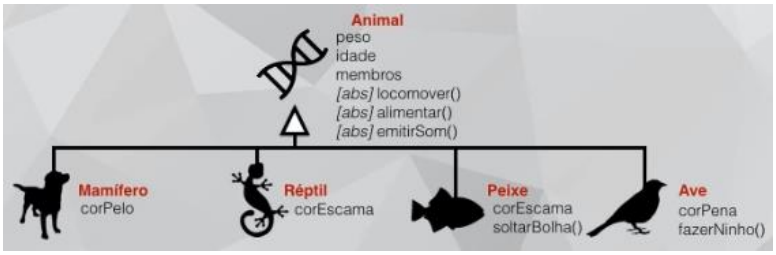
Java

Polimorfismo - Animais



Crie a Classe Principal (main).

```
package aula12;
public class Aula12 {
    public static void main(String[] args) {
        |
    }
}
```

Crie uma Nova Classe.

```
package aula12;
public class Animal {
|
}
```

A Classe Animal deverá ficar da seguinte forma; para isso, siga os passos.

```
classe abstrata Animal
  protegido peso: Real
  protegido idade: Inteiro
  protegido membros: Inteiro
  publico metodo abstrato locomover()
  publico metodo abstrato alimentar()
  publico metodo abstrato emitirSom()
FimClasse
classe Mamifero estende Animal
  privado corPelo: Caractere
  @Sobrepor
  publico metodo locomover()
     Escreva("Correndo")
  fimMetodo
  @Sobrepor
  publico metodo alimentar()
     Escreva("Mamando")
  fimMetodo
  @Sobrepor
  publico metodo emitirSom()
     Escreva("som de Mamífero")
  fimMetodo
FimClasse
```

Crie os Atributos.

```
protegido peso: Real
protegido idade: Inteiro
protegido membros: Inteiro
```

```
// Atributos de Animal
protected float peso;
protected int idade;
protected int membros;
```

Crie os Métodos Abstratos.

```
publico metodo abstrato locomover()
publico metodo abstrato alimentar()
publico metodo abstrato emitirSom()
```

```
public abstract void locomover();
public abstract void alimentar();
public abstract void emitirSom();
```

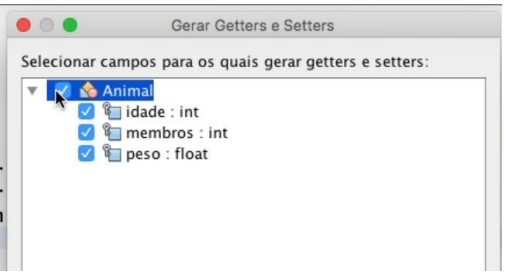
Observe que o NetBeans dá um alerta, isto pelo fato de que se os Métodos são Abstratos, a Classe também deve ser Abstrata. Basta clicar na lâmpada.

```
public class Animal {
```

```
public abstract class Animal {
```

Insira os Códigos dos Métodos Especiais Getters e Setters clicando com o botão direito do mouse.

```
package aula12;
public abstract class Animal {
    // Atributos de Animal
    protected float peso;
    protected int idade;
    protected int membros;
    // Métodos de Animal
    public abstract void locomover
    public abstract void alimentar
    public abstract void emitirSom
}
```




```
package aula12;
public abstract class Animal {
    // Atributos de Animal
    protected float peso;
    protected int idade;
    protected int membros;
    // Métodos de Animal
    public abstract void locomover();
    public abstract void alimentar();
    public abstract void emitirSom();

    public float getPeso() {
        return peso;
    }

    public void setPeso(float peso) {
        this.peso = peso;
    }

    public int getIdade() {
        return idade;
    }
```

Crie uma Nova Classe.

```
package aula12;
public class Mamifero  {

}
```

Edite o conteúdo acrescentando “extends Animal”, ou seja, a Classe “Mamifero” é uma extensão da Classe “Animal”; afinal, mamífero é um tipo de animal.

```
1  package aula12;
   public class Mamifero extends Animal {
3      Implementar Todos os Métodos Abstratos
4      Tornar Mamifero classe abstrata
```

Observe que ao acrescentar “extends Animal”, o NetBeans acusará com um alerta. Isto se dá devido a “Classe Animal” ser uma Classe Abstrata, logo deveremos implementar os Métodos Abstratos.

```
classe Mamifero estende Animal
  privado corPelo: Caractere
  @Sobrepor
  publico metodo locomover()
     Escreva(“Correndo”)
  fimMetodo
  @Sobrepor
  publico metodo alimentar()
     Escreva(“Mamando”)
  fimMetodo
  @Sobrepor
  publico metodo emitirSom()
     Escreva(“som de Mamífero”)
  fimMetodo
FimClasse
```

```
1  package aula12;
2  public class Mamifero extends Animal {
3
4      @Override
5      public void locomover() {
6          throw new UnsupportedOperationE
7      }
8
9      @Override
10     public void alimentar() {
11         throw new UnsupportedOperationE
12     }
13
14     @Override
15     public void emitirSom() {
16         throw new UnsupportedOperationE
17     }
18
19 }
```

Acrescente o Atributo cor do pelo.

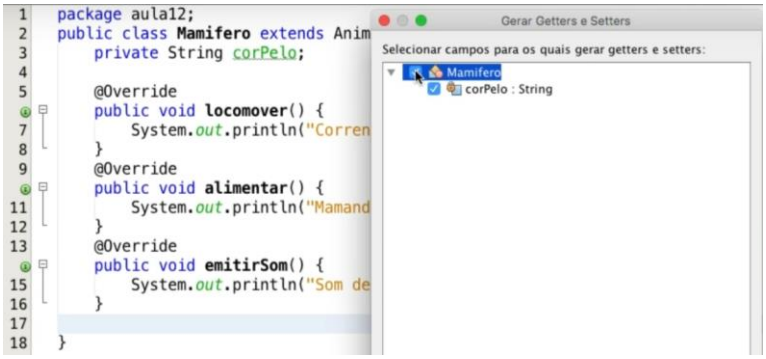
```
package aula12;
public class Mamifero extends Animal {
    private String corPelo;
```

Edite o retorno dos Métodos Abstratos.

```
classe Mamifero estende Animal
  privado corPelo: Caractere
  @Sobrepor
  publico metodo locomover()
     Escreva(“Correndo”)
  fimMetodo
  @Sobrepor
  publico metodo alimentar()
     Escreva(“Mamando”)
  fimMetodo
  @Sobrepor
  publico metodo emitirSom()
     Escreva(“som de Mamífero”)
  fimMetodo
FimClasse
```

```
1  package aula12;
2  public class Mamifero extends Animal {
3      private String corPelo;
4
5      @Override
6      public void locomover() {
7          System.out.println("Correndo");
8      }
9      @Override
10     public void alimentar() {
11         System.out.println("Mamando");
12     }
13     @Override
14     public void emitirSom() {
15         System.out.println("Som de Mamífero");
16     }
17
18 }
```

Insira os Métodos Getters e Setters.




```
9      @Override
10     public void alimentar() {
11         System.out.println("Mamando");
12     }
13     @Override
14     public void emitirSom() {
15         System.out.println("Som de Mamífero");
16     }
17
18     public String getCorPelo() {
19         return corPelo;
20     }
21
22     public void setCorPelo(String corPelo) {
23         this.corPelo = corPelo;
24     }
```

Já temos os animais mamíferos, agora crie os répteis.

```
classe Reptil estende Animal
  privado corEscama: Caractere
  @Sobrepor
  publico metodo locomover()
     Escreva("Rastejando")
  fimMetodo
  @Sobrepor
  publico metodo alimentar()
     Escreva("Comendo Vegetais")
  fimMetodo
  @Sobrepor
  publico metodo emitirSom()
     Escreva("som de Réptil")
  fimMetodo
FimClasse
```

