

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 class Exercicio1 {

    public static void main(String[] args) {

        CLista lista = new CLista();

        lista.insereComeco(1);
        lista.insereComeco(2);
        lista.insereComeco(3);
        lista.insereComeco(4);
        lista.insereComeco(5);
        lista.imprimeFormatoLista();

        System.out.println("\nInsere valor 8 antes do elemento 5:");
        lista.insereAntesDe(8, 2);
        lista.imprimeFormatoLista();

        System.out.println("\nInsere valor 9 antes do elemento 4:");
        lista.insereAntesDe(9, 4);
        lista.imprimeFormatoLista();
    }
}

public class CLista {

    [...]

    public void insereAntesDe(Object elementoAInserir, Object elemento) {
        int indiceAux = 0;

        boolean achou = false;

        CCelula aux = primeira.prox;
        while (aux != null && !achou) {
            achou = aux.item.equals(elemento);
            aux = aux.prox;
            indiceAux++;
        }
        insereIndice(elementoAInserir, indiceAux);
    }

    [...]
}
```

Saída do Programa

```
<terminated> Exercicio1 (1) [Java Application] C:\Program Files\Java\jdk-11.0.2\bin\jav
[/]->[5]->[4]->[3]->[2]->[1]->null

Insere valor 8 antes do elemento 5:
[/]->[5]->[4]->[3]->[8]->[2]->[1]->null

Insere valor 9 antes do elemento 4:
[/]->[5]->[9]->[4]->[3]->[8]->[2]->[1]->null
```

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

```
public class Exercicio2 {

    public static void main(String[] args) {

        CLista lista = new CLista();
        lista.insereComeco(1);
        lista.insereComeco(2);
        lista.insereComeco(3);
        lista.insereComeco(4);
        lista.insereComeco(5);
        lista.imprimeFormatoLista();

        System.out.println("\nInserir 9 depois do elemento 1:");
        lista.insereDepoisDe(9, 1);
        lista.imprimeFormatoLista();

        System.out.println("\nInserir -3 depois do elemento 4:");

        lista.insereDepoisDe(-3, 4);
        lista.imprimeFormatoLista();

    }
}

public class CLista {

    [...]

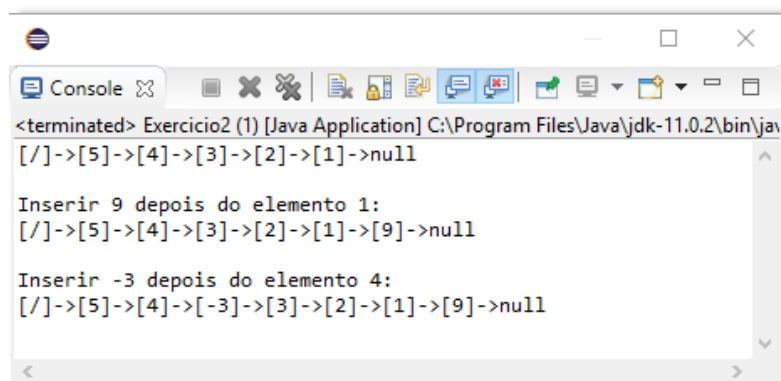
    public void insereDepoisDe(Object elementoAInserir, Object elemento) {
        int indiceAux = 0;
        boolean achou = false;

        CCelula aux = primeira.prox;
        while (aux != null && !achou) {
            achou = aux.item.equals(elemento);
            aux = aux.prox;
            indiceAux++;
        }
        if (achou) {
            insereIndice(elementoAInserir, indiceAux + 1);
        }
        // Se caso o elemento for o ultimo, inserirá através do método insereFim();
        if (indiceAux == quantidade())
            insereFim(elementoAInserir);
    }

    [...]

}
```

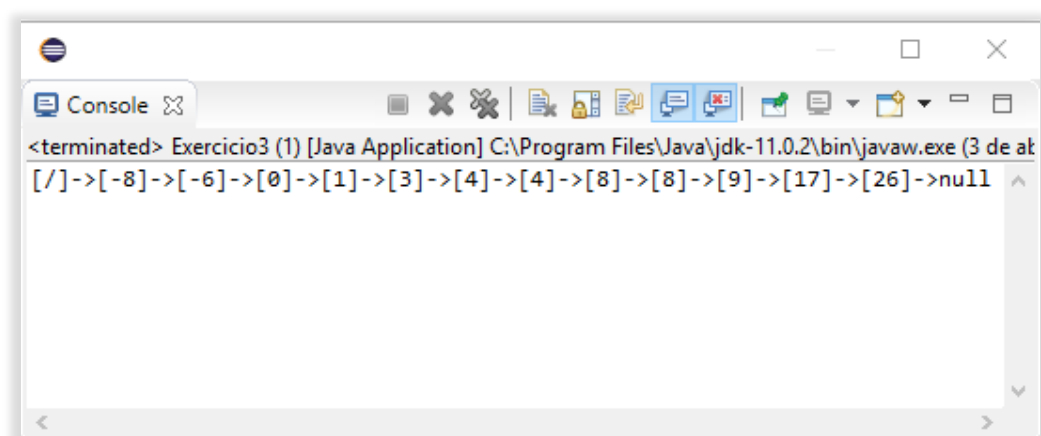
Saída do Programa



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 class CLista {  
    [...]  
    public void insereOrdenado(int elementoAInserir) {  
        if (vazia()) {  
            insereComeco(elementoAInserir);  
            maiorElementoDaLista = elementoAInserir;  
            menorElementoDaLista = elementoAInserir;  
        } else if (elementoAInserir >= maiorElementoDaLista) {  
            insereDepoisDe(elementoAInserir, maiorElementoDaLista);  
            maiorElementoDaLista = elementoAInserir;  
        } else if (elementoAInserir <= menorElementoDaLista) {  
            insereAntesDe(elementoAInserir, menorElementoDaLista);  
            menorElementoDaLista = elementoAInserir;  
        } else if (contem(elementoAInserir)) {  
            insereAntesDe(elementoAInserir, elementoAInserir);  
        }  
    }  
    [...]  
}  
  
public class Exercicio3 {  
    public static void main(String[] args) {  
        CLista lista1 = new CLista();  
  
        lista1.insereOrdenado(3);  
        lista1.insereOrdenado(1);  
        lista1.insereOrdenado(4);  
        lista1.insereOrdenado(0);  
        lista1.insereOrdenado(8);  
        lista1.insereOrdenado(17);  
        lista1.insereOrdenado(14);  
        lista1.insereOrdenado(-6);  
        lista1.insereOrdenado(4);  
        lista1.insereOrdenado(26);  
        lista1.insereOrdenado(6);  
        lista1.imprimeFormatoLista();  
    }  
}
```

