In this assignment, you will solve problems about web scraping using Python. The recommended HTML parser for Python is *Beautiful Soup*[1].

Important! To get started, download the template file `Lab02.py`[2] and write your solution for each problem in its respective function. You can write your own supporting functions but **DO NOT** change the name of the solution functions (`problem_1`, `problem_2_1`, `problem_2_2` and `problem_2_3`) or the file name (`Lab02.py`).

Problem 1 (Infobox extraction). In wiki-based systems such as Wikipedia and Fandom, infoboxes usually store core information about the corresponding entities. For example, Figure 1 displays the infobox of *Tracy McConnell* (*"The Mother"*) from *How I Met Your Mother Wiki* on Fandom (`https://how-i-met-your-mother.fandom.com/wiki/Tracy_McConnell`). This infobox includes a list of attribute-value pairs describing the character, for e.g., the actress who played the character, the names of the episodes with her first and last appearances, her birthday, etc. Infoboxes have been valuable sources for knowledge bases like YAGO[3] because of their availability, high accuracy and the simplicity for extraction. Extracting knowledge from infoboxes is, therefore, a fundamental step of automated knowledge base construction.

Your task is to write a Python function that takes as input the name of a character in the TV Series *How I Met Your Mother*, and returns a list of all attribute-value pairs written in the infobox of the wiki page for the character on Fandom's *How I Met Your Mother Wiki* (`https://how-i-met-your-mother.fandom.com/`). For instance, given *"Tracy McConnell"* as the input, your function should return a list that looks like:

```
[
    {
        "attribute": "Portrayed by",
        "value": "Cristin Millioti"
    },
    {
        "attribute": "First appearance",
        "value": "Lucky Penny"
    },
    {
        "attribute": "Last appearance",
        "value": "Last Forever - Part Two"
    },
    {
        "attribute": "Full name",
        "value": "Tracy McConnell"
    },
    # truncated...
]
```

Note that, for attributes that come with multiple values (e.g., `"Also Known As"`, `"Occupation"` and `"Romances"` in Figure 1), you can either keep all the values in a long string as a big single value, or separate them and produce a list of strings such as:

```
{
    "attribute": "Romances",
    "value": [
        "Ted Mosby (Husband)",
        "Max (ex-boyfriend - deceased)",
        "Louis (ex-boyfriend)"
    ]
}
```

Figure 1: Tracy McConnell's infobox on Fandom.

You should also try to clean the values as much as possible (e.g., for the `"Born"` attribute, your output value should only be `"September 19, 1984"` and not include the reference `[1]`).

[1] `https://www.crummy.com/software/BeautifulSoup/bs4/doc/`
[2] `https://www.mpi-inf.mpg.de/fileadmin/inf/d5/teaching/ss22_akbc/Lab02.py`
[3] `https://yago-knowledge.org/`

Problem 2 (Web scraping). In this problem, you will scrape information of courses offered to Computer Science Masters students at Saarland University in the Summer Semester 2022 on LSF. For all tasks, you must use the *English* version of the LSF website.

2.1. Write a Python function to collect all courses and their corresponding URLs listed under the following site: `https://www.lsf.uni-saarland.de/qisserver/rds?state=wtree&search=1&trex=step&root120221=320944|310559|318658|311255&P.vx=kurz&noDBAction=y&init=y`
Your function should return a list of dictionaries that looks like:

```
[
    {
        "Name of Course": "Automated Knowledge Base Construction",
        "URL": "https://www.lsf.uni-saarland.de/qisserver/...&publishid=137193&..."
    },
    {
        "Name of Course": "Artificial Intelligence",
        "URL": "https://www.lsf.uni-saarland.de/qisserver/...&publishid=136388&..."
    }
    # truncated...
]
```

2.2. Write a Python function to scrape (i) all basic information which is written in the "Basic Information" table, and (ii) the names of all responsible instructors written in the "Responsible Instructors" table of a course given its URL. For example, for the *"Ethics for Nerds"* course[4], your function should return a dictionary of attribute-value pairs that looks like:

```
{
    "Type of Course": "Advanced lecture",
    # truncated...
    "Additional Links": [
        "https://dcms.cs.uni-saarland.de/ethics_22/"
    ],
    "Language": "english",
    "Responsible Instructors": [
        "Baum, Kevin , M.A. M.Sc.",
        "Hermanns, Holger , Univ.-Prof. Dr.-Ing.",
        "Sterz, Sarah"
    ]
}
```

Note that, there are exactly 2 fields that come with multiple values, `"Additional Links"` and `"Responsible Instructors"`, for which your code should produce a list of strings as value.

2.3. Use your code from Problem 2.2, write another Python function to collect the information of *all* courses that you get from Problem 2.1. Your function should write results to a comma-separated values (CSV) file[5] named `courses.csv`. The CSV file should contain the following fields: `"Name of Course"`, all fields from the "Basic Information" table, and `"Responsible Instructors"`. For fields with multiple values, apply `json.dumps`[6] to their values before writing to file. Directly writing a list as value to a CSV file is not recommended. Using `json.dumps` makes it easier when you read the CSV file afterwards as you will only need to apply its reverse method (i.e., `json.loads`) to reconstruct the Python list. (**Hint:** the CSV file should have exactly 16 fields)

Your submission should be a zip file which only contains `Lab02.py` and your supporting Python scripts (`*.py` files) if needed. Please submit all necessary files, which are compressed into a zip file named:

Lab02_MatriculationNumber_Name.zip

to the email address: **akbc-assignments@mpi-inf.mpg.de** with title of the email: **[AKBC]Lab02_MatriculationNumber_Name**

Deadline: 23:59 09.05.2022 (Monday)

[4] `https://www.lsf.uni-saarland.de/qisserver/rds?state=verpublish&status=init&vmfile=no&publishid=136353&moduleCall=webInfo&publishConfFile=webInfo&publishSubDir=veranstaltung`

[5] `https://docs.python.org/3/library/csv.html#csv.DictWriter`

[6] `https://docs.python.org/3/library/json.html#json.dumps`