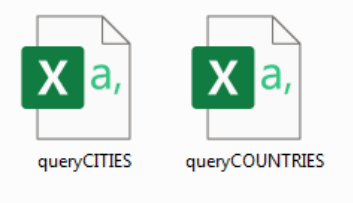
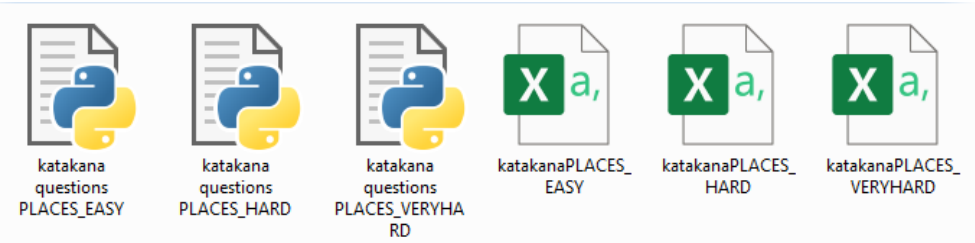
OUTPUT DESCRIPTION

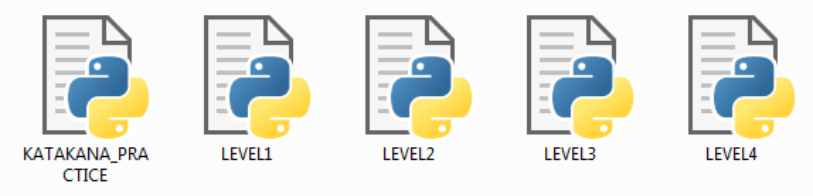
The Wikidata query returns as output the queryCITIES.csv and the queryCOUNTRIES.csv files



The three katakana-question-generation py files return as output the katanaPLACES csv file of the corresponding difficulty (the difference lies in the type of distractor)



The KATAKANA_PRACTICE py file, the main file, together with the LEVEL file it imports based on what the user selects after running the tool, **returns the output on the screen** (questions and feedback). Here I will show a sample.



The user is first given an encouraging message in Japanese, with Japanese emoji (Let's practice katakana!)

Afterwards, one common **use of the katakana script is explained** and used as justification for the choice of this theme as the focus of the katakana practice tool.

The user is asked to write whether they want to focus on cities, countries or both. **Regular expressions** are used to interpret the answer and select the **topic**. If the answer matches none of the patterns, the user is asked again.

```
カタカナを練習しましょう！ ✧ヾ(•ˬ•๑)


Katakana is often used to write the names of cities or countries outside Japan
What do you want to practice now, CITIES or COUNTRIES (or both)? just give me countries
Ok, so your practice topic will be:
COUNTRIES
```

```
Katakana is often used to write the names of cities or countries outside Japan
What do you want to practice now, CITIES or COUNTRIES (or both)? i would like countries and cities
Ok, so your practice topic will be:
BOTH CITIES AND COUNTRIES
```

The user is then given a description of the **levels** and has to chose one. If the answer given is not 1, 2, 3 or 4, the question is repeated. If the answer contains several numbers, the program selects the lowest one that matches one of the numbers (e.g. '6253' would return LEVEL 2).

```
I'll give you the name of a place in katakana and you have to select the English equivalent
But first, choose your level:
* LEVEL 1: You'll be given que transcription of the first character as a clue
* LEVEL 2: You also get a clue, but that character will appear somewhere in all the options
* LEVEL 3: You get no clues
* LEVEL 4: No clues, and all the options will be similar in Japanese
Type the number of the level you want: 1
You have chosen LEVEL 1
```

The first **question** appears, with or without a **clue** depending on the level, and with the **topic** previously selected by the user (cities/countries/both). Three **options** appear, and the user is asked to give and answer.

```
Here's a country in katakana: イスラエル Which one is it?
HINT: イ = i
*  Liberia
*  Chile
*  Israel
What's your answer?
(Copy or type, don't worry about accents of uppercase): isuraeru
✿ ビンゴ! You're correct, the answer is  Israel
```

The answers are checked based on **word similarity** (morphological similarity). Therefore, the user does not have to worry about capitalizing the word or typing accents or other marks outside the basic Latin alphabet of English. Also, one interesting feature is that the user can write the word **as it would be pronounced in Japanese**, and that is also processed correctly (as in the example of 'isuraeru', which is how the katakana transcription of 'Israel' would be pronounced).

When **the answer is correct**, the correct answer is repeated, together with positive words in English and Japanese ('bingo' and 'You're correct'), as well as a flower emoji (the closest thing to the hanamaru drawn on children's assignments when they are correct).

When **the answer is incorrect**, a disappointed emoji and the word for 'wrong' appear and the student is encouraged to keep practicing. The user is also shown an **automatic transcription of the distractors** to get some idea of why they were wrong. The transcription is done with the **jaconv library** in two **steps**: the Latin characters are transcribed in hiragana script, and from that to katakana (there is no direct function for Latin-to-katakana). The **transcription is not perfect**, but could be enough help, and using an automatic tool for this step simplifies the program a lot. Not all the automatic transcriptions are imperfect, only the ones where the English word has some letters representing sounds not existing in Japanese. For instance, syllables with "l" are transcribed as an "r" sound in Japanese. However, **jaconv** is a simple tool that cannot know the pronunciation of name places that, despite technically being in English, come from very diverse languages; therefore, it looks at the letters regardless of how they may sound and, when no simple transcription can be found, it returns the ツ symbol, which normally just emphasizes consonant sounds. Thus, as seen in the following example, Icelandic name places are difficult, but Spanish names are transcribed with fewer errors.

```
Here's a city in katakana: パルマ (モザンビーク) Which one is it?
HINT: パ = pa
*  Palma
*  Patreksfjörður
*  Paso de los Toros
What's your answer?
(Copy or type, don't worry about accents of uppercase): paso de los toros
(;´・`)> 違う Sorry, keep practicing!
Paso de los Toros would be  パソ デ ッオッ トロッ  (automatically transcribed)
Patreksfjörður would be  パッレッッッッöッðウッ  (automatically transcribed)
```

After each question, the user is asked **if they want another one**. If they type **'y'**, they get another question. If they type **'n'**, a goodbye and a waving emoji appear and the program ends. If **something else** is typed, a confused emoji appears and the program assumes that the user wants no more question and ends.

Yes:

```
Want more questions? Type y for YES or n for NO: y
Here's a city in katakana: リウォンデ Which one is it?
```

…

No:

```
Want more questions? Type y for YES or n for NO: n
またね！ ( ･ω･)ノ
```

Something else:

```
Want more questions? Type y for YES or n for NO: dunno
I'll assume you want no more questions 【・_・?】
またね！ ( ･ω･)ノ
```