Saída do Programa



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

```
public static CListaDup ConcatenaLD(CListaDup L1, CListaDup L2) {
    CListaDup AmaisB = new CListaDup();
    CListaDup auxL1 = L1;
    CListaDup auxL2 = L2;

    int tamL1 = L1.quantidade();
    int tamL2 = L2.quantidade();

    for (int i = 0; i < tamL1; i++) {
        Object a = auxL1.removeRetornaComeco();
        AmaisB.insereFim(a);
    }

    for (int j = 0; j < tamL2; j++) {
        Object b = auxL2.removeRetornaComeco();
        AmaisB.insereFim(b);
    }
    return AmaisB;
}
```

```
public static void main(String[] args) {
    CListaDup lista1 = new CListaDup();
    CListaDup lista2 = new CListaDup();

    for (int i = 6; i > 0; i--) {
        lista1.insereComeco(i);
    }

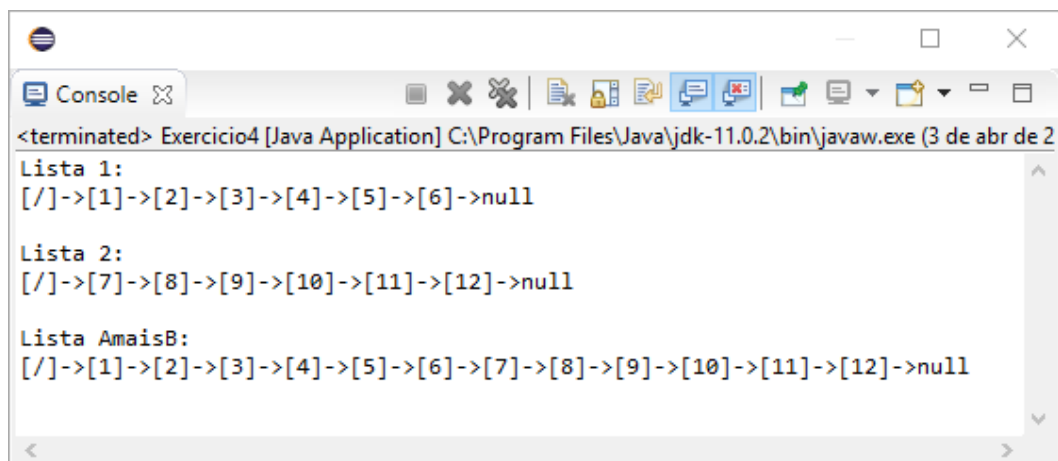
    System.out.println("Lista 1:");
    lista1.imprimeFormatoLista();

    for (int i = 12; i >= 7; i--) {
        lista2.insereComeco(i);
    }
    System.out.println("\nLista 2:");
    lista2.imprimeFormatoLista();

    CListaDup AmaisB;
    AmaisB = ConcatenaLD(lista1, lista2);

    System.out.println("\nLista AmaisB:");
    AmaisB.imprimeFormatoLista();
}
```

Saída do Programa



```
<terminated> Exercicio4 [Java Application] C:\Program Files\Java\jdk-11.0.2\bin\javaw.exe (3 de abr de 2
Lista 1:
[/]->[1]->[2]->[3]->[4]->[5]->[6]->null

Lista 2:
[/]->[7]->[8]->[9]->[10]->[11]->[12]->null

Lista AmaisB:
[/]->[1]->[2]->[3]->[4]->[5]->[6]->[7]->[8]->[9]->[10]->[11]->[12]->null
```

5 – Crie a função `CFilaConcatenaFila(CFila F1, CFila F2)` que concatena as filas F1 e F2 passadas por parâmetro.

```
public static CFila ConcatenaFila(CFila F1, CFila F2) {
    CFila F1maisF2 = new CFila();
    CFila auxF1 = F1;
    CFila auxF2 = F2;

    int tamF1 = F1.quantidade();
    int tamF2 = F2.quantidade();

    for (int i = 0; i < tamF1; i++) {
        F1maisF2.enqueue(auxF1.dequeue());
    }

    for (int j = 0; j < tamF2; j++) {
        F1maisF2.enqueue(auxF2.dequeue());
    }
    return F1maisF2;
}
```

```
public static void main(String[] args) {

    CFila fila1 = new CFila();
    CFila fila2 = new CFila();

    for (int i = 1; i <= 4; i++) {
        fila1.enqueue(i);
    }

    System.out.println("Fila 1:");
    fila1.mostra();

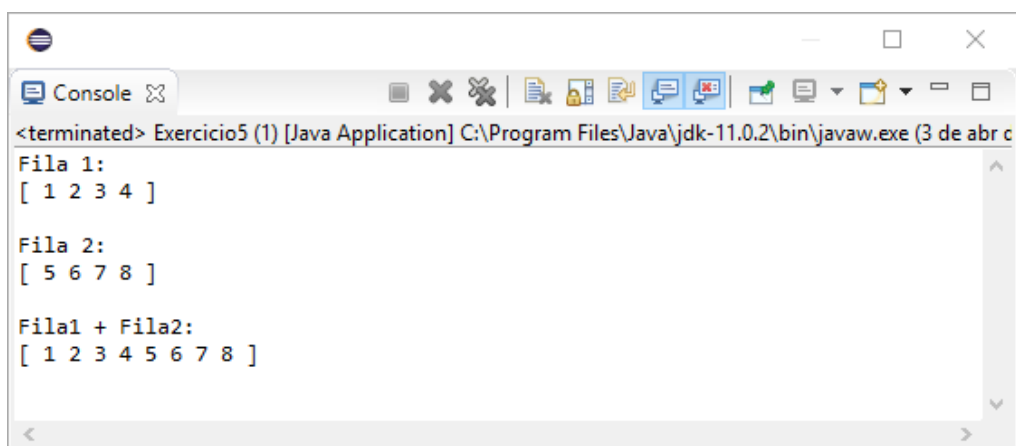
    for (int i = 5; i <= 8; i++) {
        fila2.enqueue(i);
    }

    System.out.println("\nFila 2:");
    fila2.mostra();

    CFila F1maisF2;
    F1maisF2 = ConcatenaFila(fila1, fila2);

    System.out.println("\nFila1 + Fila2:");
    F1maisF2.mostra();
}
```

Saída do Programa



The screenshot shows a Java application window titled "Exercicio5 (1) [Java Application]". The console output is as follows:

```
<terminated> Exercicio5 (1) [Java Application] C:\Program Files\Java\jdk-11.0.2\bin\javaw.exe (3 de abr c
Fila 1:
[ 1 2 3 4 ]

Fila 2:
[ 5 6 7 8 ]

Fila1 + Fila2:
[ 1 2 3 4 5 6 7 8 ]
```