Crie uma Nova Classe, edite a implementação para os Métodos Abstratos e inclua o Atributo cor da escama.

```
package aula12;
public class Reptil {

}
```

```
classe Reptil estende Animal
  privado corEscama: Caractere
```

```
1  package aula12;
2  public class Reptil extends Animal {
3      private String corEscama;
4
5  }
```

O alerta para inclusão dos Métodos Abstratos é exibido.

```
1  package aula12;
2  public class Reptil extends Animal {
3      Implementar Todos os Métodos Abstratos
4      Tornar Reptil classe abstrata
5  }
```

```
@Sobrepor
publico metodo locomover()
   Escreva("Rastejando")
fimMetodo
@Sobrepor
publico metodo alimentar()
   Escreva("Comendo Vegetais")
fimMetodo
@Sobrepor
publico metodo emitirSom()
   Escreva("som de Réptil")
fimMetodo
```

```
4      @Override
5      public void locomover() {
6          System.out.println("Rastejando");
7      }
8      @Override
9      public void alimentar() {
10         System.out.println("Comendo Vegetais");
11     }
12     @Override
13     public void emitirSom() {
14         System.out.println("Som de Reptil");
15     }
```

Inclua os Métodos Getters e Setters.

```
8      @Override
9      public void alimentar() {
10         System.out.println("Comendo Vegetais");
11     }
12     @Override
13     public void emitirSom() {
14         System.out.println("Som de Reptil");
15     }
16
17     public String getCorEscama() {
18         return corEscama;
19     }
20
21     public void setCorEscama(String corEscama) {
22         this.corEscama = corEscama;
23     }
```

Agora crie a Classe dos animais categorizados como peixes.

```
classe Peixe estende Animal
  privado corEscama: Caractere
  @Sobrepor
  publico metodo locomover()
     Escreva("Nadando")
  fimMetodo
  @Sobrepor
  publico metodo alimentar()
     Escreva("Comendo substâncias")
  fimMetodo
  @Sobrepor
  publico metodo emitirSom()
     Escreva("Peixe não faz som")
  fimMetodo
  publico metodo soltarBolha()
     Escreva("Soltou uma bolha")
  fimMetodo
FimClasse
```

Crie a nova Classe.

```
package aula12;
public class Peixe {

}
```

Edite as informações da Classe.

```
classe Peixe estende Animal
  privado corEscama: Caractere
```

```
1  package aula12;
2  public class Peixe extends Animal {
3      private String corEscama;
4
5  }
```

Insira os Métodos Abstratos.

```
1  package aula12;
2  public class Peixe extends Animal {
3
4
5  }
```

Implementar Todos os Métodos Abstratos
Tornar Peixe classe abstrata

```
1  package aula12;
2  public class Peixe extends Animal {
3      private String corEscama;
4
5      @Override
7      public void locomover() {
8          throw new UnsupportedOperati
9      }
10
12     @Override
13     public void alimentar() {
14         throw new UnsupportedOperati
15     }
16
17     @Override
19     public void emitirSom() {
20         throw new UnsupportedOperati
```

Edite os Métodos Abstratos, conforme o algoritmo.

```
classe Peixe estende Animal
  privado corEscama: Caractere
  @Sobrepor
  publico metodo locomover()
     Escreva("Nadando")
  fimMetodo
  @Sobrepor
  publico metodo alimentar()
     Escreva("Comendo substâncias")
  fimMetodo
  @Sobrepor
```

```
1  package aula12;
2  public class Peixe extends Animal {
3      private String corEscama;
4      @Override
6      public void locomover() {
7          System.out.println("Nadando");
8      }
9      @Override
10     public void alimentar() {
11         System.out.println("Comendo Substâncias");
12     }
13     @Override
```

Crie o Método para impressão do texto.

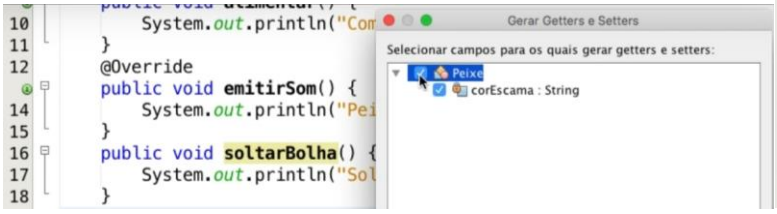
```
  publico metodo emitirSom()
     Escreva("Peixe não faz som")
  fimMetodo

publico metodo soltarBolha()
   Escreva("Soltou uma bolha")
fimMetodo
```

```
10     public void alimentar() {
11         System.out.println("Comendo Substâncias");
12     }

16     public void soltarBolha() {
17         System.out.println("Soltando Bolha");
```

Crie os Métodos Getters e Setters.




```
11     }
12     @Override
14     public void emitirSom() {
15         System.out.println("Peixe não faz som");
16     }
17     public void soltarBolha() {
18         System.out.println("Soltando Bolha");
19     }
20     public String getCorEscama() {
21         return corEscama;
22     }
23
24     public void setCorEscama(String corEscama) {
25         this.corEscama = corEscama;
26     }
```

Insira uma Nova Classe.

```
classe Ave estende Animal
  privado corPena: Caractere
  @Sobrepor
  publico metodo locomover()
     Escreva("Voando")
  fimMetodo
  @Sobrepor
  publico metodo alimentar()
     Escreva("Comendo frutas")
  fimMetodo
  @Sobrepor
  publico metodo emitirSom()
     Escreva("Som de ave")
  fimMetodo
  publico metodo fazerNinho()
     Escreva("Construiu um ninho")
  fimMetodo
FimClasse
```

```
package aula12;
public class Ave {

}
```

```
1  package aula12;
2  public class Ave extends Animal {
3      private String corPena;
4
5  }
```

Implemente os Métodos Abastratos.

Inclua o Método.

```
public void fazerNinho() {
    System.out.println("Construindo Ninho");
}
```

Insira os Métodos Getters e Setters.

```
    }
    @Override
    public void emitirSom() {
        System.out.println("Som de Ave");
    }
    public void fazerNinho() {
        System.out.println("Construindo Ninho");
    }

    public String getCorPena() {
        return corPena;
    }

    public void setCorPena(String corPena) {
        this.corPena = corPena;
    }
```

Agora retorno a Classe Principal.

Verifique que não é possível instanciar a Classe Animal pelo fato de ser Abastrato.

```
package aula12;
public class Aula12 {
    public static void main(String[] args) {
        Animal n = new Animal();
    }
}
```

Crie os demais animais abaixo:

```
package aula12;
public class Aula12 {
    public static void main(String[] args) {
        // Animal n = new Animal();
        Mamifero m = new Mamifero();
        Reptil r = new Reptil();
        Peixe p = new Peixe();
        Ave a = new Ave();

    }
}
```

Acrescente os itens abaixo e faça o teste sobre os mamíferos.

```
1  package aula12;
2  public class Aula12 {
3      public static void main(String[] args) {
4          // Animal n = new Animal();
5          Mamifero m = new Mamifero();
6          Reptil r = new Reptil();
7          Peixe p = new Peixe();
8          Ave a = new Ave();
9
10         m.setPeso(35.3f);
11         m.setCorPelo("Marrom");
12         m.alimentar();
13         m.locomover();
14         m.emitirSom();
15
16     }
17 }
```

