

LPOO
1º semestre de 2022
Prova 2
20/05/2022

Tempo limite: ≈75 Minutos

Nome: _____
RA: _____
Turma: _____

Professores: Vinicius Pereira

Esta prova contém 14 páginas e 12 questões.

Para exibir algo na saída padrão você pode usar o termo `cout()`.

Alguns métodos das interfaces `Set` e `Map`:

Set :

- `add(T o)`: adiciona um objeto `o` do tipo `T` no conjunto

Map :

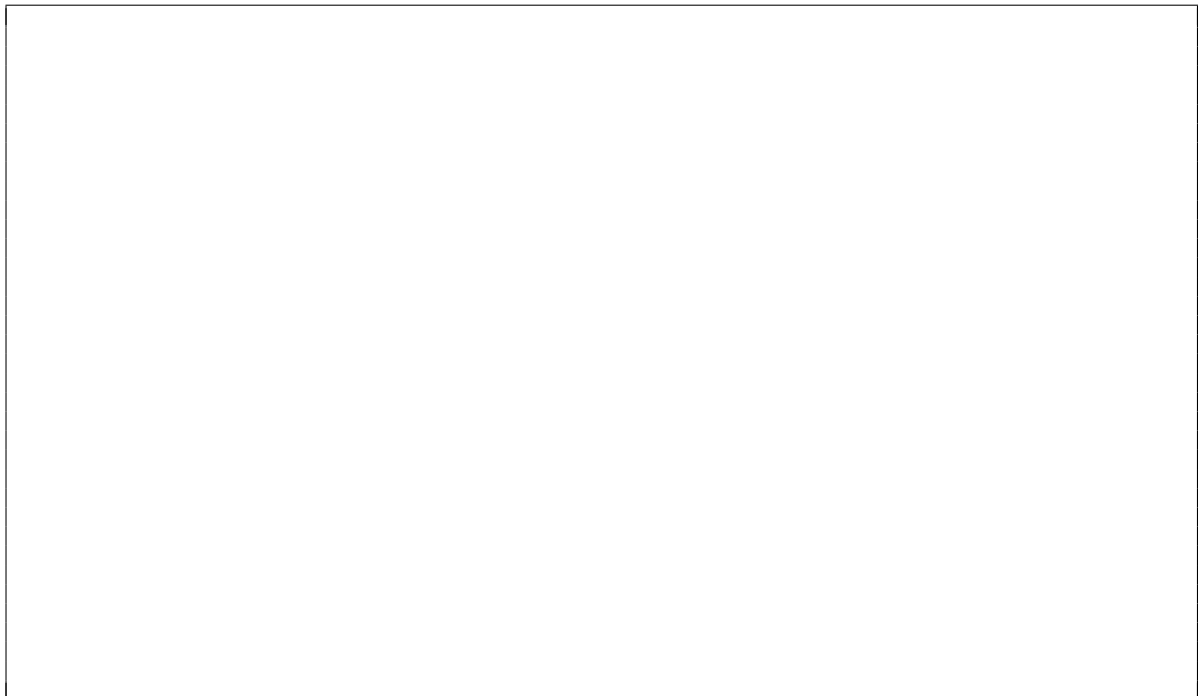
- `put(K k, V v)`: adiciona uma entrada com chave `k` do tipo `K` e valor `v` do tipo `V`.
Caso a chave já exista, substitui o valor da chave.
 - `V get(K k)`: retorna o valor relacionado à chave `k`
-

1. (1 ponto) Escreva o resultado do seguinte código
-

```
1 import java.util.Set;
2 import java.util.TreeSet;
3
4 class SetInt{
5
6     static public void main(String[] args){
7         Set<Integer> numeros = new TreeSet<>();
8         numeros.add(5);
9         numeros.add(7);
10
11         System.out.println(numeros);
12
13         numeros.add(3);
14         numeros.add(1);
15         numeros.add(9);
16
17         System.out.println(numeros);
18
19         numeros.add(5);
20         numeros.add(5);
21
22         System.out.println(numeros);
23
24     }
25 }
```

