
	}
	public Object popRight() {
		// TODO Auto-generated method stub
		if( this.isEmpty() ) return null;
		else {
			CCelulaDup cel = this.right;
```

```
                this.right = this.right.ant;
                this.right.prox = null;
                cel.prox = cel.ant = null;
                this.qntd--;
                return cel.item;
            }
        }
}
```

11 – Crie na CLista o método void RemovePos(int n) que remove o elemento na n-ésima posição da lista.

```
public Object retornaIndice(int posicao) {
      if ((posicao >= 1) && (posicao <= this.qtde) && (this.primeira
!= this.ultima)) {
CCelula aux = this.primeira.prox;
for (int i = 1; i < posicao; i++, aux = aux.prox) ;
if (aux != null)   return aux.item;
}
            return null;
}
public void removePos(int i) {
      this.remove(this.retornaIndice(i));
}
```

12 – Crie na CListaDup o método void RemovePos(int n) que remove o elemento na n-ésima posição da lista.

```
public void removePos(int index) {
      this.remove(this.retornaIndice(index));
}
```

30 – Crie as classes CCelulaDicionario e CDicionario conforme a interface abaixo.

Agora usando sua classe CDicionario, crie um dicionário com URL's e IP's dos websites abaixo e mais 5 à sua escolha. O seu dicionário deve ser implementado usando a classe Hashtable e terá a URL como chave e o IP correspondente como valor (por exemplo, se digitarmos como chave a URL www.google.com, seu programa deve retornar o IP 74.125.234.81). O seu programa deve permitir que o usuário digite uma URL e deve imprimir o IP correspondente. Para descobrir o IP de um website, basta digitar ping + URL do website (exemplo: ping www.google.com).

```
public class Ex30_Aplication {
      public static void comandoErrado(String comando) {
```

```
            System.out.println("\'" + comando + "\' is not recognized
as an internal command.");
            System.out.println();
      }
      public static void printLine() {
            System.out.println("-----------------------------------
-----------------------------------");
      }
      public static void help() {
            printLine();
            System.out.println("Command's list:");
            System.out.println("Command: ping");
            System.out.println("\tUse: ping URL\tEx:ping
www.google.com.br");
            System.out.println("\tDescription: Return the IP adress of
the URL requested.");
            System.out.println();
            System.out.println("Command: exit");
            System.out.println("\tUse: exit");
            System.out.println("\tDescription: Close the program.");
            printLine();
            System.out.println();
      }
      public static String [] trataComando(String comando) {
            return comando.split(" ");
      }
      public static void main(String[] args) throws IOException {
            // TODO Auto-generated method stub
            CDicionario dc = new CDicionario();
            Scanner ext = new Scanner(System.in);
            String [] comands;
            dc.adiciona("www.google.com.br", "172.217.28.67");
            dc.adiciona("www.pucminas.br", "200.229.32.27");
            dc.adiciona("www.gmail.com", "172.217.28.69");
            dc.adiciona("www.youtube.com", "216.58.202.14");
            dc.adiciona("www.capes.gov.br", "200.130.18.222");
            dc.adiciona("www.yahoo.com", "98.136.96.140");
            dc.adiciona("www.microsoft.com", "23.37.248.39");
            dc.adiciona("www.twitter.com", "104.244.42.129");
            dc.adiciona("www.brasil.gov.br", "170.246.252.243");
            dc.adiciona("www.wikipedia.com", "208.80.154.224");
```

```
            dc.adiciona("www.amazon.com", "52.85.163.111");
            dc.adiciona("research.microsoft.com", "13.67.218.189");
            dc.adiciona("www.facebook.com", "31.13.74.35");
            dc.adiciona("www.whitehouse.gov", "23.44.178.233");
            dc.adiciona("www.answers.com", "151.101.176.203");
            dc.adiciona("www.uol.com.br", "200.221.2.45");
            dc.adiciona("www.hotmail.com", "204.79.197.212");
            dc.adiciona("www.cplusplus.com", "167.114.170.15");
            dc.adiciona("www.nyt.com", "151.101.177.164");
            dc.adiciona("www.apple.com", "23.44.189.23");
            dc.adiciona("www.gamespot.com", "64.30.228.81");
            dc.adiciona("pt.stackoverflow.com", "151.101.193.69");
            dc.adiciona("github.com", "192.30.253.113");
            dc.adiciona("www.w3schools.com", "192.229.173.207");
            dc.adiciona("bitcoin.org", "138.68.248.245");
            System.out.println("Welcome to the URL's Dictionary");
            System.out.println("Type the command or --h for help
screen.");
            do {
                  comands = null;
                  System.out.print("c-> ");
                  comands = ext.nextLine().split(" ");
                  switch(comands[0]) {
                        case "--h":
                              help();
                              break;
                        case "ping":

