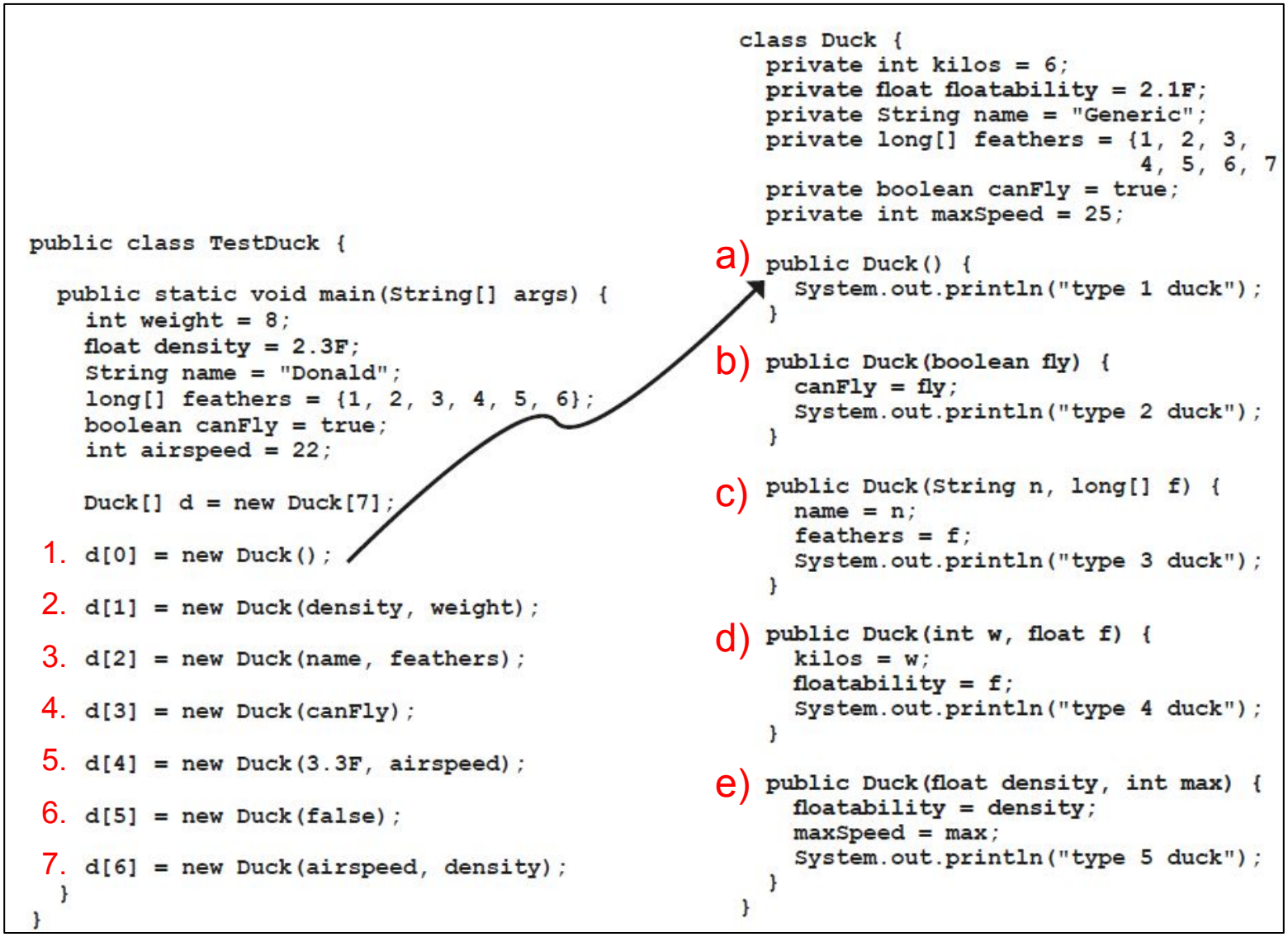
Exercícios

Tente resolver manualmente e, em seguida, programe o código para descobrir as respostas

