Equals e Vector

com a criação da classe Data, é possivel notar que ao tentar comparar dois objetos do tipo Data, o resultado será falso, pois o método equals() da classe Object compara as referências dos objetos, e não o conteúdo dos objetos.

```
public class Data {
  private int dia;
  private int mes;
  private int ano;

  public Data(int dia, int mes, int ano) {
    this.dia = dia;
    this.mes = mes;
    this.ano = ano;
  }
}

public class Teste {
  public static void main(String[] args) {
    Data d1 = new Data(1, 1, 2017);
    Data d2 = new Data(1, 1, 2017);

    System.out.println(d1.equals(d2)); // false
  }
}
```

Para resolver esse problema, precisamos sobrescrever o método equals() na classe Data.

```
public class Data {
  private int dia;
  private int mes;
  private int ano;

  public Data(int dia, int mes, int ano) {
    this.dia = dia;
    this.mes = mes;
    this.ano = ano;
  }

  @Override
  public boolean equals(Object obj) {
    if (obj == null) {
      return false;
    }

    if (obj == this) {
      return true;
    }

    if (!(obj instanceof Data)) {
      return false;
```

```
    }

    Data d = (Data) obj;

    return this.dia == d.dia && this.mes == d.mes && this.ano == d.ano;
  }
}

public class Teste {
  public static void main(String[] args) {
    Data d1 = new Data(1, 1, 2017);
    Data d2 = new Data(1, 1, 2017);

    System.out.println(d1.equals(d2)); // true
  }
}
```

Mas o que o vector faz com isso?

```
import java.util.Vector;

public class Programa {
  public static void main(String[] args) {
    try {
      Data d1 = new Data(19, 1, 1966);
      Data d2 = d1;
      Data d3 = new Data(19, 1, 1966);
      Data d4 = new Data(29, 6, 1992);


      Vector<Data> v = new Vector<Data>();
      v.add(d1);

      if (vec.contains(d3))
    }
  }
}
```