

Lista 2 - Estrutura de Dados 1 - 2020.1

Prof. Ana Luiza Bessa de Paula Barros

Nome: Natália Sales Aragão - 1461765

Repositório no Github: <https://github.com/natysls/ExerceciosED1/tree/master/Lista%202>

(é só botar na pasta Lista 2)

1.1 Questões Sobre Tipos Abstratos de Dados (TAD)

1. Crie um Tipo Abstrato de Dados para representar um número complexo $z = x + i * y$, em que $i = -1$, sendo x a sua parte real e y a parte imaginária. Implemente funções para:

(a) Criar um número complexo

(b) Destruir um número complexo

(c) Realizar a soma de dois números complexos

(d) Realizar a multiplicação de números complexos

- NumeroComplexo.java

Fiz essa classe abstrata para que suas funções sejam implementadas na classe Funcoes.java. Declarei os protótipos das funções visíveis para o usuário, os tipos das variáveis os dados globalmente acessíveis.

```

NumeroComplexo.java MainTAD1.java Funcoes.java
1 package TAD.questao1;
2
3 public abstract class NumeroComplexo{
4     private double x; //real
5     private double y; //imaginario
6     private double i;
7     private Double z1 = null; //numero complexo
8     private Double z2 = null;
9
10    public double getX() {
11        return x;
12    }
13    public void setX(double x) {
14        this.x = x;
15    }
16    public double getY() {
17        return y;
18    }
19    public void setY(double y) {
20        this.y = y;
21    }
22    public double getI() {
23        return i;
24    }
25    public void setI(double i) {
26        this.i = i;
27    }
28    public Double getZ1() {
29        return z1;
30    }
31    public void setZ1(Double z1) {
32        this.z1 = z1;
33    }
34    public Double getZ2() {
35        return z2;
36    }
37    public void setZ2(Double z2) {
38        this.z2 = z2;
39    }
40
41    public abstract Double criarNumeroComplexo(double x, double y, double i);
42
43    public abstract Double destruirNumeroComplexo(Double z);
44
45    public abstract Double somarDoisNumerosComplexos();
46
47    public abstract Double multiplicarNumerosComplexos();
48
49    @Override
50    public String toString() {
51        return "NumeroComplexo [z=" + z1 + " ]";
52    }
53 }
54
55

```

- Funções.java

Essa é a classe que implementa as funções declaradas no NumeroComplexo.java que estendi.
(obs: eu realmente esqueci de implementar $\sqrt{i} = -1$ nesse código prof 😞)

```
NumeroComplexo.java  Funcoes.java  MainTAD1.java
1 package TAD.questao1;
2
3 public class Funcoes extends NumeroComplexo {
4
5     @Override
6     public Double criarNumeroComplexo(double x, double y, double i) {
7         Double z = x + i * y;
8         if(getZ1() == null) {
9             setZ1(z);
10            System.out.println("Número z1 = " + getZ1() + " criado.\n");
11            return getZ1();
12        }
13
14        setZ2(z);
15        System.out.println("Número z2 = " + getZ2() + " criado.\n");
16        return getZ2();
17    }
18
19
20     @Override
21     public Double destruirNumeroComplexo(Double z){
22         if(z == getZ1()) {
23             setZ1(null);
24             System.out.println("O numero complexo z1 foi destruido: " + getZ1() + "\n");
25             return getZ1();
26         } else if(z == getZ2()) {
27             setZ2(null);
28             System.out.println("O numero complexo z2 foi destruido: " + getZ2() + "\n");
29             return getZ2();
30         } else {
```

```

30     } else {
31         System.out.println("Número não existe, por favor crie.\n");
32         return null;
33     }
34 }
35
36 @Override
37 public Double somarDoisNumerosComplexos() {
38     if(getZ1() == null) {
39         System.out.println("A soma deu " + getZ2() + "\n");
40         return getZ2();
41     } else if(getZ2() == null) {
42         System.out.println("A soma deu " + getZ1() + "\n");
43         return getZ1();
44     }
45     //se não
46     Double soma = getZ1() + getZ2();
47
48     System.out.println("A soma deu " + soma + "\n");
49     return soma;
50 }
51
52
53 @Override
54 public Double multiplicarNumerosComplexos() {
55     if(getZ1() == null) {
56         System.out.println("A multiplicação deu " + getZ2() + "\n");
57         return getZ2();
58     } else if(getZ2() == null) {
59         System.out.println("A multiplicação deu " + getZ1() + "\n");
60         return getZ1();
61     }
62     //se não
63     Double multi = getZ1() * getZ2();
64
65     System.out.println("A multiplicação deu " + multi + "\n");
66
67     return multi;
68 }
69
70
71 }
72

```

- MainTAD1.java

Bora botar pra funcionar

```
NumeroComplexo.java  Funcoes.java  MainTAD1.java  ✕
1 package TAD.questao1;
2
3 public class MainTAD1 {
4
5     public static void main(String[] args) throws Exception {
6         Funcoes funcao = new Funcoes();
7
8         Double z1 = funcao.criarNumeroComplexo(1, 2, 3);
9
10        Double z2 = funcao.criarNumeroComplexo(4, 5, 6);
11
12        funcao.somarDoisNumerosComplexos();
13
14        funcao.multiplicarNumerosComplexos();
15
16        funcao.destruirNumeroComplexo(z1);
17
18        funcao.somarDoisNumerosComplexos();
19
20        funcao.multiplicarNumerosComplexos();
21
22        Double z = 4.0; //numero qualquer, vai dar uma exception!!
23
24        System.out.println("Vou destruir um numero " + z + " que não criei: ");
25        funcao.destruirNumeroComplexo(z);
26    }
27
28 }
29
```

- Terminal

```
Console  ✕  Debug
<terminated> MainTAD1 [Java Application] C:\Program Files\Java\jre1.8.0_111\
Número z1 = 7.0 criado.

Número z2 = 34.0 criado.

A soma deu 41.0

A multiplicação deu 238.0

O numero complexo z1 foi destruido: null

A soma deu 34.0

A multiplicação deu 34.0

Vou destruir um numero 4.0 que não criei:
Número não existe, por favor crie.
```

2. Crie um Tipo Abstrato de Dados para representar uma esfera. Inclua as funções de inicializações necessárias e as operações que retornem o seu raio, a sua área e o seu volume.

- Esfera.java

Declarei os protótipos das funções visíveis para o usuário, os tipos das variáveis os dados globalmente acessíveis.

```
Esfera.java  Funcoes.java  MainTAD2.java
1 package TAD.questao2;
2 public abstract class Esfera {
3     //raio, a sua área e o seu volume.
4     private double raio;
5     private double volume; // (4/3) * π * r³
6     private double area; //4 . π . r²
7
8     public double getVolume() {
9         return volume;
10    }
11    public void setVolume(double volume) {
12        this.volume = volume;
13    }
14    public double getRaio() {
15        return raio;
16    }
17    public void setRaio(double raio) {
18        this.raio = raio;
19    }
20    public double getArea() {
21        return area;
22    }
23    public void setArea(double area) {
24        this.area = area;
25    }
26
27    public abstract void retornaRaio();
28    public abstract void calcularArea();
29    public abstract void calcularVolume();
30 }
```

- Funcoes.java

Classe onde estendi Esfera.java a fim de implementar as funções prototípicas.

```
Esfera.java  Funcoes.java  MainTAD2.java
1 package TAD.questao2;
2
3 public class Funcoes extends Esfera{
4     public Funcoes(double raio) {
5         super();
6         setRaio(raio);
7     }
8
9     @Override
10    public void retornaRaio() {
11        System.out.println(getRaio());
12    }
13
14    @Override
15    public void calcularArea() { //4 . π . r2
16        double area = 4 * 3.14 * getRaio() * getRaio();
17        setArea(area);
18        System.out.println(area);
19        System.out.println(getArea());
20    }
21
22    @Override
23    public void calcularVolume() { // (4/3) * π * r3
24        double volume = (4/3) * 3.14 * (getRaio() * getRaio() * getRaio());
25        setArea(volume);
26        System.out.println(volume);
27        System.out.println(getArea());
28    }
29 }
30 }
```

- MainTAD2.java e Terminal

```
Esfera.java  Funcoes.java  MainTAD2.java
1 package TAD.questao2;
2
3 public class MainTAD2 {
4     public static void main(String[] args) {
5         Funcoes func = new Funcoes(3);
6         func.retornaRaio();
7         func.calcularArea();
8         func.calcularVolume();
9     }
10 }
```

