

Low cohesion

There are several problems with the code in the exercise:

- Everything the code does is contained with a single `main` function. This makes it difficult to understand what the code is doing and how it is organized.
- It's difficult to change the code. For example, it's currently hard to reuse the data processing code in another script that reads the data from for example an Excel file.
- Because the function does so many different things, it's hard to test. For example, it's hard to test the data processing code without also running the CSV reading and writing code.

Here's an example of how you can refactor the code to increase cohesion:

```
Record = dict[str, Any]

INPUT_FILE = "data.csv"
OUTPUT_FILE = "processed.csv"
FIELD_NAMES_OUTPUT = ["name", "status", "is_active"]

def process_data(row: Record) -> Record:
    row_copy = row.copy()
    if row_copy["status"] == "active":
        row_copy["is_active"] = True
    else:
        row_copy["is_active"] = False
    return row_copy

def read_csv(filename: str) -> list[Record]:
    with open(filename) as f:
        reader = csv.DictReader(f)
        return list(reader)

def write_csv(filename: str, fieldnames: list[str], rows: list[Record]) -> None:
    with open(filename, "w") as f:
        writer = csv.DictWriter(f, fieldnames=fieldnames)
        writer.writeheader()
        writer.writerows(rows)

def main() -> None:
    data = read_csv(INPUT_FILE)
    processed_data = [process_data(row) for row in data]
    write_csv(OUTPUT_FILE, FIELD_NAMES_OUTPUT, processed_data)
```

I've done a couple of things here:

- The code has been split into several functions. This makes it easier to understand what the code is doing and how it is organized. There's now a separate function for reading csv files, a separate function for writing csv files, and a separate function for processing the data.
- I've also added some constants for the input and output file names and the field names for the output file. This makes it easier to change the file names and field names to export in the future.

For the full solution, see the [solution.py](#) file.