Homework Assignment 1

Directory Services

Introduction

We store the employment information about the employees of our university and partner companies for every one of our joint projects in an LDAP directory.

The structure of the directory is the following: the employees of the university are stored in a subtree called „University", the administrative employees are stored in the „Personnel" subtree, while the lecturers are stored in the „Faculties" subtree. The „Faculties" subtree consists of the Faculties, the Departments, and the Research Groups in the respective hierarchical order. The employees of the partner companies are listed in the „Partners" subtree, grouped by the name of the company.

The joint projects of the university and the partner companies are listed within the „Projects" subtree. Every project declares *exactly one* administrative colleague from the university, who is responsible for the contracts. The other members from every project can either be non-administrative university employees, or employees of the partner companies.

Task

The IT staff of the university has been aware that some of their colleagues in every project share the same unique identifier number (`uidNumber`), which could cause further problems. So they asked us to develop a script, which finds the people in every project sharing the same identifier number (`uidNumber`), and prints these people in a sorted order into a CSV file.

Develop a **Python 3** script which solves the task!

The name and the parameters of the script

```
get_duplicated_uids.py -p PROJECT -o OUTFILE [-d]
```

The script must use these named parameters. The order of the parameters should not be restricted.

The parameters of the script:

- `PROJECT`: the title (`CN` attribute) of the project, whose members should be inspected (compulsory, string type).
- `OUTFILE:` the path of the output file (compulsory, string type).
- `d:` optional, flag type. If it is set, the order of the records in the output file should be descending (otherwise ascending).

Example executions of the script:

```
python3 get_duplicated_uids.py -p "Project Angel" -o outfile.csv

python3 get_duplicated_uids.py -o /tmp/outfile.csv -p "Project HG2G" -d
```

Input

The script does not have any input beyond the parameters.

Output

The output of the script is an UTF-8 encoded, CSV file with header, in which the records are sorted in ascending order, primarily ordered by the `uidNumber`, secondarily ordered by the `CN` attribute of the result records. If the `d` parameter is set, the ordering should be in descending (for both the `uidNumber` and the `CN`). An example for the content of the output file:

```
uidNumber,CN,Corporation
2568,achetham,Stark Industries
2568,pwentworth,University
2568,sstanhope,LexCorp
3881,aclay,Stark Industries
3881,jsmith,X-Corporation
```

So the output CSV file contains the unique identifier (`uidNumber`), the identifier of each person (`CN`), and the name of the company (if it is the university, then print `University` as it is in the example). If a CSV file already exists at the specified output path, it should be overwritten.

If there is an error during the execution script, print an informative error message to the standard output, beginning with the `ERROR:` string, for example:

```
ERROR: Project is invalid.
```

Further requirements

- Pay attention to the error handling, e.g. always check, whether the given project exists in the LDAP directory. If it does not, print an error message in the format specified above.
- Try to avoid superfluous directory queries, use query (or search) filters that run on the LDAP server.

Hints

- To handle script parameters use the `argparse` module.
    - https://docs.python.org/dev/library/argparse.html
- To access the LDAP server use the `pyldap` module.
    - http://pyldap.readthedocs.org/en/latest/index.html
- To create the CSV file use the `csv` module.
    - https://docs.python.org/3.3/library/csv.html