Console Debug

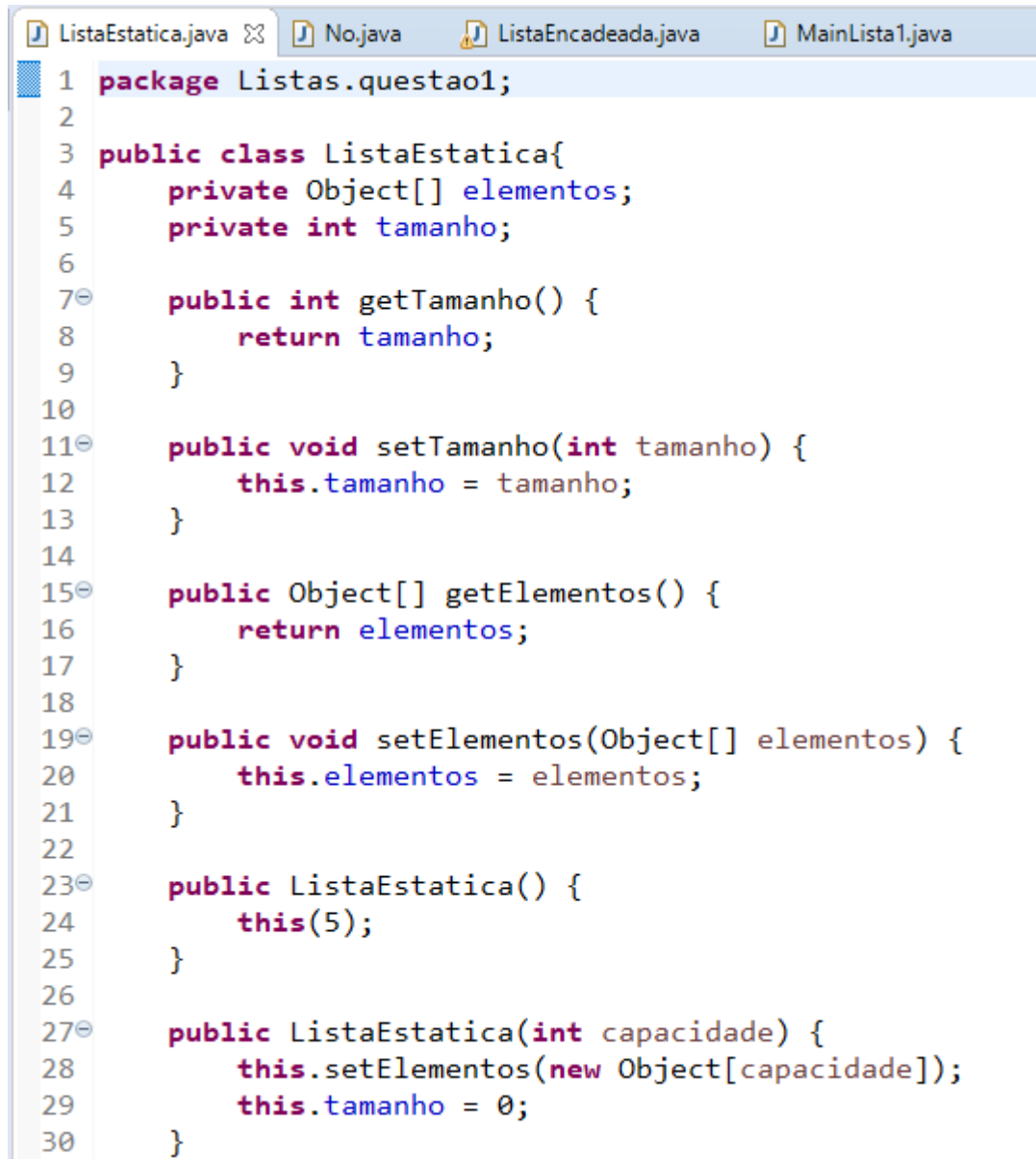
<terminated> MainTAD2 [Java Application] C:\Program Files\Java\jre1.8.0_111\bin\jav
3.0
113.03999999999999
84.78

Questões Sobre Listas

1. Escreva uma função que copie os elementos de uma lista estática para uma lista dinâmica, eliminando os itens repetidos. Implemente a mesmas funções copiando de uma lista dinâmica para uma lista estática.

- ListaEstatica.java

Implementei uma lista tamanho iniciando 0 e elementos com capacidade a atribuir.



```
1 package Listas.questao1;
2
3 public class ListaEstatica{
4     private Object[] elementos;
5     private int tamanho;
6
7     public int getTamanho() {
8         return tamanho;
9     }
10
11     public void setTamanho(int tamanho) {
12         this.tamanho = tamanho;
13     }
14
15     public Object[] getElementos() {
16         return elementos;
17     }
18
19     public void setElementos(Object[] elementos) {
20         this.elementos = elementos;
21     }
22
23     public ListaEstatica() {
24         this(5);
25     }
26
27     public ListaEstatica(int capacidade) {
28         this.setElementos(new Object[capacidade]);
29         this.tamanho = 0;
30     }
```

Daí implementei as funções de “adiciona” que atribui o elemento na lista de elementos na posição i.; e de “remove” que atribui null no elemento na posição i;


```

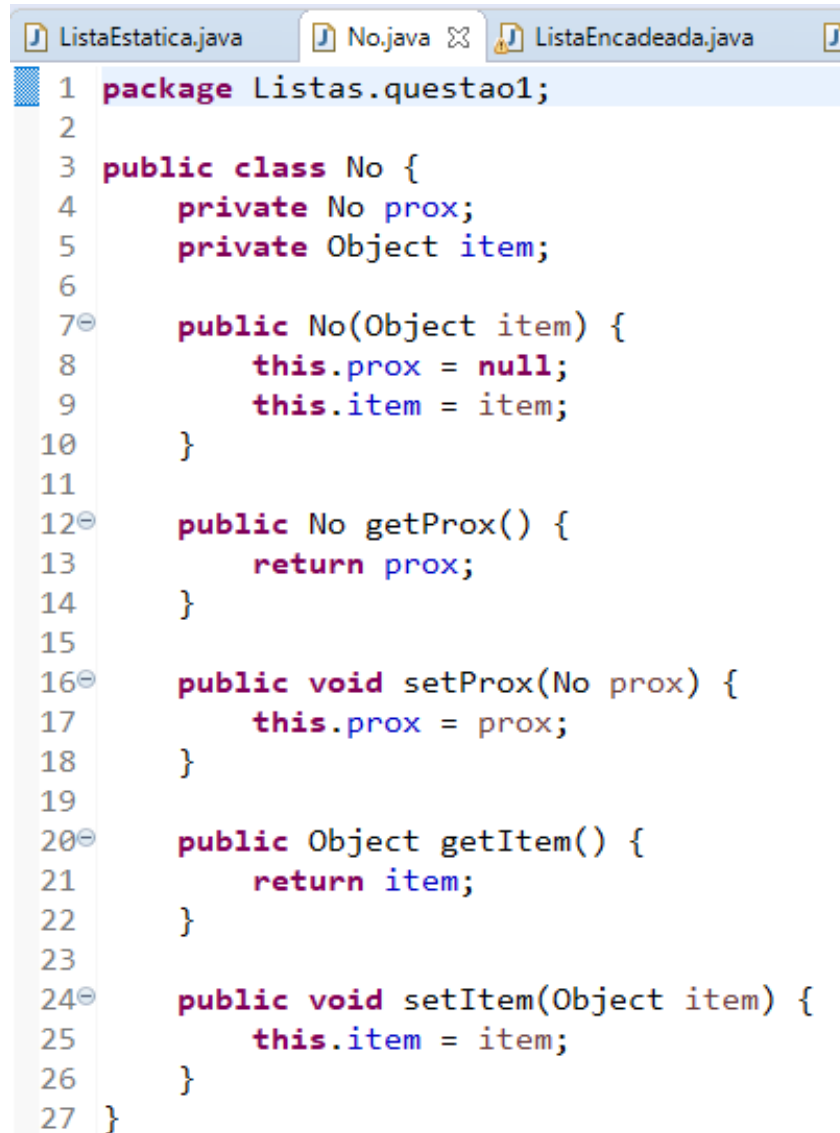
32 public boolean adiciona(Object elemento) { // e criar uma lista
33     this.aumentaCapacidade();
34     if(this.tamanho < this.elementos.length) {
35         this.elementos[this.tamanho] = elemento;
36         this.tamanho++;
37         return true;
38     }
39     return false;
40 }
41
42 public void aumentaCapacidade() {
43     if(this.tamanho == this.elementos.length) {
44         Object[] elementosNovos = new Object[this.elementos.length * 2]; // apenas aumentando a capacidade
45         for(int i=0; i<this.elementos.length; i++) {
46             elementosNovos[i] = this.elementos[i]; // para que ele possa receber o elemento i
47         }
48         this.elementos = elementosNovos;
49     }
50 }
51
52 public void remove(int posicao) {
53     for(int i=posicao; i<this.tamanho - 1; i++){
54         this.elementos[i] = this.elementos[i+1]; // i=1 -> i=2
55     }
56
57     this.elementos[this.tamanho - 1] = null;
58     this.tamanho--; // o tamanho vai passar para 4 para tirar o ultimo F
59 }
60
61 public Object copiaDinamica() {
62     ListaEncadeada encadeada = new ListaEncadeada(this.tamanho);
63     for(int i = 0; i < this.elementos.length; i++) {
64         encadeada.addFim(this.elementos[i]);
65         try {
66             if(encadeada.getElementos()[i] == this.elementos[i + 1]) {
67                 encadeada.remFim(i);
68             }
69         } catch (ArrayIndexOutOfBoundsException e) {
70             e.getMessage();
71         }
72     }
73     return encadeada;
74 }
75
76 @Override
77 public String toString() {
78     StringBuilder s = new StringBuilder();
79     s.append("[");
80     for(int i=0; i<this.tamanho-1;i++) {
81         s.append(this.elementos[i]);
82         s.append(", ");
83     }
84     if(this.tamanho > 0) {
85         s.append(this.elementos[this.tamanho - 1]);
86     }
87     s.append("]");
88     return s.toString();
89 }

