Introduction
These Finite State Transducers (FSTs) normalize whole numbers 0 - 1000 in English and French. It was built with Pynini.

Methods (How your FST works)
For both languages, I created FSTs in this order:

- **Units FSTs**: 0 - 9, and 101 - 109 for numbers above 99 (hundreds)
- **Teens FSTs**: 11 - 19, 111 - 119, …, 911 - 919
- **Tens FSTs**: 10, 20, 30, … 90, 110, 120, …, 990
- **Hundreds FSTs**: 100, 200,... 900
- **Compound numbers FSTs**: 21-29, 31-31, …, 91 -99,121 - 129, … 991 - 999.
- A single FST for 1000

Findings:

- French was particularly challenging between compound numbers 71-79, 91-99, 171 - 179, 191 - 199, 271 - 279, 291 - 299, etc. This was because of the irregular compounds, like 71 as soixante-dix, 95 as quatre vingt quinze, etc

  The workaround for this was to create a special FST for numbers within 71 - 79, and 91 - 99 to fix errors like 71 translating to ‘soixante-dix un.`
- English FSTs were more straightforward than French rules , mainly due to the distinct rules for 71-79 and 91 - 99

Experiments:
I tested these FSTs on numbers such as 105, 115, 995, 850, 400, and 75.
I also tested on full sentences. The src/normalize.py script from the repo can be called to test these normalisers. The full description of the function is available on the GitHub repo.

Conclusion:
This FST works as expected with numbers between 0 and 1000 in both French and English.

The next steps include generating FSTs for numbers that include commas and full stops.