Hadar Levi 209006360
Rotem Assa 208277319

Information Extraction

Description of the code:

Building of the ontology:

To build the ontology, we used an OOP design pattern involving Country object, and Person object. The idea is that all the data we need of a country would be represented by one, compact and intuitive object.

Country object has 8 attributes: name, doc, president, prime minister, population, area, gov_form and capital. Most are self-explanatory, except for doc, which is the urlib object used to send the xpath queries to Wikipedia.

President and Prime minister are both instances of the Person object. That object has the following attributes: name, doc, country_names, date_of_birth, birth_loc. Again, most names are self-explanatory except for doc, which is the urlib object used to send the xpath queries to Wikipedia.

So, the basic structure of building the ontology is to iterate over the countries one by one, construct the appropriate Country object, and add that to the ontology.

We add the Country's values to the ontology by simply going over them and add a trio (using RDFlib) of URIs.

We did the same thing for the person's attributes.

Some attributes are nullable, hence we check if an attribute is null before adding that trio to the ontology.

Our question:

The question that we chose to add is "How many prime ministers were born in <country>?".

Here are some examples:

- How many prime ministers were born in Israel?

```
nova 7% python3 geo_qa.py question "How many prime ministers were born in Israel?"
1
```

- How many prime ministers were born in France?

```
nova 11% python3 geo_qa.py question "How many prime ministers were born in France?"
3
```

- How many prime ministers were born in Peru?

```
nova 16% python3 geo_qa.py question "How many prime ministers were born in Peru?"
1
```

Edge cases:

We had many edge cases when performing the xpath queries. Here are 3 examples:

1. When extracting the Area of a country, we queried the following brother of a tr element which has th/text()="Area" inside it. However, some countries like our

lovely Israel do not contain this, but rather have th/text()=”Area “ (with a space at the end). So, we created a joint query that returns both those items (line 44 in the file country.py).

2. Most countries had their president's name in td/a/@href, but a very few countries had their president in td/span/a[1]/@href. So, we used a solution like the one with Area (line 25 in the file country.py).

3. Most countries had their population in /tr[th/a/text() = "population"], but a very few countries had it in /tr[th/text() = "population"]. So, we used a solution like the one with Area (line 35 in the file country.py).