```

E então, fiz a cópia da lista estática em dinâmica, nesse caso, eu só adicionei no fim pois ia acrescentando à medida que a posição do vetor elementos incrementa. Em cada elemento eu checo se é igual ao próximo elemento, se sim, ele remove.

- ListaEncadeada.java

Primeiro fiz uma classe “No” para me auxiliar na lista dinâmica.



```
1 package Listas.questao1;
2
3 public class No {
4     private No prox;
5     private Object item;
6
7     public No(Object item) {
8         this.prox = null;
9         this.item = item;
10    }
11
12    public No getProx() {
13        return prox;
14    }
15
16    public void setProx(No prox) {
17        this.prox = prox;
18    }
19
20    public Object getItem() {
21        return item;
22    }
23
24    public void setItem(Object item) {
25        this.item = item;
26    }
27 }
```

Lista Encadeada

```
ListaEstatica.java  No.java  ListaEncadeada.java  MainLista1.java

1  package Listas.questao1;
2
3  public class ListaEncadeada {
4      private Object[] elementos;
5      private No inicio, fim;
6      private int tamanho;
7
8      public ListaEncadeada(int capacidade) {
9          this.setElementos(new Object[capacidade]);
10         this.inicio = null;
11         this.fim = null;
12         this.tamanho = 0;
13     }
14
15     public ListaEncadeada() {
16         this(5);
17     }
18
19     public void addInicio(Object item) { //adiciona no inicio da lista
20         No novo = new No(item);
21
22         if(isEmpty()) { //lista vazia
23             inicio = novo;
24             fim = novo;
25         } else {
26             novo.setProx(inicio);
27             //novo irá apontar para o inicio
28
29             inicio = novo;
30             //inicio -> prox -> prox -> fim
31         }
32         tamanho++; //novo elemento na lista
33     }
34
35     public Object remInicio() {
36         if(isEmpty()) return null;
37
38         No aux = inicio;
39         inicio = aux.getProx(); //head = prox
40
41         aux.setProx(null); //anterior que era o head ta apontando para null
42         //logo o garbage collector vai verificar se ele ta sendo utilizado pela
43         //lista e vai deletá-lo da memória
44         tamanho--;
45         return aux.getItem();
46     }
```

```

48 public void addFim(Object item) {
49     No novo = new No(item);
50
51     if(isEmpty()) { //lista vazia
52         inicio = novo;
53         fim = novo;
54         adiciona(inicio);
55     }else {
56         fim.setProx(novo);
57         fim = novo;
58         //tail = ultimo
59         adiciona(fim);
60     }
61 }
62
63 public boolean adiciona(No elemento) { // e criar uma lista
64     this.aumentaCapacidade();
65     if(this.tamanho < this.elementos.length) {
66         try {
67             this.elementos[this.tamanho] = elemento.getItem();
68         }catch(ArrayStoreException e) {
69             e.getMessage();
70         }
71         this.tamanho++;
72         return true;
73     }
74     return false;
75 }
76
77 public void aumentaCapacidade() {
78     if(this.tamanho == this.elementos.length) {
79         Object[] elementosNovos = new Object[this.elementos.length * 2]; //apenas aumentando a capacidade
80         for(int i=0; i<this.elementos.length; i++) {
81             elementosNovos[i] = this.elementos[i]; //para que ele possa receber o elemento i
82         }
83         this.elementos = elementosNovos;
84     }
85 }
86
87 public Object remFim(int posicao) {
88     if(isEmpty()) return null;
89
90     No aux = inicio;
91     while(aux != null) {
92         if(aux.getProx().getProx() == null) { //aux.getProx() == tail
93             No aux2 = fim;
94             fim = aux;
95             fim.setProx(null); //cortando a referencia com o ultimo nó, limpa a memória
96             remove(posicao);
97             return aux2.getItem();
98         }
99         aux = aux.getProx();
100     }
101     return null;
102 }
103

```

```

104 public void remove(int posicao) {
105     for(int i=posicao; i<this.tamanho - 1; i++){
106         this.elementos[i] = this.elementos[i+1]; // i=1 -> i=2
107     }
108     this.elementos[this.tamanho - 1] = null;
109     this.tamanho--; //o tamanho vai passar para 4 para tirar o ultimo F
110 }
111
112 public Object copiaEstatica() { //em estática
113     ListaEstatica estatica = new ListaEstatica(this.tamanho);
114     for(int i = 0; i < this.tamanho; i++) {
115         estatica.adiciona(this.elementos[i]);
116         try {
117             if(estatica.getElementos()[i] == this.elementos[i + 1]) {
118                 estatica.remove(i);
119             }
120         } catch (ArrayIndexOutOfBoundsException e) {
121             e.getMessage();
122         }
123     }
124     return estatica;
125 }

127 public Object[] getElementos() {
130
131 public void setElementos(Object[] elementos) {
134
135 public No getInicio() {
138
139 public void setInicio(No inicio) {
142
143 public No getFim() {
146
147 public void setFim(No fim) {
150
151 public int getTamanho() {
154
155 public void setTamanho(int tamanho) {
158
159 private boolean isEmpty() {
160     return inicio == null;
161 }

163 @Override
164 public String toString() {
165     StringBuilder s = new StringBuilder();
166     s.append("[");
167     for(int i=0; i<this.tamanho-1;i++) {
168         s.append(this.elementos[i]);
169         s.append(", ");
170     }
171     if(this.tamanho > 0) {
172         s.append(this.elementos[this.tamanho -1]);
173     }
174     s.append("]");
175     return s.toString();
176 }
177 }
178

```

- MainLista1.java e Terminal

```

1 package Listas.questao1;
2 public class MainLista1 {
3     public static void main(String[] args){
4         ListaEstatica estatica = new ListaEstatica(4);
5         estatica.adiciona(1);
6         estatica.adiciona(2);
7         estatica.adiciona(2);
8         estatica.adiciona(4);
9
10        System.out.println("Lista estática = " + estatica);
11        System.out.println("Lista transformada em encadeada = " + estatica.copiaDinamica());
12
13        ListaEncadeada encadeada = new ListaEncadeada(4);
14        encadeada.addFim(5);
15        encadeada.addFim(6);
16        encadeada.addFim(7);
17        encadeada.addFim(7);
18
19        System.out.println("Lista Encadeada = " + encadeada);
20        System.out.println("Lista transformada em estática = " + encadeada.copiaEstatica());
21    }
22 }