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 P1maisP2 = new CPilha();
    CLista listaAux = new CLista();

    int tamP1 = P1.quantidade();
    int tamP2 = P2.quantidade();

    for (int i = 0; i < tamP2; i++) {
        listaAux.insereComeco(P2.desempilha());
    }

    for (int i = 0; i < tamP2; i++) {
        P1maisP2.empilha(listaAux.removeRetornaComeco());
    }

    for (int i = 0; i < tamP1; i++) {
        listaAux.insereComeco(P1.desempilha());
    }

    for (int i = 0; i < tamP1; i++) {
        P1maisP2.empilha(listaAux.removeRetornaComeco());
    }
    return P1maisP2;
}

public static void main(String[] args) {
    CPilha pilha1 = new CPilha();
    CPilha pilha2 = new CPilha();

    for (int i = 6; i >= 1; i--) {
        pilha1.empilha(i);
    }

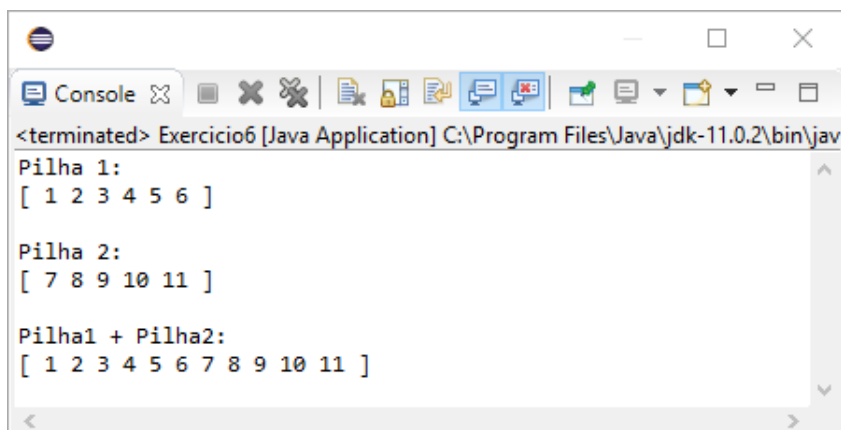
    for (int i = 11; i >= 7; i--) {
        pilha2.empilha(i);
    }

    System.out.println("Pilha 1:");
    pilha1.mostra();

    System.out.println("\nPilha 2:");
    pilha2.mostra();

    CPilha P1maisP2;
    P1maisP2 = ConcatenaPilha(pilha1, pilha2);
    System.out.println("\nPilha1 + Pilha2:");
    P1maisP2.mostra();
}
```

Saída do Programa



```
<terminated> Exercicio6 [Java Application] C:\Program Files\Java\jdk-11.0.2\bin\jav
Pilha 1:
[ 1 2 3 4 5 6 ]

Pilha 2:
[ 7 8 9 10 11 ]

Pilha1 + Pilha2:
[ 1 2 3 4 5 6 7 8 9 10 11 ]
```

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:

```
public class RandomQueue {
    private CCelula frente; // Celula cabeca.
    private CCelula tras;
    private int qtde;

    public RandomQueue() {
        frente = new CCelula();
        tras = frente;
    }

    public boolean isEmpty() {
        return frente == tras;
    }

    public void enqueue(Object item) {
        tras.prox = new CCelula(item);
        tras = tras.prox;
        qtde++;
    }

    public Object Dequeue() {
        Random random = new Random();
        int num = random.nextInt(qtde);
        int index = 1;
        Object item = null;

        for (CCelula c = frente.prox; c != null; c = c.prox) {

            if (num == 0) {
                item = frente.prox.item;
                frente.prox = frente.prox.prox;
                return item;
            }

            if (index == num) {
                if (frente != tras) {
                    item = c.prox.item;
                    c.prox = c.prox.prox;
                    qtde--;
                }
                return item;
            }
            index++;
        }
        return item;
    }

    public Object Sample() {
        Random random = new Random();
        int num = random.nextInt(qtde);
        int index = 1;
        Object item = null;

        for (CCelula c = frente.prox; c != null; c = c.prox) {
            if (num == 0) {
                item = frente.prox.item;
                return item;
            }
            if (index == num) {
                if (frente != tras) {
                    item = c.prox.item;
                    qtde--;
                }
                return item;
            }
            index++;
        }
        return item;
    }
}
```

```

    public void mostra() {
        System.out.print("[ ");
        for (CCelula c = frente.prox; c != null; c = c.prox)
            System.out.print(c.item + " ");
        System.out.println("] ");
    }
}

public static void main(String[] args) {
    RandomQueue RQ = new RandomQueue();

    for (int i = 0; i <= 20; i++) {
        RQ.enqueue(i);
    }

    System.out.println("Fila Completa: ");
    RQ.mostra();
    System.out.println("\nElemento aleatório removido: " + RQ.Dequeue());
    RQ.mostra();

    System.out.println("\nElemento aleatório removido: " + RQ.Dequeue());
    RQ.mostra();

    System.out.println("\nElementos aleatórios através do Sample():");
    System.out.print(RQ.Sample() + ", ");
    System.out.print(RQ.Sample() + ", ");
    System.out.print(RQ.Sample() + ", ");
    System.out.print(RQ.Sample() + ", ");
    System.out.print(RQ.Sample() + ", ");
    System.out.print(RQ.Sample() + ", ");
    System.out.print(RQ.Sample() + ", ");
    System.out.print(RQ.Sample() + ", ");
    System.out.print(RQ.Sample() + ", ");
    System.out.print(RQ.Sample() + ". ");
}

```

Saída do Programa

```

<terminated> Exercicio7 [Java Application] C:\Program Files\Java\jdk-11.0.2\bin\jav
Fila Completa:
[ 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 ]

Elemento aleatório removido: 0
[ 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 ]

Elemento aleatório removido: 1
[ 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 ]

Elementos aleatórios através do Sample():
3, 15, 16, 5, 14, 10, 10, 5, 8, 6.

```

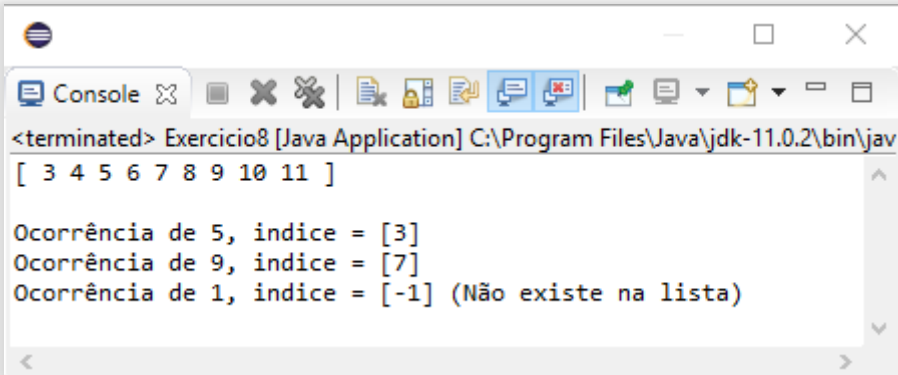

8 – Crie na CListaDup o método `int primeiraOcorrenciaDe(Object elemento)` que busca e retorna o índice da primeira ocorrência do elemento passado por parâmetro. Caso o elemento não exista, sua função deve retornar um valor negativo. Obs: considere que o primeiro elemento está na posição 1.