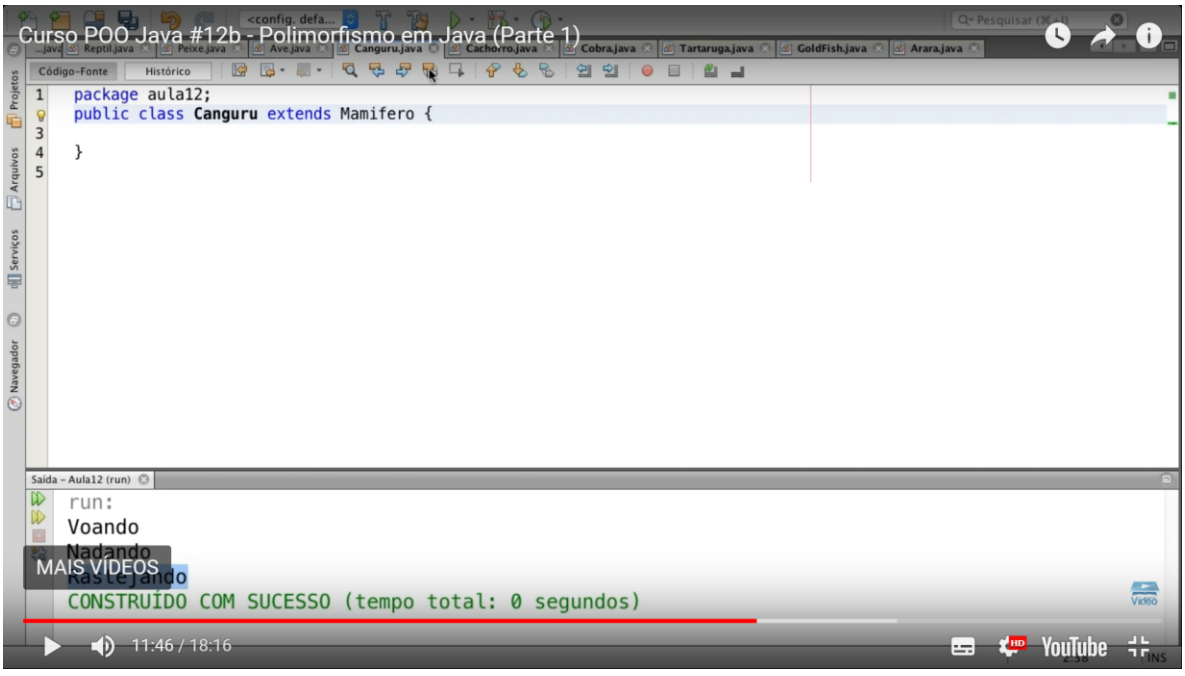
```
run:
Mamando
Correndo
Som de Mamífero
CONSTRUÍDO COM SUCESSO (tempo to
```

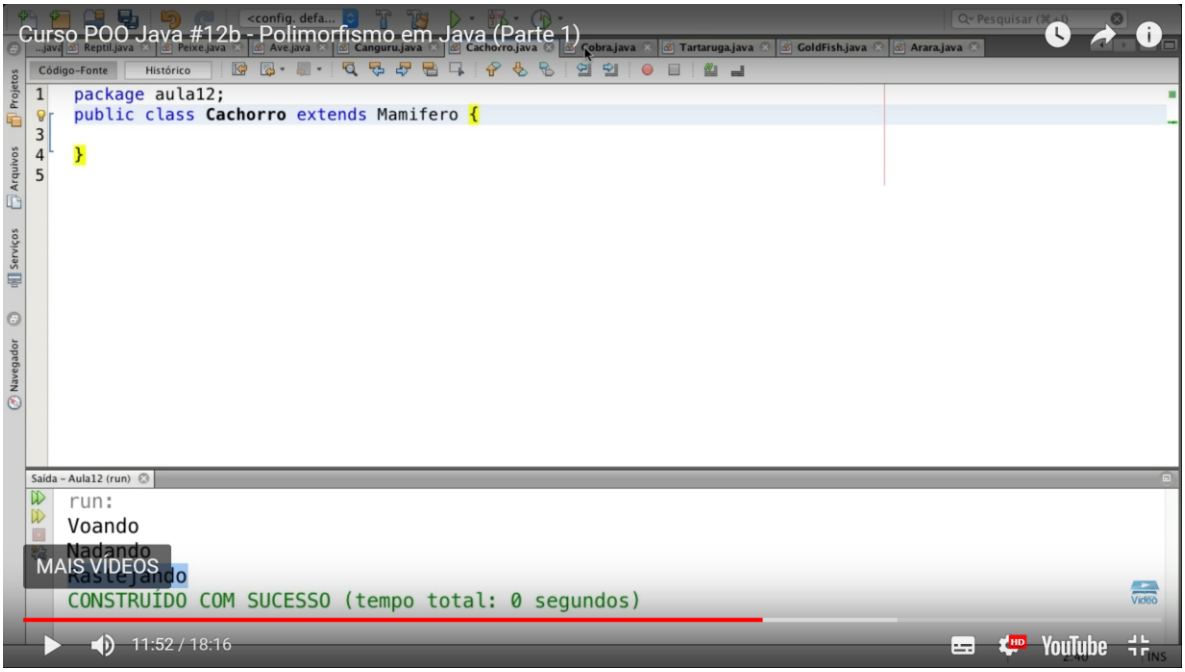
Faça o tese sobre a forma locomoção das Aves, Peixes e Répteis.

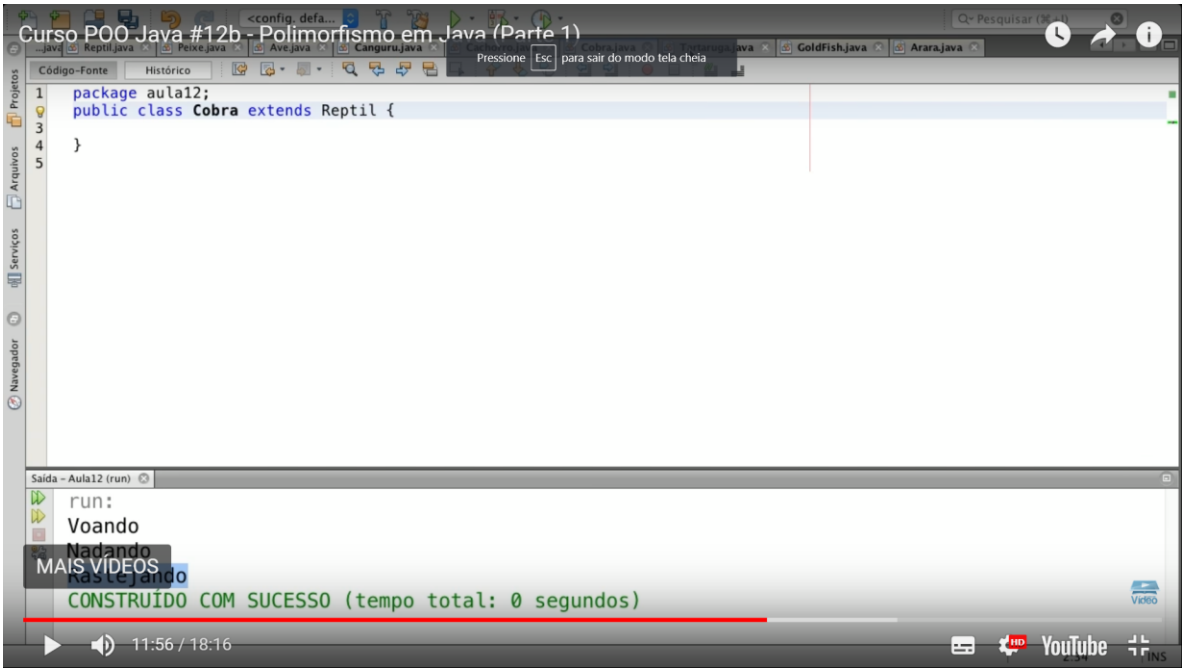
```
1  package aula12;
2  public class Aula12 {
3      public static void main(String[] args) {
4          // Animal n = new Animal();
5          Mamifero m = new Mamifero();
6          Reptil r = new Reptil();
7          Peixe p = new Peixe();
8          Ave a = new Ave();
9
10
11         a.locomover();
12         p.locomover();
13         r.locomover();
14
15     }
16 }
```