Exercício: associe as chamadas dos construtores da classe Duck na esquerda à implementação correta da direita conforme o exemplo.



```
public class TestDuck {

  public static void main(String[] args) {
    int weight = 8;
    float density = 2.3F;
    String name = "Donald";
    long[] feathers = {1, 2, 3, 4, 5, 6};
    boolean canFly = true;
    int airspeed = 22;

    Duck[] d = new Duck[7];

 1. d[0] = new Duck();

 2. d[1] = new Duck(density, weight);

 3. d[2] = new Duck(name, feathers);

 4. d[3] = new Duck(canFly);

 5. d[4] = new Duck(3.3F, airspeed);

 6. d[5] = new Duck(false);

 7. d[6] = new Duck(airspeed, density);
  }
}
```

```
class Duck {
  private int kilos = 6;
  private float floatability = 2.1F;
  private String name = "Generic";
  private long[] feathers = {1, 2, 3,
                             4, 5, 6, 7
  private boolean canFly = true;
  private int maxSpeed = 25;

a) public Duck() {
     System.out.println("type 1 duck");
   }

b) public Duck(boolean fly) {
     canFly = fly;
     System.out.println("type 2 duck");
   }

c) public Duck(String n, long[] f) {
     name = n;
     feathers = f;
     System.out.println("type 3 duck");
   }

d) public Duck(int w, float f) {
     kilos = w;
     floatability = f;
     System.out.println("type 4 duck");
   }

e) public Duck(float density, int max) {
     floatability = density;
     maxSpeed = max;
     System.out.println("type 5 duck");
   }
}
```

Exercício: Dado o código da esquerda, o que é impresso na tela após a execução da classe TestHippo: A ou B?

```
public class Animal {
   public Animal() {
      System.out.println("Making an Animal");
   }
}
```

```
public class Hippo extends Animal {
  public Hippo() {
    System.out.println("Making a Hippo");
  }
}
```

```
public class TestHippo {
  public static void main(String[] args) {
    System.out.println("Starting...");
    Hippo h = new Hippo();
  }
}
```

A

```
File Edit Window Help Swear
% java TestHippo
Starting...
Making an Animal
Making a Hippo
```

B

```
File Edit Window Help Swear
% java TestHippo
Starting...
Making a Hippo
Making an Animal
```

Exercicio: Identifique os construtores inválidos abaixo. Associe os erros de compilação da esquerda com o construtor que o causou na direita.

```
public class Boo {
  public Boo(int i) { }
  public Boo(String s) { }
  public Boo(String s, int i) { }
}
```

```
class SonOfBoo extends Boo {
  public SonOfBoo() {
    super("boo");
  }

1. public SonOfBoo(int i) {
     super("Fred");
   }

2. public SonOfBoo(String s) {
     super(42);
   }

3. public SonOfBoo(int i, String s) {
   }

4. public SonOfBoo(String a, String b, String c) {
     super(a, b);
   }

5. public SonOfBoo(int i, int j) {
     super("man", j);
   }

6. public SonOfBoo(int i, int x, int y) {
     super(i, "star");
   }
}
```

a)

```
File Edit Window Help
%javac SonOfBoo.java
cannot resolve symbol
symbol : constructor Boo
(java.lang.String,java.
lang.String)
```

b)

```
File Edit Window Help Yadayadayada
%javac SonOfBoo.java
cannot resolve symbol
symbol  : constructor Boo
(int,java.lang.String)
```

c)

```
File Edit Window Help ImNotListening
%javac SonOfBoo.java
cannot resolve symbol
symbol:constructor Boo()
```

