```
import java.util.ArrayList;

// todos os erros definidos no trabalho
enum AFD_ERRORS
{
    ERRO_01, // uso de simbolos que nao fazem parte do alfabeto de transica
    ERRO_02, // uso de simbolos que nao fazem parte do alfabeto nas palavra
    ERRO_03, // uso de estados que nao fazem parte do conjunto de estados n
transicoes
    ERRO_04, // uso de automato cujas transicoes nao determinam um AFD
};

public class AFD {
    // estado atual do AFD
    Estado estadoAtual;
    // lista de estados definidos
    ArrayList<Estado> estados;
    // alfabeto definido
    ArrayList<Character> alfabeto;

    // ids de todos os estados definidos
    private ArrayList<Character> estadosDefinidos;

    AFD(){
        this.estadoAtual = null;
        this.alfabeto = new ArrayList<>();
        this.estados = new ArrayList<>();
        this.estadosDefinidos = new ArrayList<>();
    }

    // funcao que define o alfabeto por parametros
    public void definirAlfabeto(char... tokens){
        for(char token : tokens){
            alfabeto.add(token);
        }
    }

    // adiciona um token ao alfabeto
    public void adicionarTokenAlfabeto(char token){
        alfabeto.add(token);
    }

    // adiciona um estado ao AFD
    public void adicionarEstado(Estado estado){
        if(estadoAtual == null)
            estadoAtual = estado;
        estados.add(estado);
        // definindo o conjunto de estados
        estadosDefinidos.add(estado.ID);
    }

    // retorna um estado por meio de um ID ou nulo
    public Estado estadoPorId(char ID){
        for(Estado estado : estados){
```

```
            if(estado.ID == ID)
                return estado;
        }
        return null;
    }

    // testa uma palavra e verifica se a mesma e aceita pelo AFD
    public boolean teste(String s){
        if(!checaErros(s)) System.exit(1);
        // resetando meu grandioso AFD sagrado
        estadoAtual = estados.get(0);
        for(char token : s.toCharArray()){
            if(estadoAtual.acharTransicao(token)) {
                estadoAtual = estadoAtual.fazerTransicao(token);
            } else {
                return msgErro(String.format("Erro %s, transicao nao defini
estado",AFD_ERRORS.ERRO_04, estadoAtual.ID));
            }
        }

        if(estadoAtual.estadoFinal)
            System.out.println(s + " sim");
        else
            System.out.println(s + " nao");

        return estadoAtual.estadoFinal;
    }

    // executa varios testes por meio de uma lista
    public void testes(String[] palavras){
        assert(palavras != null) : "Lista de palavras para teste nulas";
        for(String palavra : palavras)
            teste(palavra);
    }

    // printa uma mensagem e retorna false sempre
    private boolean msgErro(String msg){
        System.out.println(msg);
        return false;
    }

    // faz a verificacao de erros no automato em toda chamada
    // do metodo teste
    private boolean checaErros(String s){
        boolean sucesso = true;
        int numTransicoes = 0;

        for(Estado estado : estados){
            for(Transicao transicao : estado.transicoes){
                // caso exista algum token que nao exista no alfabeto de
transicoes
                if(!alfabeto.contains(transicao.token))
                    sucesso = msgErro(String.format("Erro %s no estado %c,
transicao com sibolo que nao faz parte do alfabeto",AFD_ERRORS.ERRO_01,
estado.ID));
```

```
                // caso exista uma transicao espontanea
                if(transicao.token.compareTo('$') == 0)
                    sucesso = msgErro(String.format("Erro %s no estado %c,
transicao vazia",AFD_ERRORS.ERRO_04, estado.ID));
                // incrementando o numeros de transicoes
                numTransicoes++;
            }
        }

        // pela definicao o numero de transiçoes de um AFD e
        // sempre igual ao numero de estados multiplicado
        // pelo numero de elementos do alfabeto
        if((estados.size() * alfabeto.size()) != numTransicoes)
            sucesso = msgErro(String.format("Erro %s numero de transicoes n
automato", AFD_ERRORS.ERRO_04));

        // erro 02
        for(char token : s.toCharArray()){
            if(!alfabeto.contains(token)){
                sucesso = msgErro(String.format("Erro %s, simbolo nao faz p
do alfabeto",AFD_ERRORS.ERRO_02));
            }
        }

        return sucesso;
    }

}
```