```

```

<terminated> MainLista1 [Java Application] C:\Program Files\Java\jre1.8.0_111\bin\javaw.exe (9 de dez de 2020 21:39:17 - 21:39:18)
Lista estática = [1, 2, 2, 4]
Lista transformada em encadeada = [1, 2, 4]
Lista Encadeada = [5, 6, 7, 7]
Lista transformada em estática = [5, 6, 7]

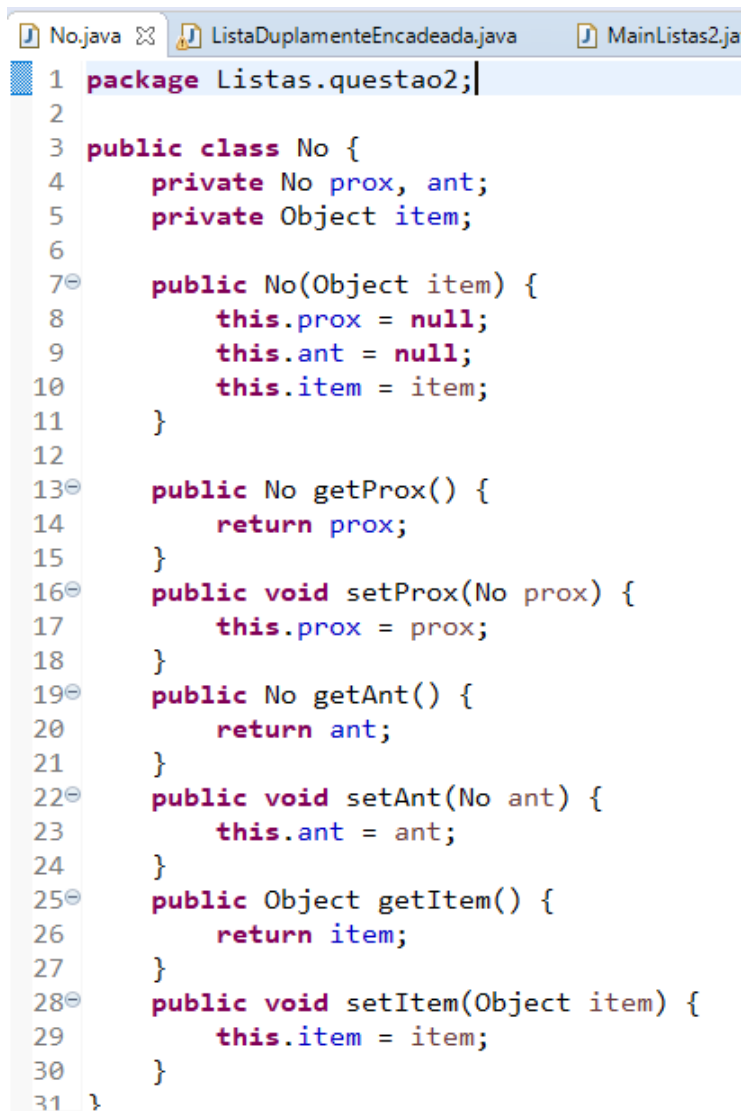
```

2. Um problema que pode surgir na manipulação de listas lineares simples é o de "voltar" atrás na lista, ou seja, percorrê-la no sentido inverso ao dos apontadores. A solução geralmente adotada é a incorporação ao elemento de um apontador para o seu antecessor. Listas desse tipo são chamadas de duplamente encadeadas. Com base nisso, escreva:

- Uma função para criar uma lista duplamente encadeada
- Uma função para inserir um elemento em uma lista duplamente encadeada
- Uma função para remover um elemento de uma lista duplamente encadeada

- No.java

Na lista duplamente encadeada, cada nó possui um ponteiro para o nó antecessor e outro para o nó sucessor.



```

1 package Listas.questao2;
2
3 public class No {
4     private No prox, ant;
5     private Object item;
6
7     public No(Object item) {
8         this.prox = null;
9         this.ant = null;
10        this.item = item;
11    }
12
13    public No getProx() {
14        return prox;
15    }
16    public void setProx(No prox) {
17        this.prox = prox;
18    }
19    public No getAnt() {
20        return ant;
21    }
22    public void setAnt(No ant) {
23        this.ant = ant;
24    }
25    public Object getItem() {
26        return item;
27    }
28    public void setItem(Object item) {
29        this.item = item;
30    }
31 }

```

- ListaDuplamenteEncadeada.java

Declarei dois objetos do tipo “No” para indicar o inicio e o fim

No.java ListaDuplamenteEncadeada.java MainListas2.java

```
3 public class ListaDuplamenteEncadeada{
4     protected Object[] elementos;
5     protected No inicio, fim;
6     protected int tamanho;
7
8     public ListaDuplamenteEncadeada(int capacidade) {
9         this.setElementos(new Object[capacidade]);
10        this.inicio = null;
11        this.fim = null;
12        this.tamanho = 0;
13    }
14
15    public ListaDuplamenteEncadeada() {
16        this(5);
17    }
18
19    public void adicionaInicio(Object item) {
20        No novo = new No(item);
21        if(vazio()) { //lista vazia
22            inicio = novo; // inicio <- novo -> fim
23            fim = novo;
24        }else {
25            inicio.setAnt(novo); //inicio <- novo
26            novo.setProx(inicio); //novo -> inicio
27            inicio = novo; //inicio = novo;
28        }
29        tamanho++; //novo elemento na lista
30    }
31
32    public void adicionaFim(Object item) {
33        No novo = new No(item);
34
35        if(vazio()) { //lista vazia
36            inicio = novo; // inicio <- novo -> fim
37            fim = novo;
38            adiciona(inicio);
39        }else {
40            novo.setAnt(fim);
41            fim.setProx(novo);
42            fim = novo;
43            adiciona(fim);
44        }
45    }
```



```

47 public boolean adiciona(No elemento) { // e criar uma lista
48     this.aumentaCapacidade();
49     if(this.tamanho < this.elementos.length) {
50         try {
51             this.elementos[this.tamanho] = elemento.getItem();
52         } catch (ArrayStoreException e) {
53             e.getMessage();
54         }
55         this.tamanho++;
56         return true;
57     }
58     return false;
59 }
60
61
62 public void aumentaCapacidade() {
63     if(this.tamanho == this.elementos.length) {
64         Object[] elementosNovos = new Object[this.elementos.length * 2];
65         for(int i=0; i<this.elementos.length; i++) {
66             elementosNovos[i] = this.elementos[i];
67         }
68         this.elementos = elementosNovos;
69     }
70 }

74 public Object removeInicio(int posicao) {
75     if(vazio()) return null; //se não tiver nada, retorna nulo
76
77     if(inicio.getProx() != null) {
78         inicio = inicio.getProx(); // inicio = prox
79         inicio.getAnt().setProx(null); // ant <- null
80         inicio.setAnt(null); //inicio <- null
81         remove(posicao);
82     } else {
83         inicio = null;
84         fim = null;
85     }
86
87     return inicio.getItem();
88 }

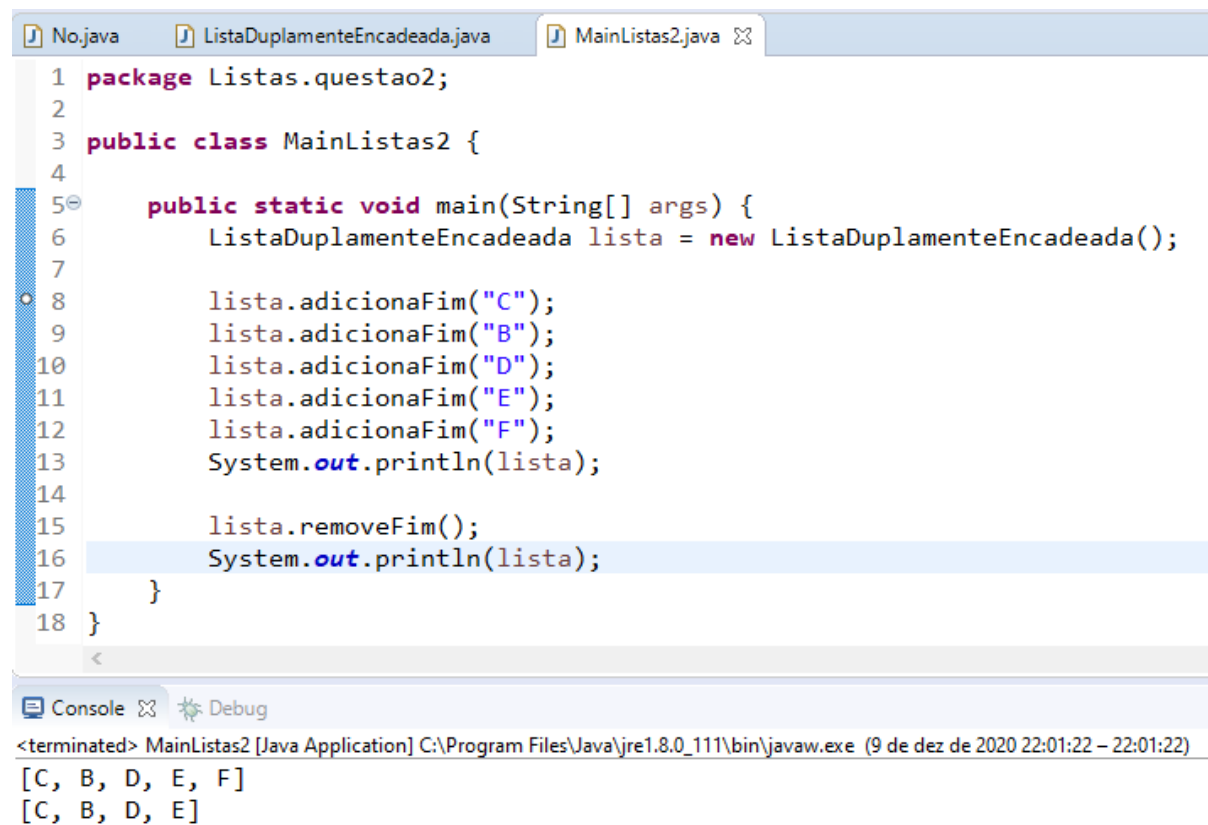
```

```

90 public Object removeFim() {
91     if(vazio()) return null;
92
93     if(fim.getAnt() != null) {
94         fim = fim.getAnt();
95         fim.getProx().setAnt(null);
96         fim.setProx(null);
97     }else {
98         inicio = null;
99         fim = null;
100     }
101     tamanho--;
102     return fim.getItem();
103 }
104
117 public Object[] getElementos() {..
120
121 public void setElementos(Object[] elementos) {..
124
125 public No getInicio() {..
128
129 public void setInicio(No inicio) {..
132
133 public No getFim() {..
136
137 public void setFim(No fim) {..
140
141 public int getTamanho() {..
144
145 public void setTamanho(int tamanho) {..
148
149 @Override
150 public String toString() {
151     StringBuilder s = new StringBuilder();
152     s.append("[");
153
154     for(int i=0; i<this.tamanho-1;i++) {
155         s.append(this.elementos[i]);
156         s.append(", ");
157     }
158     if(this.tamanho > 0) {
159         s.append(this.elementos[this.tamanho -1]);
160     }
161
162     s.append("]");