```
public int primeiraOcorrenciaDe(Object elemento) {
    if (primeira != ultima) {
        CCelulaDup aux = primeira.prox;
        boolean achou = false;
        int index = 0;
        while (aux != null && !achou) {
            achou = aux.item.equals(elemento);
            index++;
            if (!achou)
                aux = aux.prox;
        }
        if (!achou && index >= qtde) {
            return -1;
        }
        return index;
    }
    return -1;
}

public static void main(String[] args) {

    CListaDup lista = new CListaDup();
    for (int i = 3; i < 12; i++) {
        lista.insereFim(i);
    }
    lista.imprime();
    System.out.println("\n\nOcorrência de 5, indice = [lista.primeiraOcorrenciaDe(5) +
    ]");
    System.out.println("Ocorrência de 9, indice = [" +
        lista.primeiraOcorrenciaDe(9) + "]);
    System.out.println("Ocorrência de 1, indice = [" +
        lista.primeiraOcorrenciaDe(1) + "] (Não existe na lista)");
}
```

Saída do Programa



```
<terminated> Exercicio8 [Java Application] C:\Program Files\Java\jdk-11.0.2\bin\jav
[ 3 4 5 6 7 8 9 10 11 ]

Ocorrência de 5, indice = [3]
Ocorrência de 9, indice = [7]
Ocorrência de 1, indice = [-1] (Não existe na lista)
```

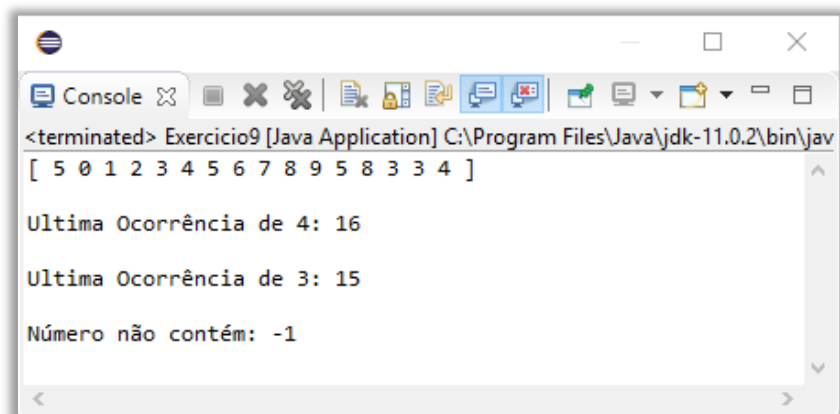
9 –Crie na CListaDup o método `int ultimaOcorrenciaDe(Object elemento)` que busca e retorna o índice da ultima ocorrência do elemento passado por parâmetro. Caso o elemento não exista, sua função deve retornar um valor negativo. Obs: considere que o primeiro elemento está na posição 1.

```
public int ultimaOcorrenciaDe(Object elemento) {
    int ultimoIndex = -1;
    if (primeira != ultima) {
        CCelulaDup aux = primeira.prox;
        boolean achou = false;
        int index = 0;
        if (!contem(elemento)) {
            return -1;
        }
        while (aux != null && index <= qtde) {
            if (aux.item.equals(elemento)) {
                ultimoIndex = index;
            }
            achou = aux.item.equals(elemento);
            index++;
            aux = aux.prox;
            if (!achou && index >= qtde) {
                return ultimoIndex + 1;
            }
        }
        if (achou) {
            return index;
        }
    }
    return -1;
}

public static void main(String[] args) {

    CListaDup lista = new CListaDup();
    for (int i = 0; i < 10; i++) {
        lista.insereFim(i);
    }
    lista.insereComeco(5);
    lista.insereFim(5);
    lista.insereFim(8);
    lista.insereFim(3);
    lista.insereFim(3);
    lista.insereFim(4);
    lista.imprime();
    System.out.println("\n\nUltima Ocorrência de 4: " + lista.ultimaOcorrenciaDe(4));
    System.out.println("\nUltima Ocorrência de 3: " + lista.ultimaOcorrenciaDe(3));
    System.out.println("\nNúmero não contém: " + lista.ultimaOcorrenciaDe(22));
}
```

Saída do Programa



```
<terminated> Exercicio9 [Java Application] C:\Program Files\Java\jdk-11.0.2\bin\jav
[ 5 0 1 2 3 4 5 6 7 8 9 5 8 3 3 4 ]

Ultima Ocorrência de 4: 16

Ultima Ocorrência de 3: 15

Número não contém: -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:

```
public class Deque {
    private C CelulaDup primeira; // Referencia a primeira celula da lista (celula cabeca)
    private C CelulaDup ultima; // Referencia a ultima celula da lista
    private int qtde;

    public Deque() {
        primeira = new C CelulaDup();
        ultima = primeira;
    }

    // Insere um novo elemento no fim da lista.
    public void pushRight(Object valorItem) {
        ultima.prox = new C CelulaDup(valorItem, ultima, null);
        ultima = ultima.prox;
        qtde++;
    }

    // Insere um novo elemento no comeco da lista.
    public void pushLeft(Object valorItem) {
        if (primeira == ultima) { // Se a lista estiver vazia insere no fim
            ultima.prox = new C CelulaDup(valorItem, ultima, null);
            ultima = ultima.prox;
        }
        else { // senao insere no comeco
            primeira.prox = new C CelulaDup(valorItem, primeira, primeira.prox);
            primeira.prox.prox.ant = primeira.prox;
        }
        qtde++;
    }

    public boolean isEmpty() {
        return primeira == ultima;
    }

    public int quantidade() {
        return qtde;
    }

    // Remove e retorna o primeiro elemento da lista.
    public Object popLeft() {
        if (primeira != ultima) {
            C CelulaDup aux = primeira.prox;
            primeira = primeira.prox;
            primeira.ant = null;
            qtde--;
            return aux.item;
        }
        return null;
    }

    // Remove e retorna o ultimo elemento da lista.
    public Object popRight() {
        if (primeira != ultima) {
            C CelulaDup aux = ultima;
            ultima = ultima.ant;
            ultima.prox = null;
            qtde--;
            return aux.item;
        }
        return null;
    }

    public void imprime() {
        C CelulaDup aux = primeira.prox;
        System.out.print("[ ");
        while (aux != null) {
            System.out.print(aux.item + " ");
        }
    }
}
```

```

        aux = aux.prox;
    }
    System.out.print("] ");
}

public static void main(String[] args) {
    Deque deque = new Deque();

    System.out.println("Inserção dos elementos 4,3,2,1 pela esquerda:");

    deque.pushLeft(1);
    deque.pushLeft(2);
    deque.pushLeft(3);
    deque.pushLeft(4);
    deque.imprime();

    System.out.println("\n\nInserção dos elementos 8,7,6 pela direita:");
    deque.pushRight(6);
    deque.pushRight(7);
    deque.pushRight(8);
    deque.imprime();

    System.out.println("\n\nRemoção dos elementos das beiradas: ");

    deque.popRight();
    deque.popLeft();
    deque.imprime();
}