2. (1 ponto) Escreva o resultado do seguinte código

```
1 import java.util.Set;
2 import java.util.TreeSet;
3
4 class SetString{
5
6     static public void main(String[] args){
7         Set<String> palavras = new TreeSet<>();
8         palavras.add("car");
9         palavras.add("dex");
10
11         System.out.println(palavras);
12
13         palavras.add("ana");
14         palavras.add("ele");
15         palavras.add("bet");
16
17         System.out.println(palavras);
18
19         palavras.add("ana");
20         palavras.add("ana");
21
22         System.out.println(palavras);
23
24     }
25 }
```



3. (1 ponto) Escreva o resultado do seguinte código

```
1 import java.util.Map;
2 import java.util.TreeMap;
3
4 class MapInt{
5     static public void main(String[] args){
6
7         Map<Integer, String> numeros = new TreeMap<>();
8
9         numeros.put(3, "tres");
10        numeros.put(5, "cinco");
11
12        System.out.println(numeros);
13
14        numeros.put(1, "um");
15        numeros.put(7, "sete");
16
17        System.out.println(numeros);
18
19        numeros.put(5, "cinco");
20        numeros.put(5, "cinco");
21
22        System.out.println(numeros);
23
24    }
25 }
```

4. (1 ponto) Escreva o resultado do seguinte código

```
1 import java.util.Map;
2 import java.util.TreeMap;
3
4 class MapString{
5     static public void main(String[] args){
6
7         Map<String, Integer> numeros = new TreeMap<>();
8
9         numeros.put("cinco", 5);
10        numeros.put("tres", 3);
11
12        System.out.println(numeros);
13
14        numeros.put("um", 1);
15        numeros.put("sete", 7);
16
17        System.out.println(numeros);
18
19        numeros.put("cinco", 55);
20        numeros.put("cinco", 555);
21
22        System.out.println(numeros.get("cinco"));
23
24    }
25 }
```

Para as seguintes questões, considere as classes “Animal”, “Gato”, e “Cachorro”:

```
1 abstract class Animal{
2
3     private String nome;
4
5     public Animal(String aNome){
6         this.nome = aNome;
7     }
8
9     public String getNome(){return this.nome;}
10
11    public void dorme(){
12        System.out.println("O animal " + this.nome + " dorme");
13    }
14
15    abstract public void fazBarulho();
16
17    @Override
18    public String toString(){
19        String res = "Animal\n";
20        res += "Nome: " + this.nome + "\n";
21        return res;
22    }
23 }
```

```
1 class Gato extends Animal{
2     public Gato(String aNome){
3         super(aNome);
4     }
5     @Override
6     public void fazBarulho(){
7         System.out.println("O gato " + this.getNome() + " mia");
8     }
9 }
```

```
1 class Cachorro extends Animal{
2     public Cachorro(String aNome){
3         super(aNome);
4     }
5     @Override
6     public void fazBarulho(){
7         System.out.println("O cachorro " + this.getNome() + " late");
8     }
9 }
```

5. (1 ponto) Verifique se o código a seguir será compilado. Se for, escreva o resultado do código, caso não seja, explique porque não irá compilar.

```
1 class TesteAnimal{
2     static public void main(String[] args){
3         Animal animal = new Animal("Ani");
4
5         System.out.println(animal);
6
7         animal.fazBarulho();
8
9     }
10 }
```

6. (1 ponto) Verifique se o código a seguir será compilado. Se for, escreva o resultado do código, caso não seja, explique porque não irá compilar.

```
1 class TesteGato{
2     static public void main(String[] args){
3         Animal gato = new Gato("Chi");
4
5         System.out.println(gato);
6
7         gato.fazBarulho();
8
9     }
10 }
```

