Nome: Vinícius Andrade Frederico RA: 139223 Turma: A

Obs: O arquivo lab06.zip contém o código por completo. Abaixo estão somente os códigos necessários para o relatório.

Item 1)

Ao usar o modificador abstract, não é possível instanciar um objeto da classe abstrata, pois ela não define um objeto, simplesmente serve como um modelo.

Item 2)

Não funciona pois como Veiculo é uma classe abstrata, e Caminhao e Carro são classes que herdam Veiculo, elas devem ter todos os métodos de Veiculo.

Item 3)

Pelo mesmo motivo do Item 1, não é possível instanciar um objeto da classe Caminhao ou Carro.

Item 4)

Ao redefinir o método ligar como abaixo (Carro/Caminho), o código funciona corretamente.

```
public boolean ligar(){

		System.out.println("O Carro foi ligado corretamente");
		return true;
	}
public boolean ligar(){

		System.out.println("O Caminhao foi ligado corretamente");
		return true;
	}
```

Item 5)

Função ligarTodos:

```
	public void ligarTodos() {
		for (int i = 0; i < ultimaPosicao; i++) {
			veiculos[i].ligar();
		}
	}
```

Adicionado em Inicial.java:

```
	fila.ligarTodos();
```

Item 6)

Inicial.java:

```
public static void main(String[] args) {
        Fila<Veiculo> fila = new Fila<Veiculo>();
        System.out.println("A Fila de Veiculos contem:");
        fila.adiciona(new Carro(5, 2, 1999, "VOLKSWAGEN", "GOL", "EST3245"));
        fila.adiciona(new Caminhao(15000, 3, 1997, "VOLKSWAGEN", "TITAN", "KED9871"));
        fila.adiciona(new Carro(5, 1, 1998, "FIAT", "PALIO", "JKU2171"));
        fila.adiciona(new Carro(5, 5, 2001, "FORD", "FIESTA", "JNM2464"));
        fila.adiciona(new Caminhao(10000, 2, 2000, "FORD", "CARGO", "KMG4171"));
        fila.adiciona(new Carro(4, 2, 2001, "CHEVROLET", "CELTA", "JGH5432"));
        fila.adiciona(new Caminhao(8000, 4, 1996, "FORD", "SCANIA", "DEY6429"));
        if (!fila.Vazia()) {
                fila.mostraFila();
        //      fila.ligarTodos();
        } else {
                System.out.println("Fila está vazia");
        }
}
```

Saída:

```
A Fila de Veiculos contem:
br.unicamp.ic.mc302.veiculos.Carro@21c3dc66
br.unicamp.ic.mc302.veiculos.Caminhao@41babddb
br.unicamp.ic.mc302.veiculos.Carro@4b069693
br.unicamp.ic.mc302.veiculos.Carro@1d87b360
br.unicamp.ic.mc302.veiculos.Caminhao@381172c5
br.unicamp.ic.mc302.veiculos.Carro@1860045
br.unicamp.ic.mc302.veiculos.Caminhao@47bb2cb
```

O código define uma Fila de Veiculo. Ao imprimir um objeto, ele chama o método toString. Como o método não foi sobrescrito, ele simplesmente imprime a classe e o endereço do objeto.

Métodos toString() de Carro e Caminhao:

```
public String toString(){
        return "\nTipo-->Carro" + "\nLotação = "+lotacao+ "\nNumero de Portas = "+numPortas + "\n";
 }
public String toString(){
        return "\nTipo--->Caminhao" + "\nCapacidade = " + capacidade + "\nNumero de Eixos = " + numDeEixos;
}
```

Saída:

A Fila de Veiculos contem:

Tipo-->Carro
Lotação = 5
Numero de Portas = 2

Tipo--->Caminhao
Capacidade = 15000
Numero de Eixos = 3

Tipo-->Carro
Lotação = 5
Numero de Portas = 1

Tipo-->Carro
Lotação = 5
Numero de Portas = 5

Tipo--->Caminhao
Capacidade = 10000
Numero de Eixos = 2

Tipo-->Carro
Lotação = 4
Numero de Portas = 2

Tipo--->Caminhao
Capacidade = 8000
Numero de Eixos = 4

Item 7)

Passando a definição de placa de Veiculo, e de seu construtor, para Caminho e Carro e para seus construtores, e removendo de Aviao, resolvemos isso.

Item 8)

O objeto Circulo, possui na interface pública, mostrar, esconder, e mover além das provenientes da classe base Object. Sendo que mover é definido em Figura e esconder e mostrar é abstrato em Figura e redefinido em Circulo.

Item 9)

Ao fazer os métodos mostrar() e esconder() de Figura deixarem de ser abstratos e imprimirem a mensagem, nada mudou no resultado pois os métodos mostrar() e esconder() de Circulo e Linha sobrescreviam eles. Ao comentar as redefinições de mostrar() em Circulo e esconder() em Linha, ao serem chamados, é chamado o de Figura.

Item 10)

Código no arquivo zipado.