```

Saída do Programa

```

<terminated> Exercicio10 [Java Application] C:\Program Files\Java\jdk-11.0.2\bin\javaw.
Inserção dos elementos 4,3,2,1 pela esquerda:
[ 4 3 2 1 ]

Inserção dos elementos 8,7,6 pela direita:
[ 4 3 2 1 6 7 8 ]

Remoção dos elementos das beiradas (popRight() & popLeft())
[ 3 2 1 6 7 ]

```

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

```
public void removePos(int n) {
    if ((n >= 1) && (n <= qtde) && (primeira != ultima)) {
        int i = 0;
        CCelula aux = primeira;
        while (i < n - 1) {
            aux = aux.prox;
            i++;
        }
        aux.prox = aux.prox.prox;
        if (aux.prox == null)
            ultima = aux;
        qtde--;
    }
}
```

```
public static void main(String[] args) {

    CLista lista = new CLista();
    for (int i = 1; i < 30; i += 3) {
        lista.insereFim(i);
    }

    System.out.println("Lista Completa:");
    lista.imprimeFormatoLista();

    System.out.println("\nLista Remoção Posição[5]:");
    lista.removePos(5);
    lista.imprimeFormatoLista();

    System.out.println("\nLista Remoção Posição[3]:");
    lista.removePos(3);
    lista.imprimeFormatoLista();

    System.out.println("\nLista Remoção Posição[8]:");
    lista.removePos(8);
    lista.imprimeFormatoLista();

}
```

Saída do Programa

```
<terminated> Exercicio11 [Java Application] C:\Program Files\Java\jdk-11.0.2\bin\javaw.exe (3 de abr de 201
Lista Completa:
[/]->[1]->[4]->[7]->[10]->[13]->[16]->[19]->[22]->[25]->[28]->null

Lista Remoção Posição[5]:
[/]->[1]->[4]->[7]->[10]->[16]->[19]->[22]->[25]->[28]->null

Lista Remoção Posição[3]:
[/]->[1]->[4]->[10]->[16]->[19]->[22]->[25]->[28]->null

Lista Remoção Posição[8]:
[/]->[1]->[4]->[10]->[16]->[19]->[22]->[25]->null
```

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

```
public void removePos(int n) {
    if ((n >= 1) && (n <= qtde) && (primeira != ultima)) {
        int i = 0;
        C CelulaDup aux = primeira;
        while (i < n - 1) {
            aux = aux.prox;
            i++;
        }
        aux.prox = aux.prox.prox;
        if (aux.prox == null)
            ultima = aux;
        qtde--;
    }
}

public static void main(String[] args) {

    CListaDup lista = new CListaDup();
    for (int i = 1; i < 30; i += 3) {
        lista.insereFim(i);
    }

    lista.imprimeFormatoLista();
    System.out.println("\nRemove através de um laço todas as posições[i]: \n");

    for (int i = 10; i > 0; i--) {
        lista.removePos(i);
        lista.imprimeFormatoLista();
    }
}
```

Saída do Programa

```
<terminated> Exercicio12 [Java Application] C:\Program Files\Java\jdk-11.0.2\bin\javaw.exe (3 de abr de 201
[/]->[1]->[4]->[7]->[10]->[13]->[16]->[19]->[22]->[25]->[28]->null

Remove através de um laço todas as posições[i]:

[/]->[1]->[4]->[7]->[10]->[13]->[16]->[19]->[22]->[25]->null
[/]->[1]->[4]->[7]->[10]->[13]->[16]->[19]->[22]->null
[/]->[1]->[4]->[7]->[10]->[13]->[16]->[19]->null
[/]->[1]->[4]->[7]->[10]->[13]->[16]->null
[/]->[1]->[4]->[7]->[10]->[13]->null
[/]->[1]->[4]->[7]->[10]->null
[/]->[1]->[4]->[7]->null
[/]->[1]->[4]->null
[/]->[1]->null
[/]->null
```

13 – Crie na CFile o método `int qtdeOcorrencias(Object elemento)` a qual retorna a quantidade de vezes que o elemento passado como parâmetro está armazenado na CFile.

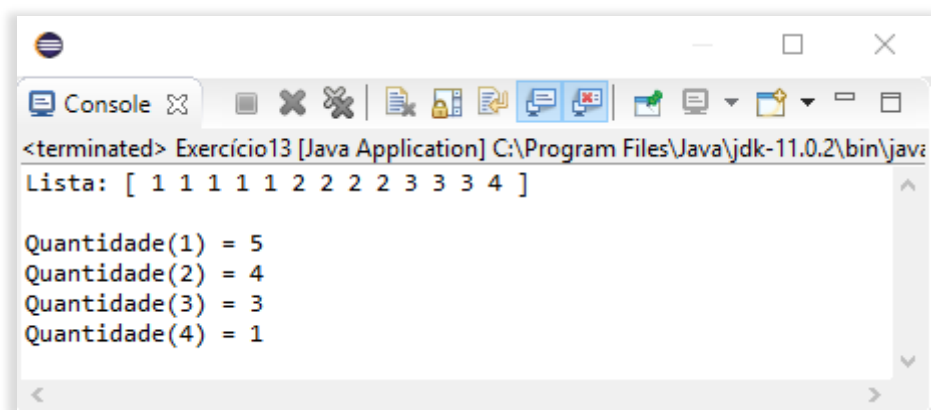
```
public int qtdeOcorrencia(Object elemento) {
    int contador = 0;
    int qtdeOcorrencia = 0;
    CCellula aux = frente;
    while (aux.prox != null && contador <= qtde) {
        aux = aux.prox;
        if (aux.item.equals(elemento)) {
            qtdeOcorrencia++;
        }
        contador++;
    }
    return qtdeOcorrencia;
}

public static void main(String[] args) {
    CFile fila = new CFile();
    fila.enqueue(1);
    fila.enqueue(1);
    fila.enqueue(1);
    fila.enqueue(1);
    fila.enqueue(1);
    fila.enqueue(2);
    fila.enqueue(2);
    fila.enqueue(2);
    fila.enqueue(2);
    fila.enqueue(3);
    fila.enqueue(3);
    fila.enqueue(3);
    fila.enqueue(4);

    System.out.print("Lista: ");

    fila.mostra();
    System.out.println("\nQuantidade(1) = " + fila.qtdeOcorrencia(1));
    System.out.println("Quantidade(2) = " + fila.qtdeOcorrencia(2));
    System.out.println("Quantidade(3) = " + fila.qtdeOcorrencia(3));
    System.out.println("Quantidade(4) = " + fila.qtdeOcorrencia(4));
}
```