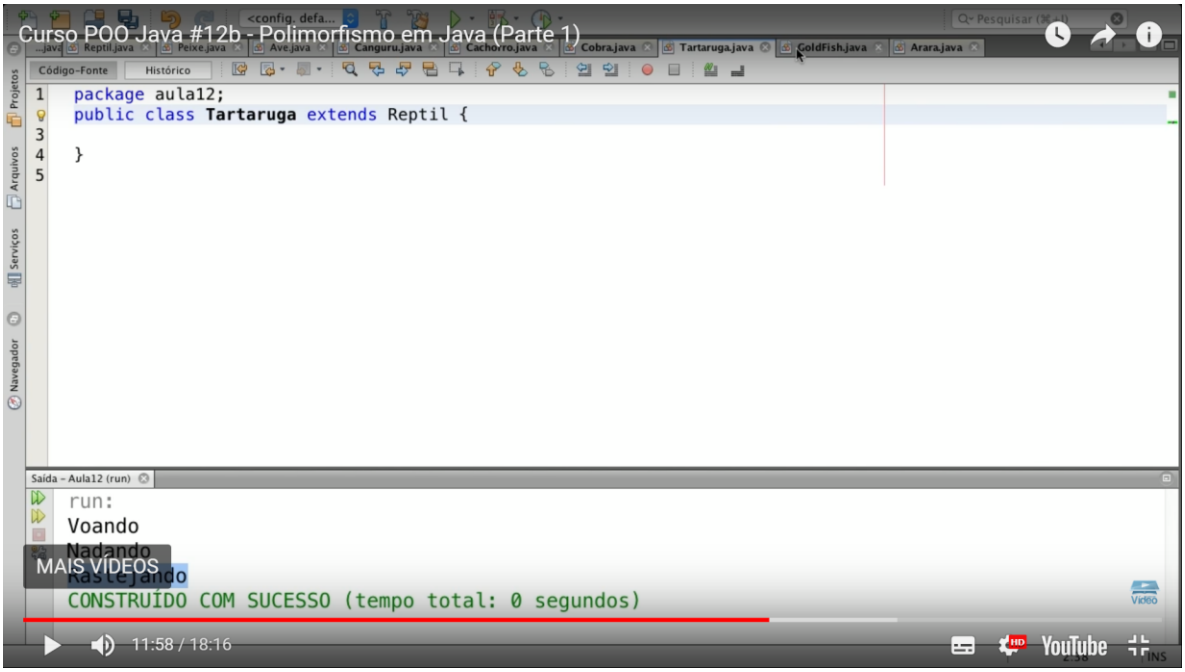
```
run:
Voando
Nadando
Rastejando
CONSTRUÍDO COM SUCESSO (tempo to
```

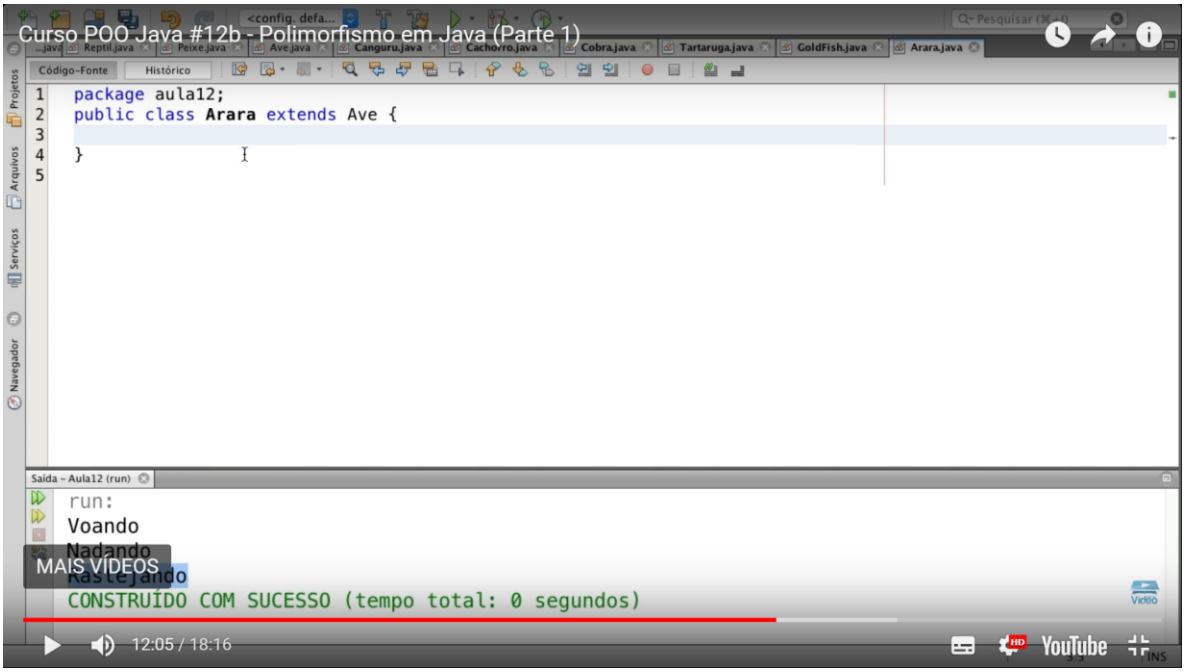
Crie as Classes abaixo:

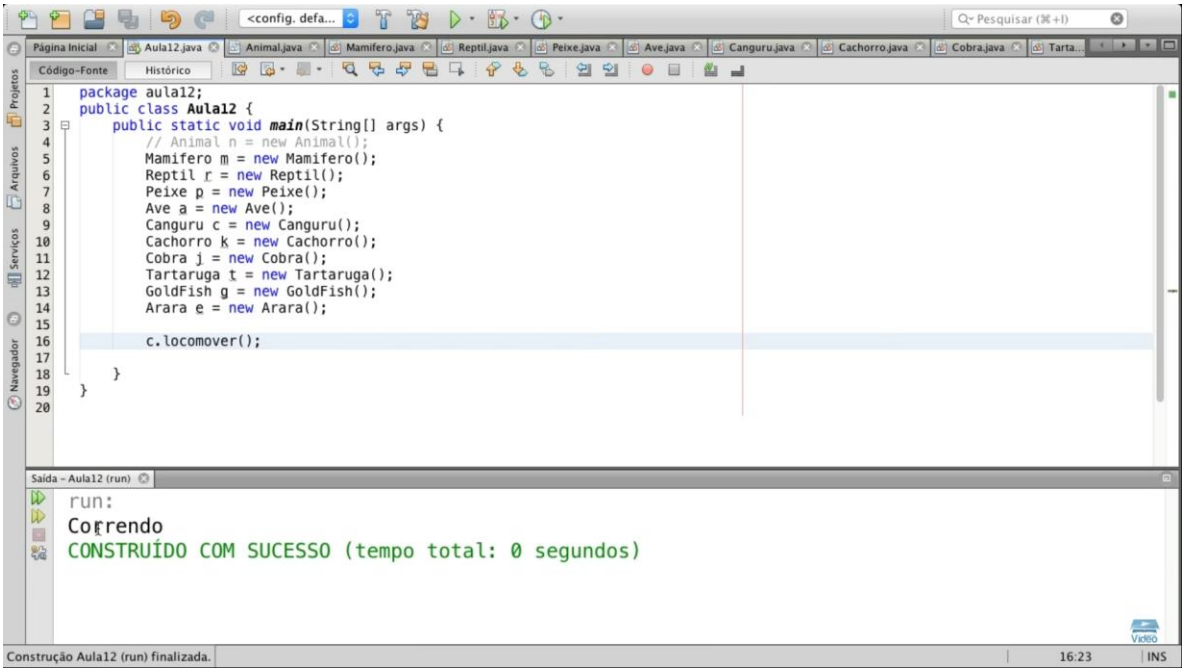


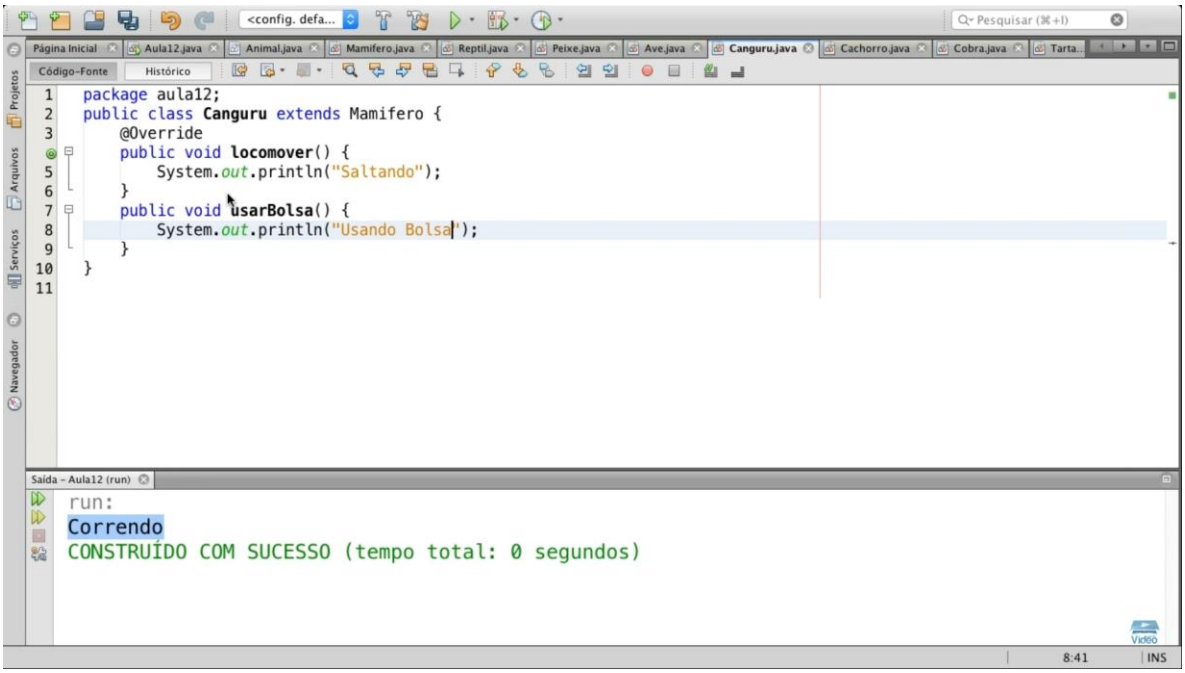

Curso POO Java #12b - Polimorfismo em Java (Parte 1)
package aula12;
public class Cachorro extends Mamifero {
}
Saída – Aula12 (run)
run:
Voando
Nadando
CONSTRUÍDO COM SUCESSO (tempo total: 0 segundos)
MAIS VÍDEOS
11:52 / 18:16
YouTube

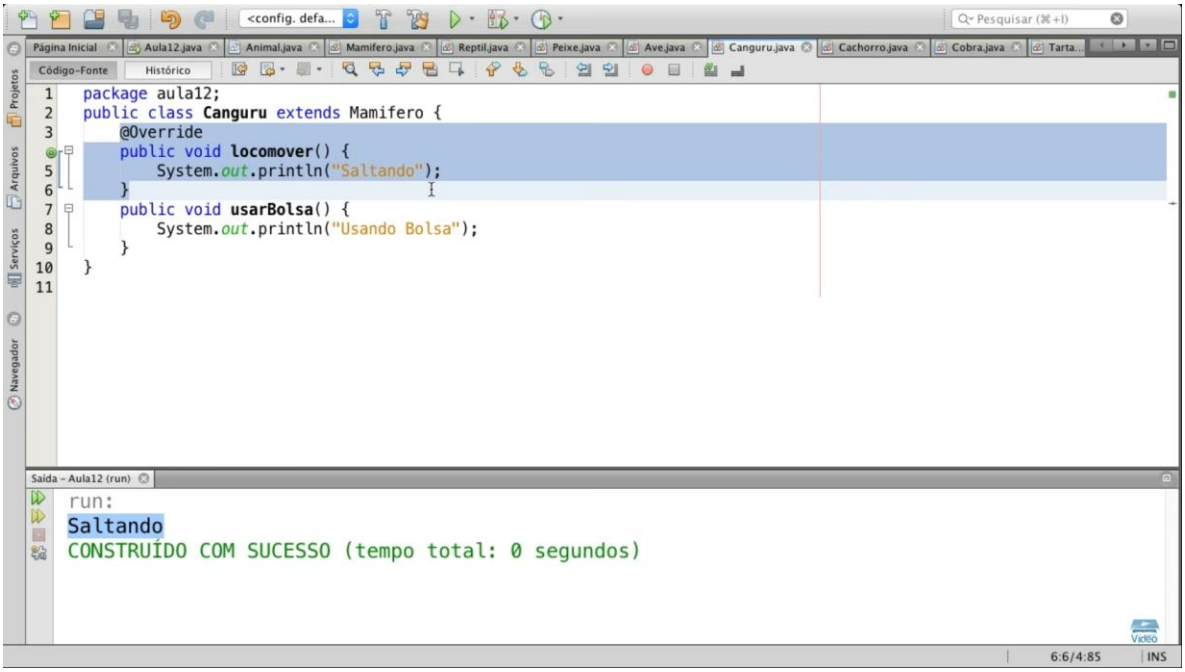

Curso POO Java #12b - Polimorfismo em Java (Parte 1)
Pressione Esc para sair do modo tela cheia
package aula12;
public class Cobra extends Reptil {
}
Saída – Aula12 (run)
run:
Voando
Nadando
CONSTRUÍDO COM SUCESSO (tempo total: 0 segundos)
MAIS VÍDEOS
11:56 / 18:16
YouTube