Exercício: Quais das linhas de código à direita, se adicionadas à classe à esquerda no ponto A, fariam com que exatamente um objeto adicional ficasse elegível para o Garbage Collector?

```
public class GC {
  public static GC doStuff() {
    GC newGC = new GC();
    doStuff2(newGC);
    return newGC;
  }

  public static void main(String[] args) {
    GC gc1;
    GC gc2 = new GC();
    GC gc3 = new GC();
    GC gc4 = gc3;
    gc1 = doStuff();

    (A)

    // call more methods
  }

  public static void doStuff2(GC copyGC) {
    GC localGC = copyGC;
  }
}
```

```
1   copyGC = null;
2   gc2 = null;
3   newGC = gc3;
4   gc1 = null;
5   newGC = null;
6   gc4 = null;
7   gc3 = gc2;
8   gc1 = gc4;
9   gc3 = null;
```

Exercício: Qual objeto tem mais variáveis fazendo referência a ele: Bees, Raccon, Kit ou Bear? Explique sua resposta.

```
class Bees {
  Honey[] beeHoney;
}

class Raccoon {
  Kit rk;
  Honey rh;
}

class Kit {
  Honey honey;
}

class Bear {
  Honey hunny;
}
```

```
public class Honey {
  public static void main(String[] args) {
    Honey honeyPot = new Honey();
    Honey[] ha = {honeyPot, honeyPot, honeyPot, honeyPot};
    Bees bees = new Bees();
    bees.beeHoney = ha;
    Bear[] bears = new Bear[5];
    for (int i = 0; i < 5; i++) {
      bears[i] = new Bear();
      bears[i].hunny = honeyPot;
    }
    Kit kit = new Kit();
    kit.honey = honeyPot;
    Raccoon raccoon = new Raccoon();

    raccoon.rh = honeyPot;
    raccoon.rk = kit;
    kit = null;
  } //  end of main
}
```

Exercício: Qual(quais) trecho(s) de código abaixo não apresentaria erro de compilação?

①
```
public class Foo {
  static int x;

  public void go() {
    System.out.println(x);
  }
}
```

②
```
public class Foo2 {
  int x;

  public static void go() {
    System.out.println(x);
  }
}
```

③
```
public class Foo3 {
  final int x;

  public void go() {
    System.out.println(x);
  }
}
```

④
```
public class Foo4 {
  static final int x = 12;

  public void go() {
    System.out.println(x);
  }
}
```

⑤
```
public class Foo5 {
  static final int x = 12;

  public void go(final int x) {
    System.out.println(x);
  }
}
```

⑥
```
public class Foo6 {
  int x = 12;

  public static void go(final int x) {
    System.out.println(x);
  }
}
```

Exercise: Determine se o código abaixo irá compilar. Caso não compile, como você o corrigiria? Quando executado, qual seria sua saída: A ou B?

```
class StaticSuper {
  static {
    System.out.println("super static block");
  }

  StaticSuper () {
    System.out.println("super constructor");
  }
}

public class StaticTests extends StaticSuper {
  static int rand;

  static {
    rand = (int) (Math.random() * 6);
    System.out.println("static block " + rand);
  }

  StaticTests() {
    System.out.println("constructor");
  }

  public static void main(String[] args) {
    System.out.println("in main");
    StaticTests st = new StaticTests();
  }
}
```

A.

```
File  Edit  Window  Help  Cling
%java StaticTests
static block 4
in main
super static block
super constructor
constructor
```

B.

```
File  Edit  Window  Help  Electricity
%java StaticTests
super static block
static block 3
in main
super constructor
constructor
```

Exercício: Determine se as afirmações abaixo são verdadeiras ou falsas.

1. Para usar a classe Math, o primeiro passo é criar uma instância dela.
2. Você pode marcar um construtor com a palavra-chave **static**.
3. Métodos estáticos não têm acesso ao estado de variáveis de instância do objeto "this".
4. É uma boa prática chamar um método estático usando uma variável de referência.
5. Variáveis estáticas podem ser usadas para contar as instâncias de uma classe.
6. Construtores são chamados antes que as variáveis estáticas sejam inicializadas.
7. **MAX_SIZE** seria um bom nome para uma variável estática final.
8. Um bloco inicializador estático é executado antes que o construtor de uma classe seja executado.
9. Se uma classe for marcada como final, todos os seus métodos também devem ser marcados como final.
10. Um método final pode ser sobrescrito apenas se sua classe for estendida.