Skill Builder 4 - Designing and Implementing Finite State Machines

At the end of this Skill Builder, a studnet will be able to: - Design an FSMs - Apply process to convert an FSM to code - Implement an FSM in a function

Designing and Implementing an FSM

Design a finite-state machine that determines whether a string contains a sub-string that begins with a **t** or **T** and ends with a **y** or **Y**.
The sub-string may appear anywhere inside the string. For example,

```
Hello, my name is Tom Brady and I am thirsty.
```

contains two sub-string that start with a **t** and end in a **y**.

```
Hello, my name is Tom Brady and I am thirsty.
```

Also,

```
Jasmine was rather terribly insulting!
```

contains a sub-string that starts with a **t** in the word **rather** and ends in a **y** in the word **terribly**.

```
Jasmine was rather terribly insulting!
```

Use software to draw the FSM. Any online solution will work as long as you can save it as a pdf (see draw.io). PowerPoint, Keynote, or LibreOffice may be the most convenient due to availability. Make sure to properly label the diagram including states, transitions, and actions. See instructions for submission of FSMs.

Click the link below to watch the video that shows how to draw an FSM in PowerPoint and

Keynote.

Drawing FSM

Implement Method

Implement the finite-state machine designed above in the SkillBuilder4 class method called

```
public static String findTYPattern(String s)
```

The function returns the first substring that starts with a **t** or **T** and ends with a **y** or **Y**; otherwise it returns an empty string. For example,

```
SkillBuilder4.findTYPattern("Hello, my name is Tom Brady and I am thirsty")
```

should return the string `Tom Brady`.

3. Submission of FSM and Implementation

The FSM should be in a pdf document called **fsm.pdf**. You are required to submit th **SkillBuilder4.java** and **fsm.pdf** files on CodeGrade.