      System.out.println(pesquisarUrl(comands[1], dc));
                              System.out.println();
                              break;
                        case "exit":
                              break;
                        default:
                              comandoErrado(comands[0]);
                              break;
                  }
            } while( comands[0].compareToIgnoreCase("exit") != 0);
            ext.close();
      }
      private static String pesquisarUrl(String url, CDicionario dc) {
```

```
            // TODO Auto-generated method stub
            Object val = dc.recebeValor(url);
            if ( val != null) {
                  return "IP " + (String)val;
            }
            return "Could not find Host IP \'" + url + "\'";
      }
}
```

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 Ex31_Aplication {
      public static void main (String[] args) {
            CDicionario gc = new CDicionario();
            Scanner ext = new Scanner(System.in);
            String [] comands;

            gc.adiciona("UUU", "Fenilalanina");
            gc.adiciona("UUC", "Fenilalanina");
            gc.adiciona("UUA", "Leucina");
            gc.adiciona("UUG", "Leucina");
            gc.adiciona("CUU", "Leucina");
            gc.adiciona("CUC", "Leucina");
            gc.adiciona("CUA", "Leucina");
            gc.adiciona("CUG", "Leucina");
            gc.adiciona("AUU", "Isoleucina");
            gc.adiciona("AUC", "Isoleucina");
            gc.adiciona("AUA", "Isoleucina");
            gc.adiciona("AUG", "Metionina");
            gc.adiciona("GUU", "Valina");
            gc.adiciona("GUC", "Valina");
            gc.adiciona("GUA", "Valina");
            gc.adiciona("GUG", "Valina");
            gc.adiciona("UCU", "Serina");
            gc.adiciona("UCC", "Serina");
            gc.adiciona("UCA", "Serina");
```

```
gc.adiciona("UCG", "Serina");
gc.adiciona("CCU", "Prolina");
gc.adiciona("CCC", "Prolina");
gc.adiciona("CCA", "Prolina");
gc.adiciona("CCG", "Prolina");
gc.adiciona("ACU", "Treonina");
gc.adiciona("ACC", "Treonina");
gc.adiciona("ACA", "Treonina");
gc.adiciona("ACG", "Treonina");
gc.adiciona("GCU", "Alanina");
gc.adiciona("GCC", "Alanina");
gc.adiciona("GCA", "Alanina");
gc.adiciona("GCG", "Alanina");
gc.adiciona("UAU", "Tirosina");
gc.adiciona("UAC", "Tirosina");
gc.adiciona("UAA", "Parada");
gc.adiciona("UAG", "Parada");
gc.adiciona("CAU", "Histidina");
gc.adiciona("CAC", "Histidina");
gc.adiciona("CAA", "Glutamina");
gc.adiciona("CAG", "Glutamina");
gc.adiciona("AAU", "Asparagina");
gc.adiciona("AAC", "Asparagina");
gc.adiciona("AAA", "Lisina");
gc.adiciona("AAG", "Lisina");
gc.adiciona("GAU", "Aspartato");
gc.adiciona("GAC", "Aspartato");
gc.adiciona("GAA", "Glutamato");
gc.adiciona("GAG", "Glutamato");
gc.adiciona("UGU", "Cisteína");
gc.adiciona("UGC", "Cisteína");
gc.adiciona("UGA", "Parada");
gc.adiciona("UGG", "Triptofano");
gc.adiciona("CGU", "Arginina");
gc.adiciona("CGC", "Arginina");
gc.adiciona("CGA", "Arginina");
gc.adiciona("CGG", "Arginina");
gc.adiciona("AGU", "Serina");
gc.adiciona("AGC", "Serina");
gc.adiciona("AGA", "Arginina");
gc.adiciona("AGG", "Arginina");
```

```
            gc.adiciona("GGU", "Glicina");
            gc.adiciona("GGC", "Glicina");
            gc.adiciona("GGA", "Glicina");
            gc.adiciona("GGG", "Glicina");

            System.out.println("Welcome to the Genect Code
Dictionary");
            System.out.println("Type the command or --h for help
screen.");

            do {
                  comands = null;

                  System.out.print("c-> ");
                  comands = ext.nextLine().split(" ");

                  switch(comands[0]) {
                        case "--h":
                              help();
                              break;
                        case "code":
                              System.out.println(search(comands[1],
gc));
                              System.out.println();
                              break;
                        case "exit":
                              break;
                        default:
                              comandoErrado(comands[0]);
                              break;
                  }

            } while( comands[0].compareToIgnoreCase("exit") != 0);

            ext.close();
      }

      private static String search(String trinca, CDicionario gc) {
            Object val = gc.recebeValor(trinca.toUpperCase());
            if ( val != null) {
                  return "Amino acid " + (String)val;
```

```
            }
            return "Could not find the name of the amino acid of the
requested genetic code";
      }

      public static void printLine() {
            System.out.println("----------------------------------------
-----------------------------------");
      }

      public static void help() {
            printLine();
            System.out.println("Command's list:");
            System.out.println("Command: code");
            System.out.println("\tUse: code \"Genect Code\"\tEx:code
UUU");
            System.out.println("\tDescription: Return the amino acid
name of the code requested.");
            System.out.println();
            System.out.println("Command: exit");
            System.out.println("\tUse: exit");
            System.out.println("\tDescription: Close the program.");
            printLine();
            System.out.println();

      }

      public static void comandoErrado(String comando) {
            System.out.println("\'" + comando + "\' is not recognized
as an internal command.");
            System.out.println();
      }
}
```