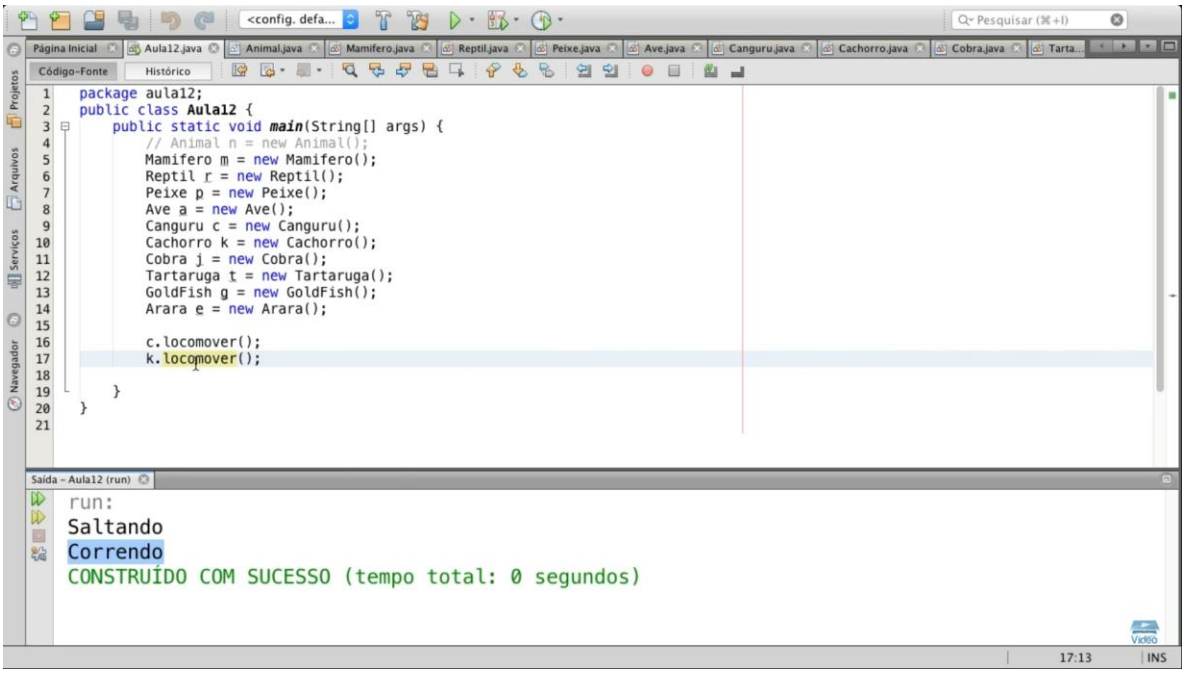

Curso POO Java #12b - Polimorfismo em Java (Parte 1)
package aula12;
public class Tartaruga extends Reptil {
}
Saída – Aula12 (run)
run:
Voando
Nadando
CONSTRUÍDO COM SUCESSO (tempo total: 0 segundos)
MAIS VÍDEOS
11:58 / 18:16

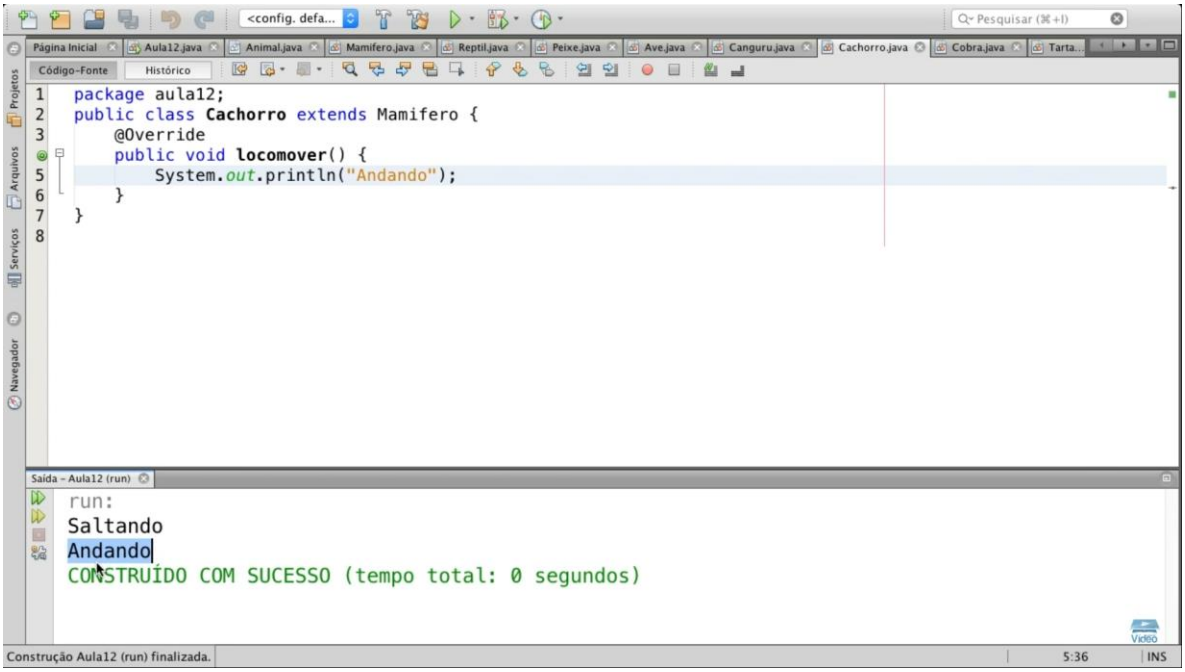

Curso POO Java #12b - Polimorfismo em Java (Parte 1)
package aula12;
public class Arara extends Ave {
}
Saída – Aula12 (run)
run:
Voando
Nadando
CONSTRUÍDO COM SUCESSO (tempo total: 0 segundos)
MAIS VÍDEOS
12:05 / 18:16

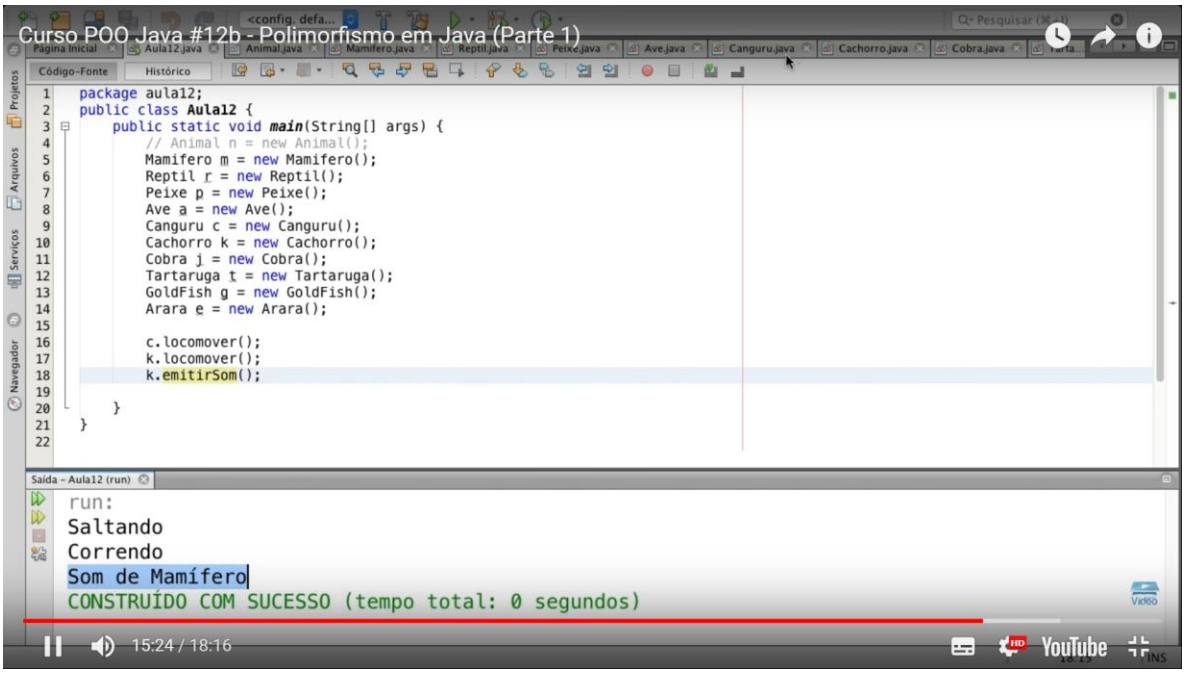

Página Inicial
Aula12.java
Animal.java
Mamifero.java
Reptil.java
Peixe.java
Ave.java
Canguru.java
Cachorro.java
Cobra.java
Tarta...
Código-Fonte
Histórico
1 package aula12;
2 public class Aula12 {
3 public static void main(String[] args) {
4 // Animal n = new Animal();
5 Mamifero m = new Mamifero();
6 Reptil r = new Reptil();
7 Peixe p = new Peixe();
8 Ave a = new Ave();
9 Canguru c = new Canguru();
10 Cachorro k = new Cachorro();
11 Cobra j = new Cobra();
12 Tartaruga t = new Tartaruga();
13 GoldFish g = new GoldFish();
14 Arara e = new Arara();
15
16 c.locomover();
17
18 }
19 }
20
Saída – Aula12 (run)
run:
Correndo
CONSTRUÍDO COM SUCESSO (tempo total: 0 segundos)
Construção Aula12 (run) finalizada.
16:23
INS