Saída do Programa



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

```
public void inverte() {
    CLista lista = new CLista();

    int tamPilha = this.quantidade();

    for (int i = 0; i < tamPilha; i++) {
        lista.insereFim(this.desempilha());
    }

    for (int i = 0; i < tamPilha; i++) {
        this.empilha(lista.removeRetornaComeco());
    }
}

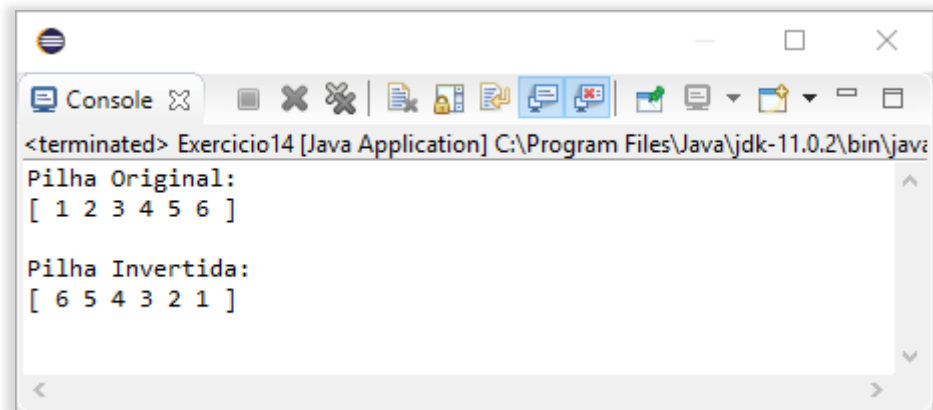
public static void main(String[] args) {

    CPilha pilha = new CPilha();
    pilha.empilha(6);
    pilha.empilha(5);
    pilha.empilha(4);
    pilha.empilha(3);
    pilha.empilha(2);
    pilha.empilha(1);

    System.out.println("Pilha Original:");
    pilha.mostra();

    pilha.inverte();
    System.out.println("\nPilha Invertida:");
    pilha.mostra();
}
```

Saída do Programa



15 – Crie na CFile o método **void inverte()** que inverte a ordem dos elementos da Fila.

```
public void inverte() {
    CLista lista = new CLista();

    int tamPilha = this.quantidade();

    for (int i = 0; i < tamPilha; i++) {
        lista.insereComeco(this.desenfileira());
    }

    for (int i = 0; i < tamPilha; i++) {
        this.enfileira(lista.removeRetornaComeco());
    }
}

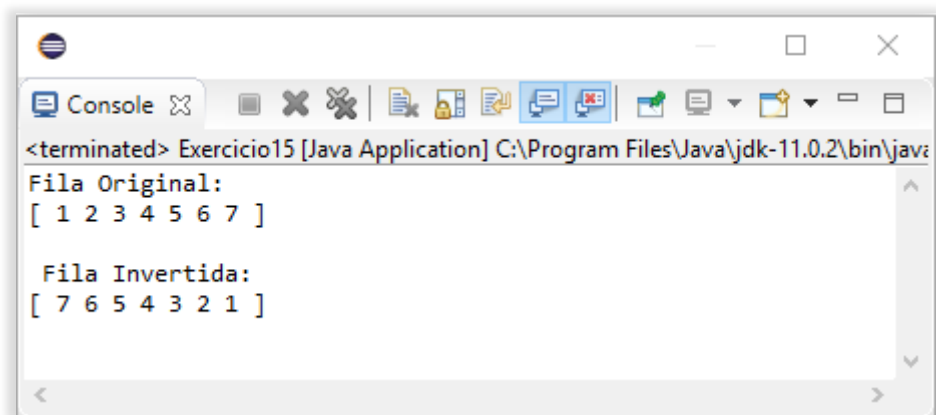
public static void main(String[] args) {

    CFile fila = new CFile();
    fila.enfileira(1);
    fila.enfileira(2);
    fila.enfileira(3);
    fila.enfileira(4);
    fila.enfileira(5);
    fila.enfileira(6);
    fila.enfileira(7);

    System.out.println("Fila Original:");
    fila.mostra();

    fila.inverte();
    System.out.println("\n Fila Invertida:");
    fila.mostra();
}
```

Saída do Programa



30 – Crie as classes CCelulaDicionario e CDicionario

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

// Construtora que anula os três atributos da célula
    public CCelulaDicionario() {
        key = null;
        value = null;
        prox = null;
    }

// Construtora que inicializa key e value com os argumentos passados
// por parâmetro e anula a referência à próxima célula
    public CCelulaDicionario(Object chave, Object valor) {
        key = chave;
        value = valor;
        prox = null;
    }

// Construtora que inicializa todos os atributos da célula com os argumentos
// passados por parâmetro
    public CCelulaDicionario(Object chave, Object valor, CCelulaDicionario proxima) {
        key = chave;
        value = valor;
        prox = proxima;
    }
}

public class CDicionario {
    private CCelulaDicionario primeira, ultima;
    private int qtde;

    public CDicionario(){
        primeira = new CCelulaDicionario();
        ultima = primeira;
    }

    public boolean vazio() {
        return primeira == ultima;
    }

    public void adiciona(Object chave, Object valor) {
        int contador = 0;
        CCelulaDicionario aux = primeira;
        // Verifica se não há nenhuma outra CCelulaDicionario com o mesma chave
        while (aux.prox != null && contador <= qtde) {
            aux = aux.prox;
            if (aux.key.equals(chave)) {
                return;
            }
            contador++;
        }
        ultima.prox = new CCelulaDicionario(chave, valor);
        ultima = ultima.prox;
        qtde++;
    }

    public Object recebeValor(Object chave) {
        int contador = 0;
        CCelulaDicionario aux = primeira;
        while (aux.prox != null && contador <= qtde) {
            aux = aux.prox;
            if (aux.key.equals(chave)) {
                return aux.value;
            }
            contador++;
        }
        return null;
    }
}
```