32 – Crie a classe CListaSimples que é uma lista simplesmente 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.

```
public class CCelulaSimples {
      public int item;
      public CCelulaSimples prox;
}
```

```
public class CListaSimples {
	private CCelulaSimples primeira, ultima;

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

	public boolean vazia() {
		return this.primeira == null && this.ultima == null ;
	}

	public void insereComeco(int valorItem) {
		if(this.vazia()) {
			this.primeira = new CCelulaSimples();
			this.ultima = this.primeira;
			this.primeira.item = valorItem;
			this.primeira.prox = null;
		}
		else {
			CCelulaSimples tmp = new CCelulaSimples();
			tmp.item = valorItem;
			tmp.prox = this.primeira;
			this.primeira = tmp;
		}
	}

	public Object removeComeco() {
		if (this.vazia()) return null;
		else {
			CCelulaSimples resp = new CCelulaSimples();
			resp = this.primeira;
			this.primeira = this.primeira.prox;
			if (this.primeira == null) this.ultima = null;
			return resp.item;
		}
	}

	public void insereFim(int valorItem) {
		if (this.vazia()) {
			this.primeira = new CCelulaSimples();
			this.ultima = this.primeira;
```

```
            this.primeira.item = valorItem;
            this.primeira.prox = null;
      }
      else {
            CCelulaSimples aux = new CCelulaSimples();
            aux.item = valorItem;
            aux.prox = null;
            this.ultima.prox = aux;
            this.ultima = this.ultima.prox;
      }
}

public Object removeFim() {
      if (this.vazia()) return null;
      else if (this.primeira == this.ultima) {
            CCelulaSimples resp = this.primeira;
            this.primeira = this.ultima = null;
            return resp.item;
      }
      else {
            CCelulaSimples aux = this.primeira;
            CCelulaSimples resp ;
            while (aux.prox != this.ultima) {
                  aux = aux.prox;

            }
            resp = this.ultima;
            this.ultima = aux;
            this.ultima.prox = null;
            return resp.item;
      }

}

public void imprime() {
      CCelulaSimples tmp = this.primeira;
      while (tmp != null) {
            System.out.print(tmp.item + " ");
            tmp = tmp.prox;
      }
}
```

```
	public boolean contem(Object elemento) {
		if (this.vazia()) return false;
		else {
			CCelulaSimples tmp = this.primeira;
			while(tmp != null) {
				if (elemento.equals(tmp.item)) return true;
				tmp = tmp.prox;
			}
			return false;
		}
	}
}
```