```

- MainLista2 e Terminal



The screenshot shows an IDE with three tabs: 'No.java', 'ListaDuplamenteEncadeada.java', and 'MainListas2.java'. The 'MainListas2.java' tab is active, displaying the following code:

```
1 package Listas.questao2;
2
3 public class MainListas2 {
4
5     public static void main(String[] args) {
6         ListaDuplamenteEncadeada lista = new ListaDuplamenteEncadeada();
7
8         lista.adicionaFim("C");
9         lista.adicionaFim("B");
10        lista.adicionaFim("D");
11        lista.adicionaFim("E");
12        lista.adicionaFim("F");
13        System.out.println(lista);
14
15        lista.removeFim();
16        System.out.println(lista);
17    }
18 }
```

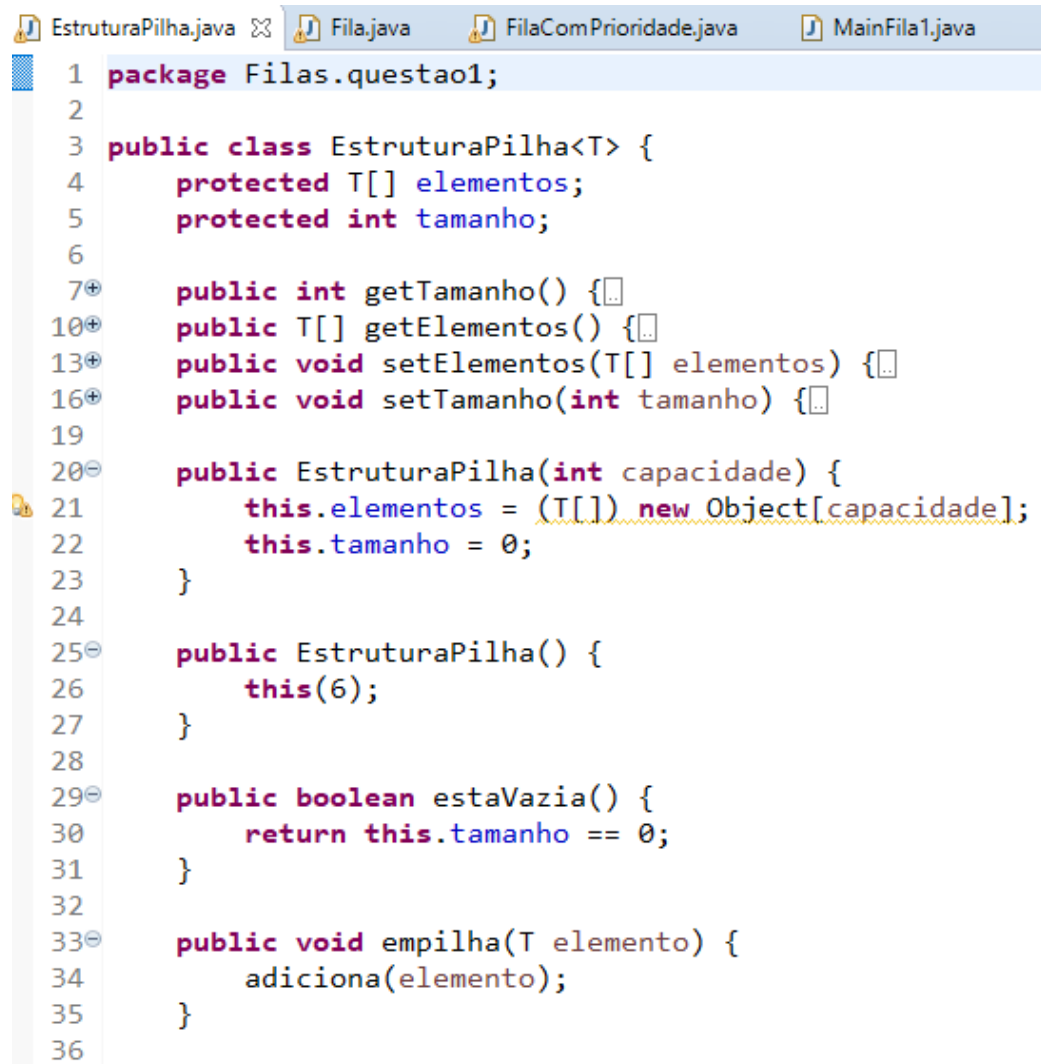
Below the code editor, the 'Console' tab is active, showing the following output:

```
<terminated> MainListas2 [Java Application] C:\Program Files\Java\jre1.8.0_111\bin\javaw.exe (9 de dez de 2020 22:01:22 - 22:01:22)
[C, B, D, E, F]
[C, B, D, E]
```

Questões Sobre Filas

1. Escreva uma função para inverter uma fila. Utilize uma pilha como estrutura de apoio para realizar a inversão. Faça a função para ambos os tipos de fila: estática e dinâmica.