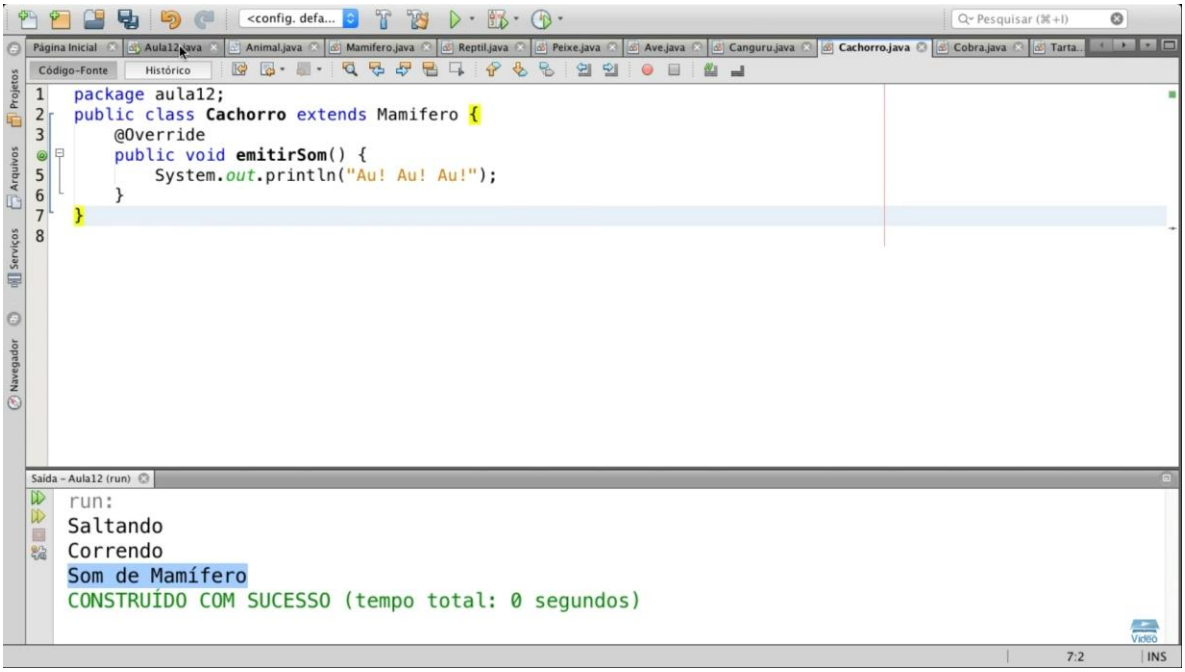

Página Inicial
Aula12.java
Animal.java
Mamifero.java
Reptil.java
Peixe.java
Ave.java
Canguru.java
Cachorro.java
Cobra.java
Tarta...
Código-Fonte
Histórico
1 package aula12;
2 public class Canguru extends Mamifero {
3 @Override
5 public void locomover() {
6 System.out.println("Saltando");
7 }
8 public void usarBolsa() {
9 System.out.println("Usando Bolsa");
10 }
11 }
Saída – Aula12 (run)
run:
Correndo
CONSTRUÍDO COM SUCESSO (tempo total: 0 segundos)
8:41
INS

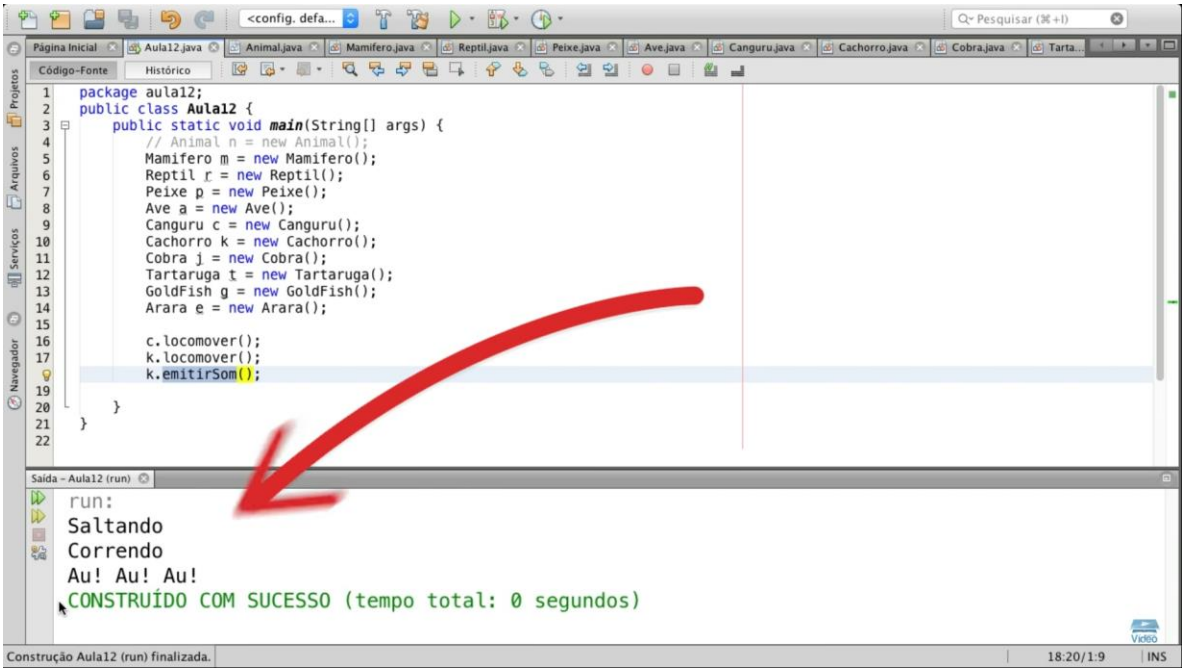

Página Inicial
Aula12.java
Animal.java
Mamifero.java
Reptil.java
Peixe.java
Ave.java
Canguru.java
Cachorro.java
Cobra.java
Tarta...
Código-Fonte
Histórico
1 package aula12;
2 public class Canguru extends Mamifero {
3 @Override
4 public void locomover() {
5 System.out.println("Saltando");
6 }
7 public void usarBolsa() {
8 System.out.println("Usando Bolsa");
9 }
10 }
11
Saída – Aula12 (run)
run:
Saltando
CONSTRUÍDO COM SUCESSO (tempo total: 0 segundos)
6:6/4:85
INS