7. (1 ponto) Verifique se o código a seguir será compilado. Se for, escreva o resultado do código, caso não seja, explique porque não irá compilar.

```
1 class TesteCachorro{
2     static public void main(String[] args){
3         Animal cachorro = new Cachorro("Brutus");
4
5         System.out.println(cachorro);
6
7         cachorro.fazBarulho();
8
9     }
10 }
```

Para as seguintes questões, considere a interface “Voavel” e as classes “SuperHomem” e “Aviao”:

```
1 interface Voavel{
2     void voa();
3 }
```

```
1 class SuperHomem implements Voavel{
2     public SuperHomem(){
3     }
4     public void voa(){
5         System.out.println("SuperHomem voa como um jato");
6     }
7     @Override
8     public String toString(){
9         return "SuperHomem";
10    }
11 }
```

```
1 class Aviao implements Voavel{
2     String modelo;
3     double velocidade;
4     public Aviao(String aModelo, double aVelocidade){
5         this.modelo = aModelo;
6         this.velocidade = aVelocidade;
7     }
8     public void voa(){
9         System.out.println("O aviao " + this.modelo + " voa a " +
10            this.velocidade + " km/h");
11    }
12    @Override
13    public String toString(){
14        String res = "";
15        res += "Aviao " + this.modelo + "\n";
16        res += "Velocidade maxima: " + this.velocidade + "\n";
17        return res;
18    }
19 }
```

8. (1 ponto) Verifique se o código a seguir será compilado. Se for, escreva o resultado do código, caso não seja, explique porque não irá compilar.

```
1 class TesteVoavel{
2     static public void main(String[] args){
3         Voavel voavel = new Voavel();
4
5         System.out.println(voavel);
6
7         voavel.voa();
8
9     }
10 }
```

9. (1 ponto) Verifique se o código a seguir será compilado. Se for, escreva o resultado do código, caso não seja, explique porque não irá compilar.

```
1 class TesteSuperHomem{
2     static public void main(String[] args){
3         Voavel voavel = new SuperHomem();
4
5         System.out.println(voavel);
6
7         voavel.voa();
8
9     }
10 }
```

10. (1 ponto) Verifique se o código a seguir será compilado. Se for, escreva o resultado do código, caso não seja, explique porque não irá compilar.

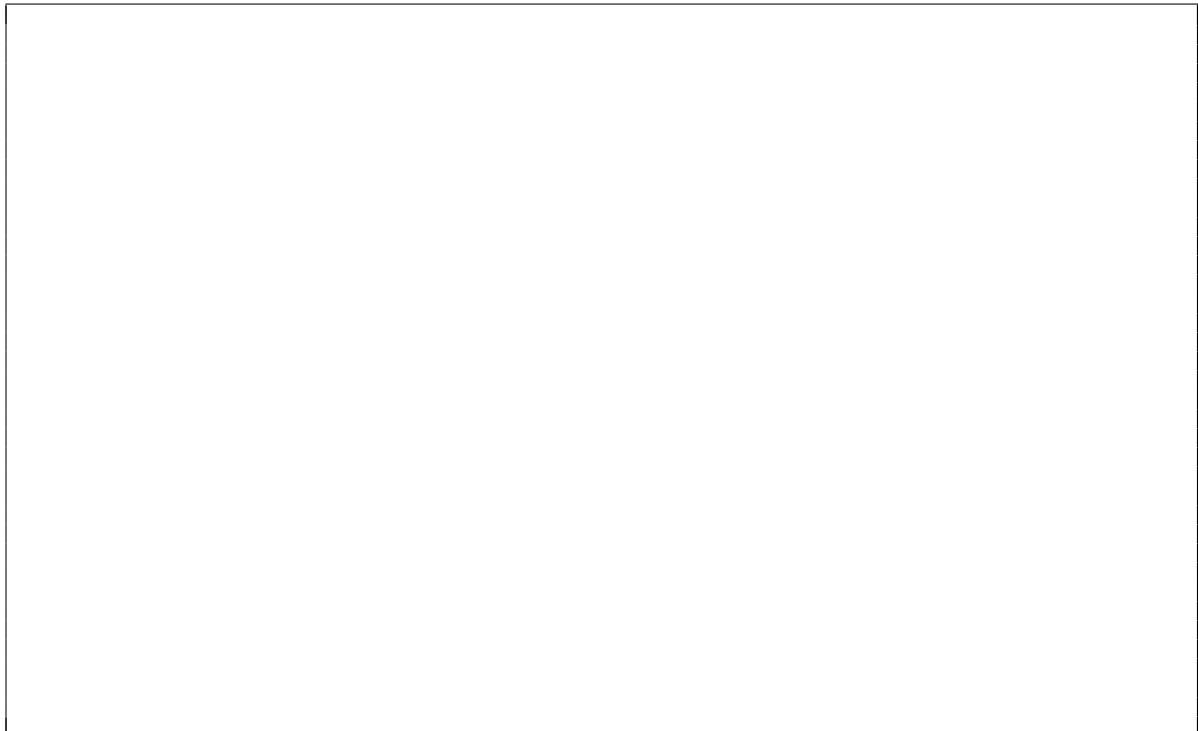
```
1 class TesteAviao{
2     static public void main(String[] args){
3         Voavel voavel = new Aviao("Boo", 101);
4
5         System.out.println(voavel);
6
7         voavel.voa();
8
9     }
10 }
```

