CS 539-001 Natural Language Processing, Fall 2019
EX 1: Finite State Transducers

Instructions:

- Unlike HWs which are done in groups of up to 3, EXs should be done individually.
- Due Wednesday 10/2 at 11:59pm, on **Canvas**. No late submission will be accepted.
- Only `report.txt` (or `.pdf`), `pluralize2.fst`, and optionally `pluralize3.fst` should be submitted.
- Worth 5% of the final grade. Will be graded by completeness rather than correctness.
- The solutions will be posted on Canvas the day after it is due, which will help you for HW1.

This Exercise is designed to prepare you for HW1. You will be taking a simple FST that pluralizes English words, and modifying it to handle more cases. Please download `http://classes.engr.oregonstate.edu/eecs/fall2019/cs539-001/ex1/ex1.tgz` which contains:

- `carmel`: To start off, you will need to install the FST toolkit Carmel.[1] To simplify your job, we provided two executables, one for Linux64 and one for Mac OS X. Alternatively, you can use the OSU ENGR servers (such as `flip.engr.oregonstate.edu`), where it has been installed (just type `carmel`).
- `pluralize.fst`: The initial FST file that tries to add an **s** to the end of a word.
- `pluralize2.fst`: A slightly better FST that tries to handle **-es** but has a bug.

1 An FST for Pluralization

You can run pluralize.fst using:[2]

```
> echo "a p p l e" | carmel -sliOEQk 5 pluralize.fst

Input line 1: a p p l e
(12 states / 11 arcs reduce-> 7/6)
a p p l e s 1
```

And you should see that the FST added an **-s** to the end of "apple".

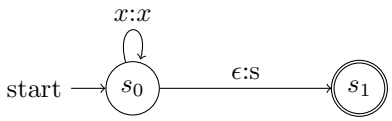

Figure 1: An FST that tries to add an **-s** to the end of a word. Note that x is a variable that matches any letter. That self arrow means that we read in any letter and output back that same letter. Check out the `pluralize.fst` file to see how this FST is defined in Carmel.

This is great but will also add an **-s** to words like "bus" that already have an **s** at the end.

[1] You can also download it for yourself from `http://www.isi.edu/licensed-sw/carmel/`, but it is rather challenging to compile it from source. So we strongly recommend you take our provided binaries.

[2] Don't worry about all of the crazy flags just yet. They are explained at the top of HW1.

```
> echo "b u s" | carmel -sliOEQk 5 pluralize.fst

Input line 1: b u s
(8 states / 7 arcs reduce-> 5/4)
b u s s 1
```

This FST turns "bus" → "buss". Which is not correct. We went ahead and provided another FST file **`pluralize2.fst`** that tries to add **-es** instead of **-s** to the end of bus.

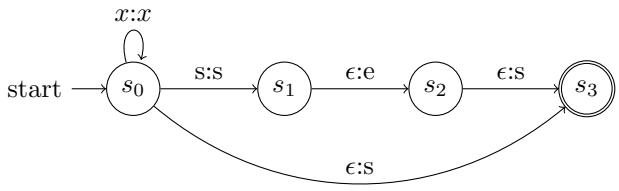

Figure 2: An FST that tries to add an **-es** or an **-s** to the end of a word.

This FST now almost does the right thing with "bus":

```
echo "b u s" | carmel -sliOEQk 5 pluralize2.fst

Input line 1: b u s
(10 states / 10 arcs reduce-> 7/7)
b u s s 1
b u s e s 1
```

How do you fix it so that it only outputs the correct "buses"? And what happens if you change the input to "sus"? Or "fuss"? Why? Figure out a way to fix this, so that `pluralize2.fst` can properly handle words like "bus", "bass", "sass", or "rise". Submit your modified `pluralize2.fst` file. Describe in your report the modifications made to the FST.

2 Optional: Other Pluralizations

Modify your FST again to handle some other pluralization rules.[3] Some examples might include

- "cherry" → "cherries"
- "leaf" → "leaves"
- "matrix" → "matrices"
- "automaton" → "automata"

Save this as a new file `pluralize3.fst` and submit that along with `pluralize2.fst`. Describe in your report what you tried, the examples that you handled, examples that you cannot handle, and how you modified your FST to accomplish this.

[3] Check out `https://en.wikipedia.org/wiki/English_plurals` for many examples.