Please note the following regarding the tasks below:

- Language used should be Python
- The first two tasks (DE) are a must
- The third task (ML) is not a must (but we left links and hints to help you out), however, we would like to see how you would approach it, since you will work closely with our data scientists
- Push your code to GitHub, and send us the link to the repository
- The deadline is Monday, August 8th, EOD
- Lastly, do not hesitate to contact us for any questions :)

[1] DE Task

Write a decorator which takes a sequence of tags and logs a message in case of an exception in the decorated function. For the purpose of this task you can assume you have the following function available to do you log calls:

```
def log(exception_string: str, tags: List[str]):
    …
```

Example usage:

```
@internal_exception_logger('tag_1', 'tag_2')
def test_call():
    raise Exception('This is an exception')
```

[2] DE Task

Two integers A and B are given as input. You are allowed to perform **only one** of the following operations at each step:

- Increase A with A = A * 2 (double it)
- Increase B with B = B * 2 (double it)
- Decrease both A and B by one, e.g. A = A - 1 and B = B - 1

Calculate the minimum number of steps required to make both A and B equal to 0 at the exact same step.

Example 1

A = 1 and B = 2 is the given input. In this case, three operations are required to obtain the desired result:

- First step: A = A * 2 = 2 (now both A and B are equal to 2)
- Second step: A = A - 1 and B = B - 1 (now both A and B are equal to 1)
- Third step: A = A - 1 and B = B - 1 (now both A and B reached 0 in the same step)

Example 2

A = 100 and B = 100 is the given input. In this case, we need 100 steps to achieve our goal. At each step we do the following operation: A = A - 1 and B = B - 1, until both A and B are 0.

Implement the method *find_min_steps* that takes two parameters A and B, and returns the minimum number of steps to achieve our goal. If it's not possible for both A and B to get to 0 at the exact same step, print the appropriate message. The method should look like this:

```
def find_min_steps(a, b):
    …….
    return min_num_of_steps
```

[3] ML Task

Use the following dataset (the *car_data.csv* file) to predict whether a person will purchase a car or not (***Purchased*** column). Remove the ***User ID*** column, and convert the ***Gender*** column to a numerical value (e.g. you can map *Male* to 1, and *Female* to 2, but feel free to take a different approach). You can use any of the supervised algorithms from scikit-learn (some suggestions: Random Forest, KNN, SVM, Naive Bayes). Using the same algorithm, train three models, so that the data is split as follows:

1. 50% data used for training, 50% data used for testing
2. 70% data used for training, 30% data used for testing
3. 80% data used for training, 20% data used for testing

For each version, find the accuracy of the model and store the specs of the model and its accuracy in a log file. For example, the output can look like this:

```
2022-08-03 14:30:00 model=Random Forest, split=50-50, accuracy=0.57
2022-08-03 14:30:00 model=Random Forest, split=70-30, accuracy=0.68
2022-08-03 14:30:00 model=Random Forest, split=80-20, accuracy=0.77
```