- EstruturaPilha.java



```
1 package Filas.questao1;
2
3 public class EstruturaPilha<T> {
4     protected T[] elementos;
5     protected int tamanho;
6
7     public int getTamanho() {
8
9     }
10    public T[] getElementos() {
11
12    }
13    public void setElementos(T[] elementos) {
14
15    }
16    public void setTamanho(int tamanho) {
17
18    }
19
20    public EstruturaPilha(int capacidade) {
21        this.elementos = (T[]) new Object[capacidade];
22        this.tamanho = 0;
23    }
24
25    public EstruturaPilha() {
26        this(6);
27    }
28
29    public boolean estaVazia() {
30        return this.tamanho == 0;
31    }
32
33    public void empilha(T elemento) {
34        adiciona(elemento);
35    }
36
```

```
37 public T desempilha() {
38     if(this.estaVazia()) {
39         return null;
40     }
41     T elemento = this.elementos[tamanho - 1];
42     tamanho--;
43     return elemento;
44 }
45
46 public T topo() {
47     if(this.estaVazia()) {
48         return null;
49     }
50     return this.elementos[tamanho - 1];
51 }
52
53 protected boolean adiciona(T elemento) {
54     this.aumentaCapacidade();
55     if(this.tamanho < this.elementos.length) {
56         this.elementos[this.tamanho] = elemento;
57         this.tamanho++;
58         return true;
59     }
60     return false;
61 }
62
```

```

63  protected boolean adiciona(int posicao, T elemento) {
64      if(!(posicao >= 0 && posicao < tamanho)) {
65          throw new IllegalArgumentException("Posição Inválida");
66      }
67      this.aumentaCapacidade();
68
69      //mover todos os elementos
70      for(int i=this.tamanho - 1; i>=posicao; i--){
71          this.elementos[i + 1] = this.elementos[i]; // i=3 -> i=2
72      }
73      this.elementos[posicao] = elemento;
74      this.tamanho++;
75
76      return true;
77  }
78
79  protected void remove(int posicao) {
80      if(!(posicao >= 0 && posicao < tamanho)) {
81          throw new IllegalArgumentException("Posição Inválida");
82      }
83
84      //mover todos os elementos para trás
85      for(int i=posicao; i<this.tamanho - 1; i++){
86          this.elementos[i] = this.elementos[i+1]; // i=1 -> i=2
87      }
88
89      this.elementos[this.tamanho - 1] = null;
90      this.tamanho--; //o tamanho vai passar para 4 para tirar o ultimo F
91  }
92
93  protected int busca(T elemento) {
94      for(int i = 0; i<this.tamanho; i++) {
95          if(this.elementos[i].equals(elemento)) {
96              return i;
97          }
98      }
99      return -1; //(nao existe)
100  }
101
102  protected void aumentaCapacidade() {
103      if(this.tamanho == this.elementos.length) {
104          T[] elementosNovos = (T[]) new Object[this.elementos.length * 2];
105          for(int i=0; i<this.elementos.length; i++) {
106              elementosNovos[i] = this.elementos[i];
107          }
108          this.elementos = elementosNovos;
109      }
110  }
111
112  public String toString() {
113  }
114
115  public String toString() {
116  }
117
118  public String toString() {
119  }
120  }

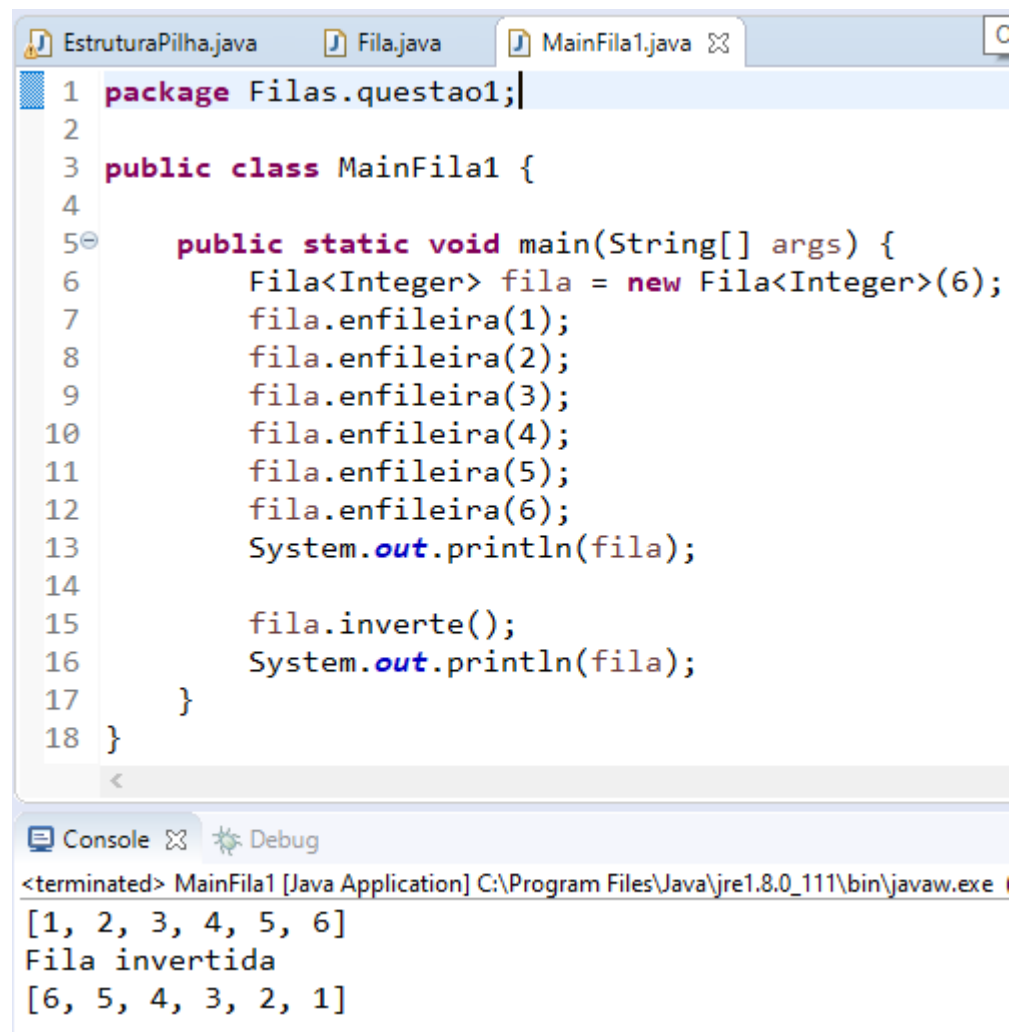
```

- Fila.java

Essa classe estende de EstruturaPilha.java pra reutilizar alguns métodos. Além de implementar o método `inverte` que inverte a ordem dos elementos.

```
EstruturaPilha.java  Fila.java  FilaComPrioridade.java  MainFila1.java
1 package Filas.questao1;
2
3 public class Fila<T> extends EstruturaPilha<T>{
4     public Fila() {
5         super();
6     }
7
8     public Fila(int capacidade) {
9         super(capacidade);
10    }
11
12    public void enqueue(T elemento) {
13        this.adiciona(elemento);
14    }
15
16    public void inverte() {
17        Fila<T> filaInvertida = new Fila<T>();
18
19        for(Integer i=this.elementos.length + 1; i>=0; i--) {
20            try {
21                filaInvertida.enqueue(this.elementos[i]);
22            }catch(ArrayIndexOutOfBoundsException e) {
23                e.getMessage();
24            }
25        }
26        System.out.println("Fila invertida");
27        this.elementos = filaInvertida.elementos;
28    }
29
30    public T dequeue() {
31        final int POS = 0;
32        if(this.estaVazia()) {
33            return null;
34        }
35
36        T elementoASerRemovido = this.elementos[POS];
37
38        this.remove(POS);
39
40        return elementoASerRemovido;
41    }
42
43    public String toString() {
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61 }
```

- MainFila1.java e Terminal



The screenshot shows an IDE with three tabs: `EstruturaPilha.java`, `Fila.java`, and `MainFila1.java`. The `MainFila1.java` tab is active, displaying the following Java code:

```
1 package Filas.questao1;
2
3 public class MainFila1 {
4
5     public static void main(String[] args) {
6         Fila<Integer> fila = new Fila<Integer>(6);
7         fila.enqueue(1);
8         fila.enqueue(2);
9         fila.enqueue(3);
10        fila.enqueue(4);
11        fila.enqueue(5);
12        fila.enqueue(6);
13        System.out.println(fila);
14
15        fila.inverte();
16        System.out.println(fila);
17    }
18 }
```

Below the code editor, the `Console` tab is active, showing the output of the program:

```
<terminated> MainFila1 [Java Application] C:\Program Files\Java\jre1.8.0_111\bin\javaw.exe
[1, 2, 3, 4, 5, 6]
Fila invertida
[6, 5, 4, 3, 2, 1]
```