```

    }

    public void imprimeFormatoLista() {
        System.out.print("[/->");
        for (CCelulaDicionario aux = primeira.prox; aux != null; aux = aux.prox)
            System.out.print("[ " + aux.key + ", " + aux.value + "]->");
        System.out.println("null");
    }
}

```

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 static void main(String[] args) {

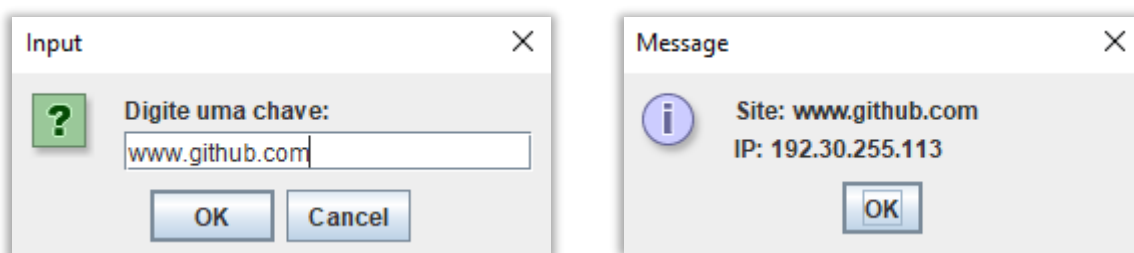
    CDicionario dicionario;
    dicionario = new CDicionario();

    dicionario.adiciona("www.google.com", "172.217.5.110");
    dicionario.adiciona("www.pucminas.br", "200.229.32.27");
    dicionario.adiciona("www.gmail.com", "216.58.192.5");
    dicionario.adiciona("www.youtube.com", "172.217.0.46");
    dicionario.adiciona("www.capes.gov.br", "200.130.18.222");
    dicionario.adiciona("www.yahoo.com", "98.138.219.231");
    dicionario.adiciona("www.microsoft.com", "40.76.4.15");
    dicionario.adiciona("www.twitter.com", "104.244.42.1");
    dicionario.adiciona("www.brasil.gov.br", "170.246.252.243");
    dicionario.adiciona("www.wikipedia.com", "198.35.26.96");
    dicionario.adiciona("www.amazon.com", "172.231.14.116");
    dicionario.adiciona("research.microsoft.com", "13.67.218.189");
    dicionario.adiciona("www.facebook.com", "157.240.22.35");
    dicionario.adiciona("www.whitehouse.gov", "172.231.33.117");
    dicionario.adiciona("www.answers.com", "151.101.188.203");
    dicionario.adiciona("www.uol.com.br", "54.239.132.31");
    dicionario.adiciona("www.hotmail.com", "204.79.197.212");
    dicionario.adiciona("www.cplusplus.com", "167.114.170.15");
    dicionario.adiciona("www.nyt.com", "151.101.189.164");
    dicionario.adiciona("www.apple.com", "172.230.107.90");
    dicionario.adiciona("www.terra.com", "208.70.188.57");
    dicionario.adiciona("www.globo.com", "186.192.81.5");
    dicionario.adiciona("www.stackoverflow.com", "151.101.193.69");
    dicionario.adiciona("www.devmedia.com.br", "187.85.74.81");
    dicionario.adiciona("www.github.com", "192.30.255.113");
    dicionario.adiciona("www.onedrive.live.com", "2.17.196.247");

    String key = JOptionPane.showInputDialog("Digite uma chave:");
    JOptionPane.showMessageDialog(null, "Site: " + key + "\n" +
                                         "IP: " +
dicionario.recebeValor(key));
}

```

Saída do Programa



31 –Um biólogo precisa de um programa que traduza uma trinca de nucleotídeos em seu aminoácidocorrespondente. Por exemplo, a trinca de aminoácidos ACG é traduzida como o aminoácido Treonina, eGCA em Alanina. Crie um programa emJavaque usea sua classeCDicionariopara criar um dicionário docódigo genético. O usuário deve digitar uma trinca (chave) e seu programa deve mostrar o nome (valor)do aminoácido correspondente. Use a tabelaa seguirpara cadastrar todas as trincas/aminoácidos.

```
public static void main(String[] args) {

    CDicionario dicionario;
    dicionario = new CDicionario();

    dicionario.adiciona("UUU", "Fenilalanina");
    dicionario.adiciona("UUC", "Fenilalanina");
    dicionario.adiciona("UUA", "Leucina");
    dicionario.adiciona("UUG", "Leucina");
    dicionario.adiciona("CUU", "Leucina");
    dicionario.adiciona("CUC", "Leucina");
    dicionario.adiciona("CUA", "Leucina");
    dicionario.adiciona("CUG", "Leucina");
    dicionario.adiciona("AUU", "Isoleucina");
    dicionario.adiciona("AUC", "Isoleucina");
    dicionario.adiciona("AUA", "Isoleucina");
    dicionario.adiciona("AUG", "Metionina");
    dicionario.adiciona("GUU", "Valina");
    dicionario.adiciona("GUC", "Valina");
    dicionario.adiciona("GUA", "Valina");
    dicionario.adiciona("GUG", "Valina");
    dicionario.adiciona("UCU", "Serina");
    dicionario.adiciona("UCC", "Serina");
    dicionario.adiciona("UCA", "Serina");
    dicionario.adiciona("UCG", "Serina");
    dicionario.adiciona("CCU", "Prolina");
    dicionario.adiciona("CCC", "Prolina");
    dicionario.adiciona("CCA", "Prolina");
    dicionario.adiciona("CCG", "Prolina");
    dicionario.adiciona("ACU", "Treonina");
    dicionario.adiciona("ACC", "Treonina");
    dicionario.adiciona("ACA", "Treonina");
    dicionario.adiciona("ACG", "Treonina");
    dicionario.adiciona("GCU", "Alanina");
    dicionario.adiciona("GCC", "Alanina");
    dicionario.adiciona("GCA", "Alanina");
    dicionario.adiciona("GCG", "Alanina");
    dicionario.adiciona("UAU", "Tirosina");
    dicionario.adiciona("UAC", "Tirosina");
    dicionario.adiciona("UAA", "Parada");
    dicionario.adiciona("UAG", "Parada");
    dicionario.adiciona("CAU", "Histidina");
    dicionario.adiciona("CAC", "Histidina");
    dicionario.adiciona("CAA", "Glutamina");
    dicionario.adiciona("CAG", "Glutamina");
    dicionario.adiciona("AAU", "Asparagina");
    dicionario.adiciona("AAC", "Asparagina");
    dicionario.adiciona("AAA", "Lisina");
    dicionario.adiciona("AAG", "Lisina");
    dicionario.adiciona("GAU", "Aspartato");
    dicionario.adiciona("GAC", "Aspartato");
    dicionario.adiciona("GAA", "Glutamato");
    dicionario.adiciona("GAG", "Glutamato");
    dicionario.adiciona("UGU", "Cisteina");
    dicionario.adiciona("UGC", "Cisteina");
    dicionario.adiciona("UGA", "Parada");
    dicionario.adiciona("UGG", "Triptofano");
    dicionario.adiciona("CGU", "Arginina");
    dicionario.adiciona("CGC", "Arginina");
    dicionario.adiciona("CGA", "Arginina");
    dicionario.adiciona("CGG", "Arginina");
    dicionario.adiciona("AGU", "Serina");
    dicionario.adiciona("AGC", "Serina");
    dicionario.adiciona("AGA", "Arginina");
```

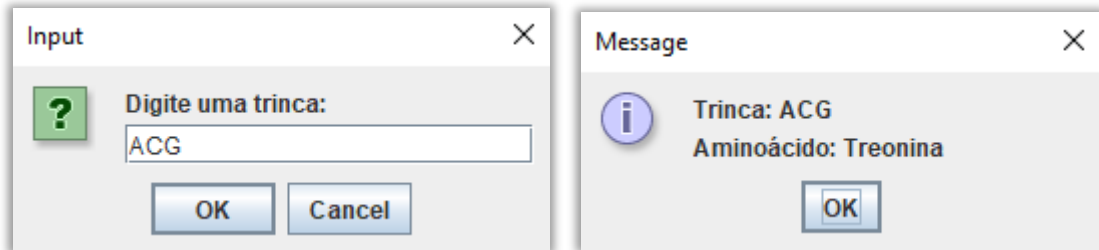
```

dicionario.adiciona("AGG", "Arginina");
dicionario.adiciona("GGU", "Glicina");
dicionario.adiciona("CGC", "Glicina");
dicionario.adiciona("GGA", "Glicina");
dicionario.adiciona("GGG", "Glicina");

String key = JOptionPane.showInputDialog("Digite uma trinca:");
JOptionPane.showMessageDialog(null, "Trinca: " + key + "\n" + "Aminoácido: " +
dicionario.recebeValor(key));
}

