Task 1: Pattern Matching Practice

Goal: Extract values using pattern matching.

```
defmodule Matcher do
  def extract_name({:ok, %{name: name}}), do: name
  def extract_name(_), do: "No name found"
end
```

Exercise: Modify the function to handle:

- A tuple like {:error, "not found"}
- A map with keys :first_name and :last_name

Task 2: Working with Lists

Goal: Count how many even numbers are in a list.

```
defmodule ListOps do
  def count_evens(list) do
    # Your code here
  end
end
```

Extension: Return both the even numbers and their count.

Task 3: Tuples and Maps

Goal: Convert a list of tuples into a map.

```
defmodule TupleToMap do
  def convert([{:name, "Alice"}, {:age, 25}]) do
    # Expected output: %{name: "Alice", age: 25}
  end
end
```

Extension: Write the reverse function: convert a map into a list of tuples.

Task 4: Enum Functions

Goal: Capitalize all names in a list.

```
defmodule NameFormatter do
  def capitalize_names(["alice", "bob", "carol"]) do
    # Your code here
  end
end
```

Extension: Filter out names shorter than 4 characters.

Task 5: Conditionals Practice

Goal: Categorize a number as positive, negative, or zero.

```
defmodule Categorizer do
  def categorize(n) do
    # Use cond or case
  end
end
```

Task 6: Recursive Function

Goal: Implement a recursive function to sum a list of numbers.

```
defmodule Recursion do
  def sum([]), do: 0
  def sum([head | tail]) do
    # Your code here
  end
end
```

Task 7: Simple Structs

Goal: Define a User struct and write a function that takes a %User{} and returns a greeting.

```
defmodule User do

# Your code here

end

defmodule Greeter do

  # Your code here

end
```