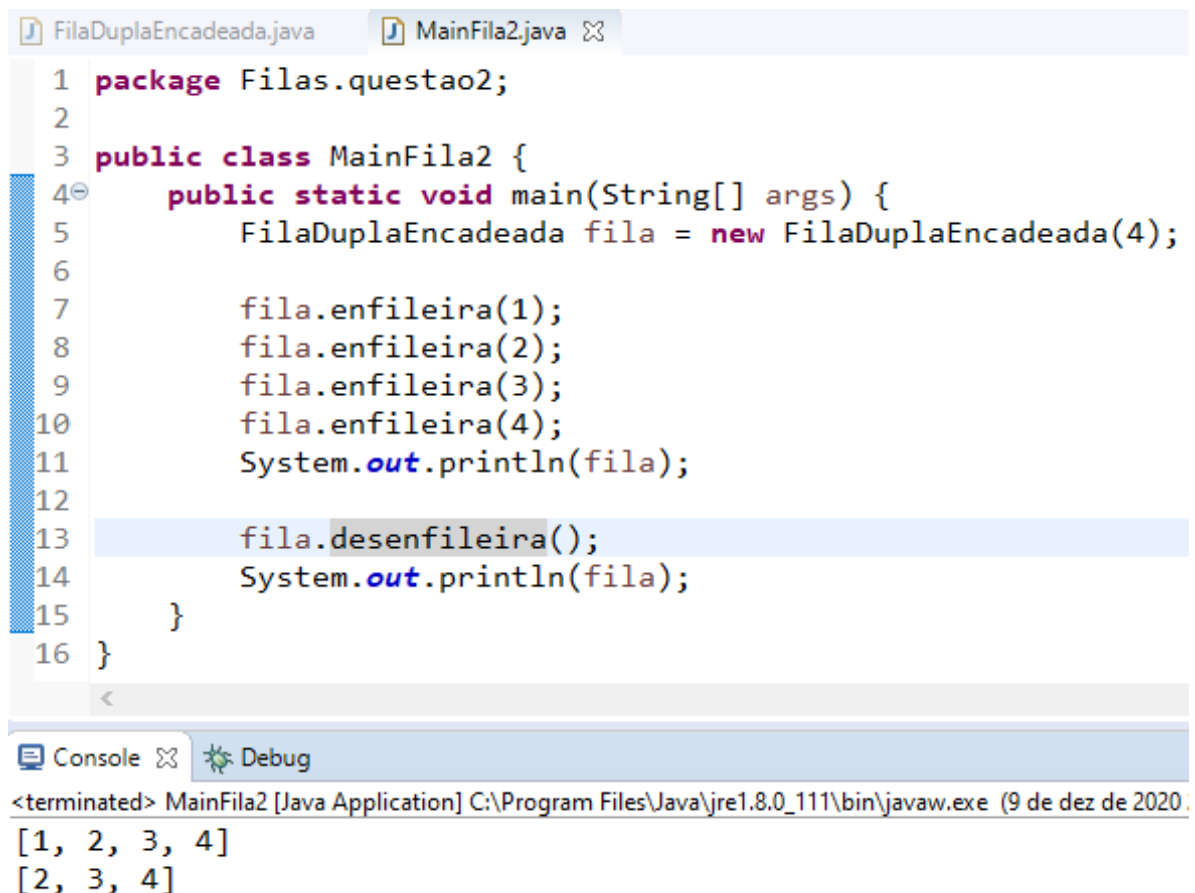

2. Utilizando a lista duplamente encadeada criada no exercício anterior, crie um TAD que faça essa lista funcionar como uma fila. Implemente as operações desse TAD.

- FilaDuplamenteEncadeada.java

Estendi a ListaDuplamenteEncadeada.java

```
FilaDuplaEncadeada.java  MainFila2.java
1 package Filas.questao2;
2
3 import Listas.questao2.ListaDuplamenteEncadeada;
4
5 public class FilaDuplaEncadeada extends ListaDuplamenteEncadeada{
6     public FilaDuplaEncadeada() {
7         super();
8     }
9
10    public FilaDuplaEncadeada(int capacidade) {
11        super(capacidade);
12    }
13
14    public void enqueue(Object item) {
15        this.adicionaFim(item);
16    }
17
18    public Object dequeue() {
19        final int POS = 0;
20        if(this.vazio()) {
21            return null;
22        }
23        Object elementoASerRemovido = this.elementos[POS];
24        this.removeInicio(POS);
25        return elementoASerRemovido;
26    }
27
28
29
30    public String toString() {
46 }
```

- MainFila2.java e Terminal



```

1 package Filas.questao2;
2
3 public class MainFila2 {
4     public static void main(String[] args) {
5         FilaDuplaEncadeada fila = new FilaDuplaEncadeada(4);
6
7         fila.enqueue(1);
8         fila.enqueue(2);
9         fila.enqueue(3);
10        fila.enqueue(4);
11        System.out.println(fila);
12
13        fila.dequeue();
14        System.out.println(fila);
15    }
16 }

```

Console

```

<terminated> MainFila2 [Java Application] C:\Program Files\Java\jre1.8.0_111\bin\javaw.exe (9 de dez de 2020)
[1, 2, 3, 4]
[2, 3, 4]

```

Questões Sobre Pilhas

1. Um dos problemas que podem ocorrer com expressões matemáticas é o uso incorreto de parênteses. Escreva uma função que utilize uma pilha para verificar se uma expressão está com a parentetização correta, ou seja, se existe um "fecha parênteses" para cada "abre parênteses" inserido, e vice-versa.

Exemplos:

- Correto: $() - (()) - ()$
- Incorreto: $)(- (()-)(($

- Pilha.java

```

1 package Pilhas.questao1;
2
3 public class Pilha{
4     private Object[] elementos;
5     private int tamanho;
6
7     public Object[] getElementos() { return elementos; }
8     public void setElementos(Object[] elementos) { this.elementos = elementos; }
9     public int getTamanho() { return tamanho; }
10    public void setTamanho(int tamanho) { this.tamanho = tamanho; }
11
12    public Pilha() { //tamanho padrão do tamanho caso o user não queira botar a capacidade
13        this(5);
14    }
15
16    public Pilha(int capacidade) {
17        this.elementos = (Object[]) new Object[capacidade]; //solução mais elegante
18        this.tamanho = 0;
19    }
20
21    public boolean empilha(Object elemento) {
22        this.aumentaCapacidade();
23        if(this.tamanho < this.elementos.length) {
24            this.elementos[this.tamanho] = elemento;
25            this.tamanho++;
26            return true;
27        }
28        return false;
29    }
30 }

```

A seguir, o método existeParenteseRepetido, porém ele dá erro. Eu tenho uma dificuldade de utilizar for com for...

```

32    public boolean existeParenteseRepetido() {
33        for(int i = 0; i < this.elementos.length; i++) {
34            for(int j=1; j < this.tamanho; j++) {
35
36                if(this.elementos[i].equals("(")) {
37                    if(this.elementos[i+j].equals(")")) {
38                        this.elementos[i+j] = this.elementos[i+j+1];
39
40                        return false;
41                    }
42                    return true;
43                }else {
44                    return false;
45                }
46            }
47        }
48        return false;
49    }
50 }
51
52 }

```

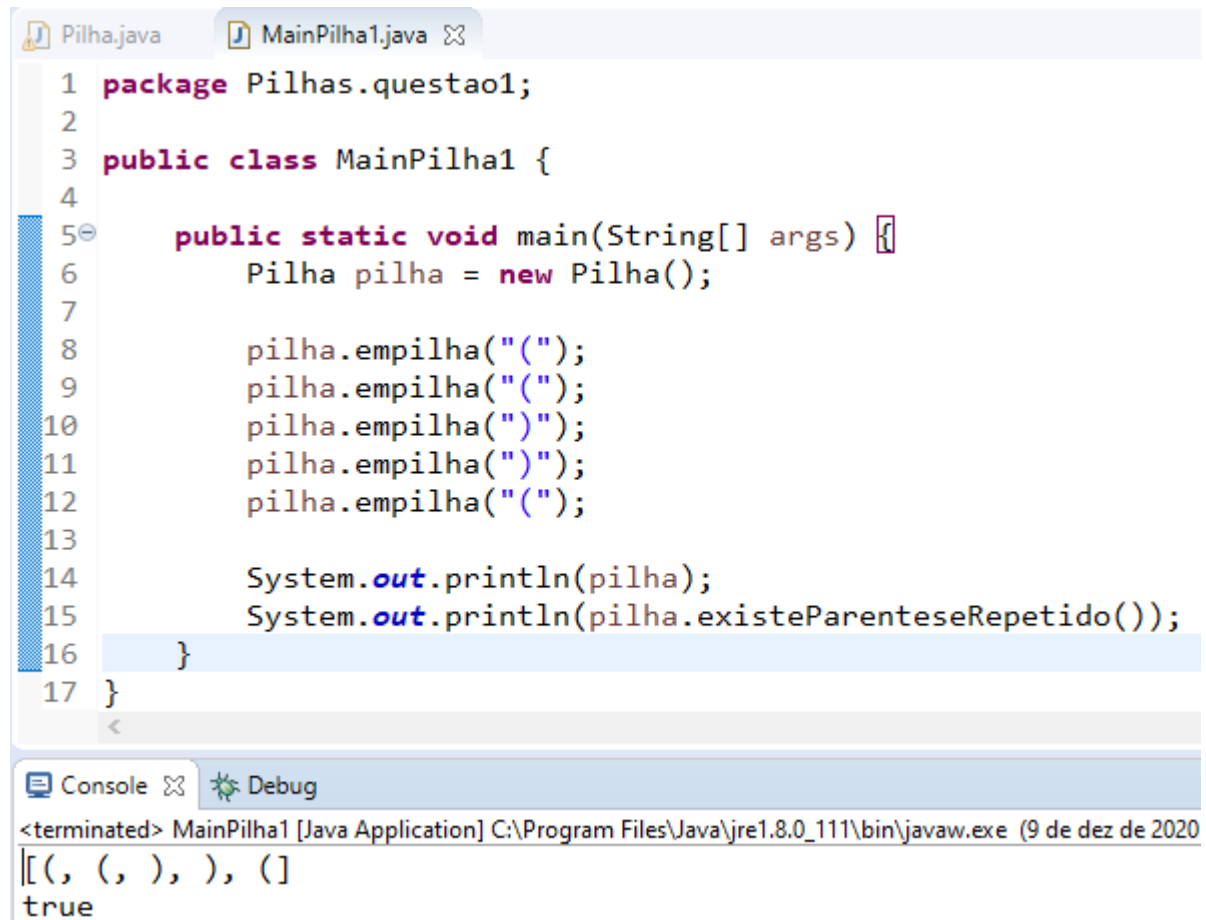
```

54 private void aumentaCapacidade() {
55     if(this.tamanho == this.elementos.length) {
56         Object[] elementosNovos =(Object[]) new Object[this.elementos.length * 2];
57         for(int i=0; i<this.elementos.length; i++) {
58             elementosNovos[i] = this.elementos[i];
59         }
60         this.elementos = elementosNovos;
61     }
62 }
63
64 public Object desempilha() {
65     if(this.estaVazia()) {
66         return null;
67     }
68     Object elemento = this.elementos[tamanho - 1];
69     tamanho--;
70     return elemento;
71 }
72
73 public Object topo() {
74     if(this.estaVazia()) {
75         return null;
76     }
77     return this.elementos[tamanho -1];
78 }
79
80 private boolean estaVazia() {
81     return this.tamanho == 0;
82 }
83
84 public String toString() {
85
86 }