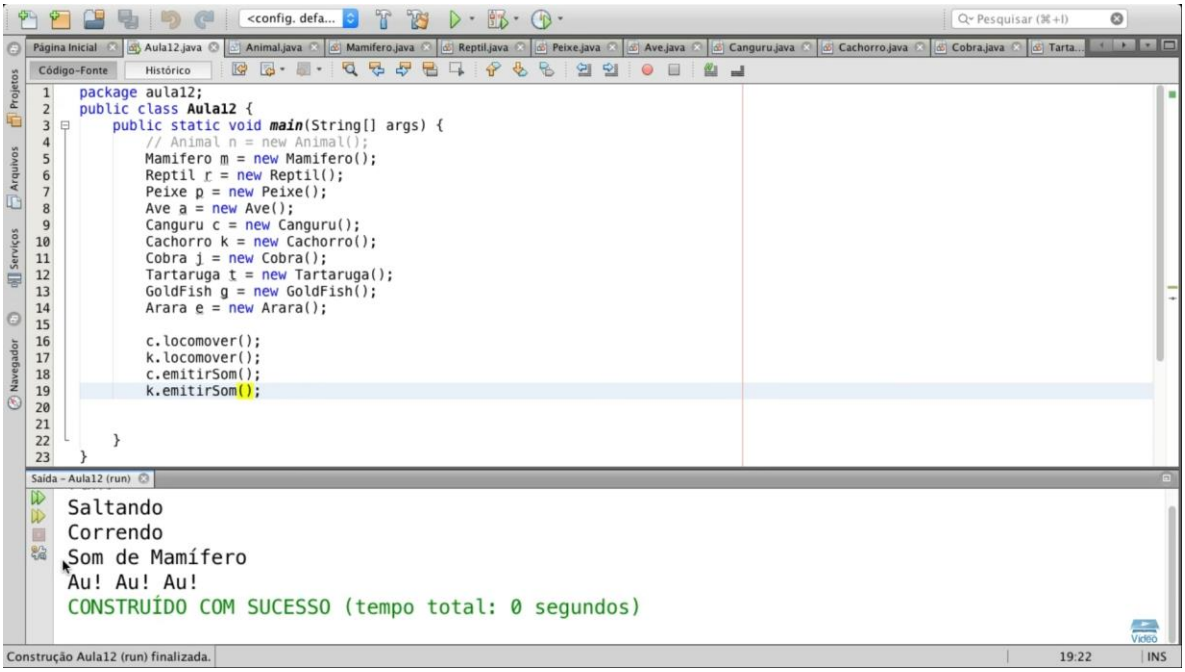

Página Inicial
Aula12.java
Animal.java
Mamifero.java
Reptil.java
Peixe.java
Ave.java
Canguru.java
Cachorro.java
Cobra.java
Tarta...
Código-Fonte
Histórico
1 package aula12;
2 public class Aula12 {
3 public static void main(String[] args) {
4 // Animal n = new Animal();
5 Mamifero m = new Mamifero();
6 Reptil r = new Reptil();
7 Peixe p = new Peixe();
8 Ave a = new Ave();
9 Canguru c = new Canguru();
10 Cachorro k = new Cachorro();
11 Cobra j = new Cobra();
12 Tartaruga t = new Tartaruga();
13 GoldFish g = new GoldFish();
14 Arara e = new Arara();
15
16 c.locomover();
17 k.locomover();
18
19 }
20 }
21
Saída – Aula12 (run)
run:
Saltando
Correndo
CONSTRUÍDO COM SUCESSO (tempo total: 0 segundos)
17:13
INS


Página Inicial
Aula12.java
Animal.java
Mamifero.java
Reptil.java
Peixe.java
Ave.java
Canguru.java
Cachorro.java
Cobra.java
Código-Fonte
Histórico
1 package aula12;
2 public class Cachorro extends Mamifero {
3 @Override
4 public void locomover() {
5 System.out.println("Andando");
6 }
7 }
8
Saída – Aula12 (run)
run:
Saltando
Andando
CONSTRUÍDO COM SUCESSO (tempo total: 0 segundos)
Construção Aula12 (run) finalizada.
5:36
INS


Curso POO Java #12b - Polimorfismo em Java (Parte 1)
Página Inicial
Aula12.java
Animal.java
Mamifero.java
Reptil.java
Peixe.java
Ave.java
Canguru.java
Cachorro.java
Cobra.java
Código-Fonte
Histórico
1 package aula12;
2 public class Aula12 {
3 public static void main(String[] args) {
4 // Animal n = new Animal();
5 Mamifero m = new Mamifero();
6 Reptil r = new Reptil();
7 Peixe p = new Peixe();
8 Ave a = new Ave();
9 Canguru c = new Canguru();
10 Cachorro k = new Cachorro();
11 Cobra j = new Cobra();
12 Tartaruga t = new Tartaruga();
13 GoldFish g = new GoldFish();
14 Arara e = new Arara();
15
16 c.locomover();
17 k.locomover();
18 k.emitirSom();
19
20 }
21 }
22
Saída – Aula12 (run)
run:
Saltando
Correndo
Som de Mamífero
CONSTRUÍDO COM SUCESSO (tempo total: 0 segundos)
15:24 / 18:16
YouTube
INS


Página Inicial
Aula12.java
Animal.java
Mamifero.java
Reptil.java
Peixe.java
Ave.java
Canguru.java
Cachorro.java
Cobra.java
Tarta...
Código-Fonte
Histórico
1 package aula12;
2 public class Cachorro extends Mamifero {
3 @Override
4 public void emitirSom() {
5 System.out.println("Au! Au! Au!");
6 }
7 }
8
Saída – Aula12 (run)
run:
Saltando
Correndo
Som de Mamífero
CONSTRUÍDO COM SUCESSO (tempo total: 0 segundos)
7:2
INS


Página Inicial
Aula12.java
Animal.java
Mamifero.java
Reptil.java
Peixe.java
Ave.java
Canguru.java
Cachorro.java
Cobra.java
Tarta...
Código-Fonte
Histórico
1 package aula12;
2 public class Aula12 {
3 public static void main(String[] args) {
4 // Animal n = new Animal();
5 Mamifero m = new Mamifero();
6 Reptil r = new Reptil();
7 Peixe p = new Peixe();
8 Ave a = new Ave();
9 Canguru c = new Canguru();
10 Cachorro k = new Cachorro();
11 Cobra j = new Cobra();
12 Tartaruga t = new Tartaruga();
13 GoldFish g = new GoldFish();
14 Arara e = new Arara();
15
16 c.locomover();
17 k.locomover();
18 k.emitirSom();
19
20 }
21 }
22
Saída – Aula12 (run)
run:
Saltando
Correndo
Au! Au! Au!
CONSTRUÍDO COM SUCESSO (tempo total: 0 segundos)
Construção Aula12 (run) finalizada.
18:20/1:9
INS


Página Inicial
Aula12.java
Animal.java
Mamifero.java
Reptil.java
Peixe.java
Ave.java
Canguru.java
Cachorro.java
Cobra.java
Tarta...
Código-Fonte
Histórico
Projetos
Arquivos
Serviços
Navegador
1 package aula12;
2 public class Aula12 {
3 public static void main(String[] args) {
4 // Animal n = new Animal();
5 Mamifero m = new Mamifero();
6 Reptil r = new Reptil();
7 Peixe p = new Peixe();
8 Ave a = new Ave();
9 Canguru c = new Canguru();
10 Cachorro k = new Cachorro();
11 Cobra j = new Cobra();
12 Tartaruga t = new Tartaruga();
13 GoldFish g = new GoldFish();
14 Arara e = new Arara();
15
16 c.locomover();
17 k.locomover();
18 c.emitirSom();
19 k.emitirSom();
20
21
22 }
23 }
Saída – Aula12 (run)
Saltando
Correndo
Som de Mamífero
Au! Au! Au!
CONSTRUÍDO COM SUCESSO (tempo total: 0 segundos)
Construção Aula12 (run) finalizada.
19:22
INS