Para as seguintes questões, considere a classe “Pato”:

```
1 import java.util.Objects;
2
3 class Pato implements Comparable<Pato>{
4
5     private String nome;
6
7     public Pato(String aNome){
8         this.nome = aNome;
9     }
10
11     public String getNome(){return this.nome;}
12
13     @Override
14     public boolean equals(Object outro){
15         if(this==outro) {return true;};
16         if(outro==null){return false;};
17         if(this.getClass()!=outro.getClass()){return false;};
18
19         Pato outroPato = (Pato)outro;
20
21         return Objects.equals(this.getNome(), outroPato.getNome());
22     }
23
24
25     @Override
26     public int compareTo(Pato outroPato){
27         return this.getNome().compareTo(outroPato.getNome());
28     }
29
30     @Override
31     public String toString(){
32         return "Pato " + this.getNome() + "\n";
33     }
34 }
```

11. (2 pontos) Escreva o resultado do seguinte código

```
1 import java.util.Set;
2 import java.util.TreeSet;
3
4 class SetPato{
5
6     static public void main(String[] args){
7         Set<Pato> patos = new TreeSet<>();
8         patos.add(new Pato("car"));
9         patos.add(new Pato("dex"));
10
11         System.out.println(patos);
12
13         patos.add(new Pato("ana"));
14         patos.add(new Pato("ele"));
15
16         System.out.println(patos);
17
18         patos.add(new Pato("ana"));
19         patos.add(new Pato("ana"));
20
21         System.out.println(patos);
22     }
23 }
24 }
```



12. (2 pontos) Escreva o resultado do seguinte código

```
1  import java.util.Map;
2  import java.util.TreeMap;
3
4  class MapPato{
5      static public void main(String[] args){
6
7          Map<Pato, String> patos_tel = new TreeMap<>();
8
9          patos_tel.put(new Pato("car"), "123");
10         patos_tel.put(new Pato("dex"), "123");
11
12         System.out.println(patos_tel);
13
14         patos_tel.put(new Pato("ana"), "321");
15         patos_tel.put(new Pato("ele"), "321");
16
17         System.out.println(patos_tel);
18
19         patos_tel.put(new Pato("ana"), "333");
20         patos_tel.put(new Pato("ana"), "101");
21
22         System.out.println(patos_tel.get(new Pato("ana")));
23
24     }
25 }
```