```

- MainPilha1.java e Terminal

Como mencionei antes, da errado pois não verifica abre, abre, fecha, fecha parênteses. Ele dá como true só pelo fato de ter parêntese repetido.



The screenshot shows an IDE with two tabs: 'Pilha.java' and 'MainPilha1.java'. The 'MainPilha1.java' tab is active, displaying the following Java code:

```
1 package Pilhas.questao1;
2
3 public class MainPilha1 {
4
5     public static void main(String[] args) {
6         Pilha pilha = new Pilha();
7
8         pilha.empilha("(");
9         pilha.empilha("(");
10        pilha.empilha(")");
11        pilha.empilha(")");
12        pilha.empilha("(");
13
14        System.out.println(pilha);
15        System.out.println(pilha.existeParenteseRepetido());
16    }
17 }
```

Below the code editor, there is a 'Console' and 'Debug' tab. The 'Console' tab is active, showing the output of the program:

```
<terminated> MainPilha1 [Java Application] C:\Program Files\Java\jre1.8.0_111\bin\javaw.exe (9 de dez de 2020)
[(, (, ), ), (]
true
```

2. Escreva uma função que utilize uma pilha para fazer a conversão de um número em qualquer base para a base decimal. A função deve receber dois parâmetros:

- (a) A base na qual está representado o número a ser convertido
- (b) O ponteiro que referencia a pilha que contém os algarismos do número a ser convertido.

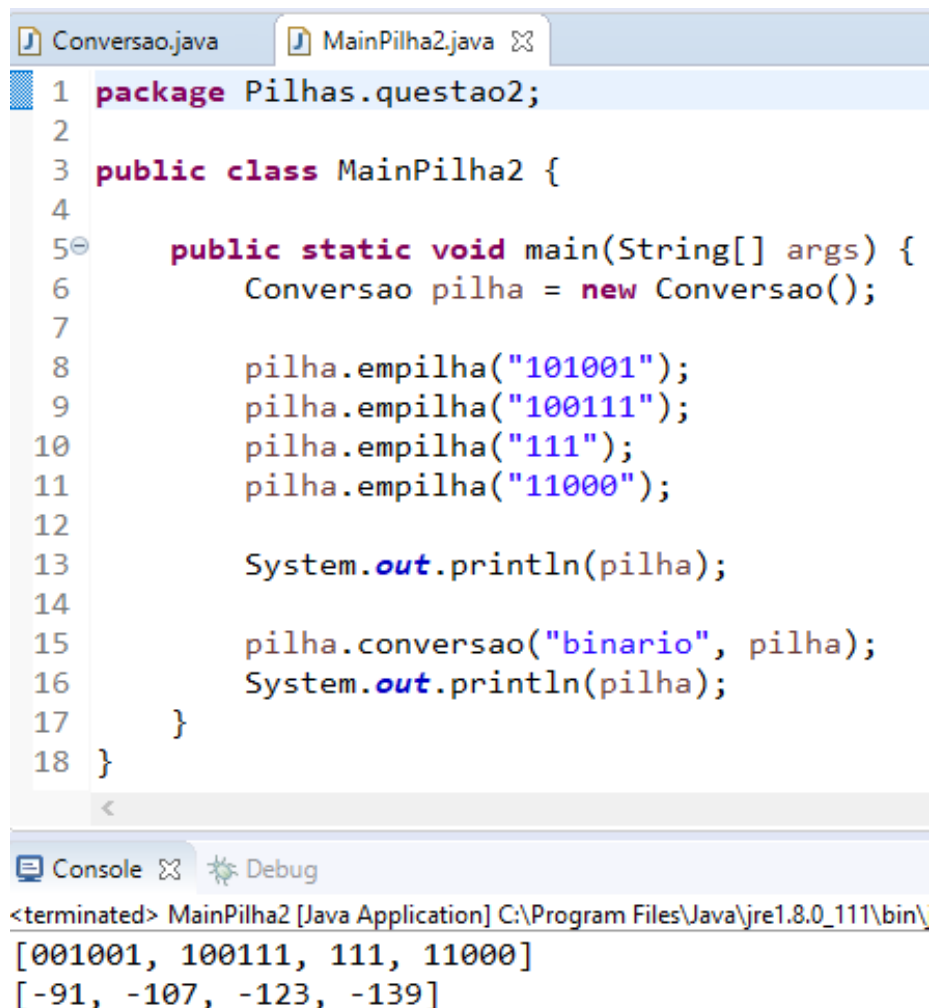
- Conversao.java

Conforme o int i vai percorrendo o vetor de elementos a 0, ele vai decrementando e fazendo a conversão do binário para o decimal e vai adicionando no vetor de elementos.

```
Conversao.java MainPilha2.java
1 package Pilhas.questao2;
2
3 import Pilhas.questao1.Pilha;
4
5 public class Conversao extends Pilha{
6     int potencia = 0;
7     int decimal = 0;
8
9     public void conversao(String base, Pilha pilha) {
10         Object[] elementosNovos = new Object[getElementos().length];
11
12         if(base.equalsIgnoreCase("binario")) {
13             for(int i = pilha.getElementos().length - 1; i >= 0; i--) {
14                 for(int j = 0; j < pilha.getElementos().length; j++) {
15                     decimal += Math.pow(2, potencia) * Character.getNumericValue(i);
16                     pilha.getElementos()[j] = decimal;
17                     elementosNovos[j] = pilha.getElementos()[j];
18                 }
19                 //pilha.getElementos() = elementosNovos; //não da certo isso
20                 potencia++;
21             }
22             pilha.getElementos().equals(elementosNovos);
23         }
24     }
25
26     public String toString() {
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44 }
45 }
```

- MainPilha2.java e Terminal

Porém de alguma forma não deu certo a conversão, tá muito errada, tentei.



The screenshot shows an IDE with two tabs: 'Conversao.java' and 'MainPilha2.java'. The 'MainPilha2.java' tab is active, displaying the following code:

```
1 package Pilhas.questao2;
2
3 public class MainPilha2 {
4
5     public static void main(String[] args) {
6         Conversao pilha = new Conversao();
7
8         pilha.empilha("101001");
9         pilha.empilha("100111");
10        pilha.empilha("111");
11        pilha.empilha("11000");
12
13        System.out.println(pilha);
14
15        pilha.conversao("binario", pilha);
16        System.out.println(pilha);
17    }
18 }
```

Below the code editor, the 'Console' tab is active, showing the output of the program:

```
<terminated> MainPilha2 [Java Application] C:\Program Files\Java\jre1.8.0_111\bin\
[001001, 100111, 111, 11000]
[-91, -107, -123, -139]
```