```

Saída do Programa



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 CListaSimples {
    public NewCCelula primeira;
    public NewCCelula ultima;

    public CListaSimples() {
        primeira = new NewCCelula();
        ultima = primeira;
    }

    public boolean vazia() {
        return primeira.item == null;
    }

    public void insereComeco(Object valorItem) {
        Object aux = new CCelula();
        if (primeira.item == null) {
            primeira = new NewCCelula();
            primeira.item = valorItem;
            ultima = primeira;
        } else {
            aux = primeira;
            primeira = new NewCCelula();
            primeira.item = valorItem;
            primeira.prox = (NewCCelula) aux;
        }

        if (primeira.prox == null)
            ultima = primeira;
    }

    public Object removeComeco() {
        if (primeira != ultima) {
            NewCCelula aux = primeira.prox;
            primeira = aux;
            if (primeira == null)
                ultima = primeira;
            return aux.item;
        }
        return null;
    }

    public void insereFim(Object valorItem) {
        ultima.prox = new NewCCelula();
        ultima.prox.item = valorItem;
    }
}

```

```

        ultima = ultima.prox;
    }

    public Object removeFim() {
        if (primeira != ultima) {
            NewCCelula aux = primeira;
            while (aux.prox != ultima)
                aux = aux.prox;
            NewCCelula aux2 = aux.prox;
            ultima = aux;
            ultima.prox = null;
            return aux2.item;
        }
        return null;
    }

    public void imprime() {
        for (NewCCelula aux = primeira; aux != null; aux = aux.prox)
            System.out.print "[" + aux.item + "]->";
        System.out.println("null");
    }

    public boolean contem(Object elemento) {
        boolean achou = false;
        NewCCelula aux = primeira;
        while (aux != null && !achou) {
            achou = aux.item.equals(elemento);
            aux = aux.prox;
        }
        return achou;
    }
}

public class NewCCelula {

    public Object item;
    public NewCCelula prox;
}

public static void main(String[] args) {
    CListaSimples listaSimples = new CListaSimples();
    System.out.print("Lista simples vazia?");
    System.out.println(" " + listaSimples.vazia());

    System.out.println("Insere no começo elementos (1),(3),(5):");
    listaSimples.insereComeco(5);
    listaSimples.insereComeco(3);
    listaSimples.insereComeco(1);
    listaSimples.imprime();
    System.out.println("\nContém(3) na lista? " + listaSimples.contem(3));
    System.out.println("Contém(4) na lista? " + listaSimples.contem(4));

    System.out.println("\nInsere no começo da lista elemento (4):");
    listaSimples.insereComeco(4);
    listaSimples.imprime();
    System.out.println("Contém(4) na lista? " + listaSimples.contem(4));

    System.out.println("\nRemove Começo:");
    listaSimples.removeComeco();
    listaSimples.imprime();

    System.out.println("\nInsereFim (8),(9):");
    listaSimples.insereFim(8);
    listaSimples.insereFim(9);
}

```

```

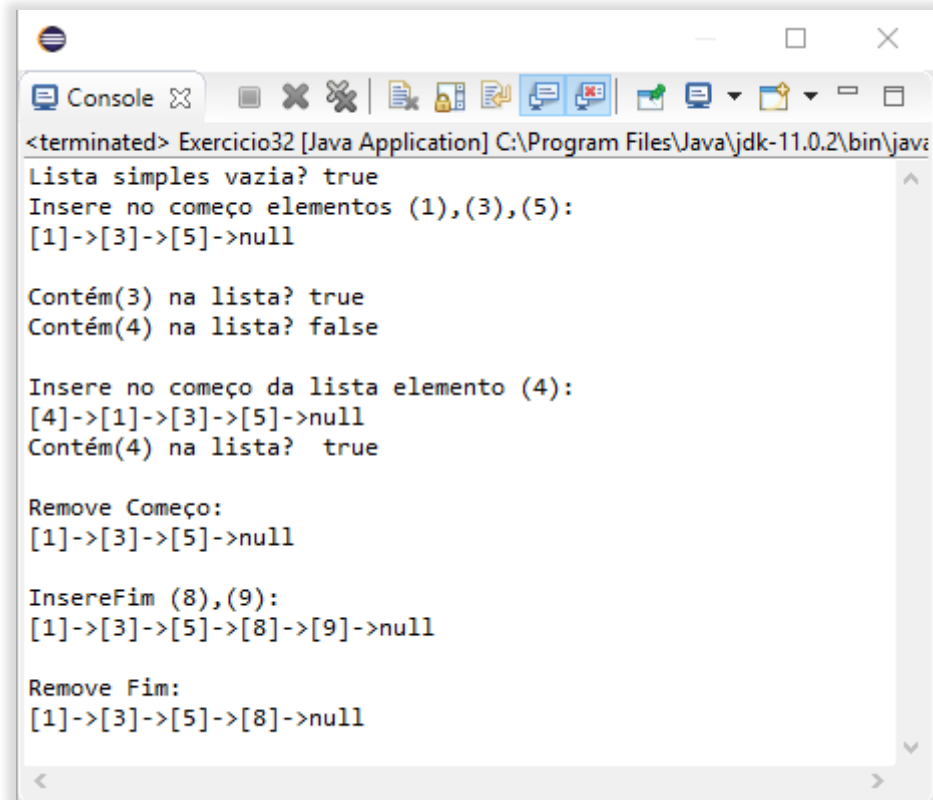
listaSimples.imprime();

System.out.println("\nRemove Fim:");
listaSimples.removeFim();
listaSimples.imprime();

}

```

Saída do Programa



The screenshot shows a Java console window titled "Exercicio32 [Java Application] C:\Program Files\Java\jdk-11.0.2\bin\java". The console output is as follows:

```

<terminated> Exercicio32 [Java Application] C:\Program Files\Java\jdk-11.0.2\bin\java
Lista simples vazia? true
Insere no começo elementos (1),(3),(5):
[1]->[3]->[5]->null

Contém(3) na lista? true
Contém(4) na lista? false

Insere no começo da lista elemento (4):
[4]->[1]->[3]->[5]->null
Contém(4) na lista? true

Remove Começo:
[1]->[3]->[5]->null

InsereFim (8),(9):
[1]->[3]->[5]->[8]->[9]->null

Remove Fim:
[1]->[3]->[5]->[8]->null

```