Exercise 1:

Question 1:

How would you write a “transform” function having the following behavior:

```
const input = [
  ["key1", 1, 2, 3, 4],
  ["key2", 4, 5, 6, 7]
]
transform(input)
// output :
// {
//   key1: [1, 2, 3, 4],
//   key2: [4, 5, 6, 7],
// }
```

Question 2:

How would you write a “transform” function having the following behavior:

```
const input = [
  {
    price: 3000,
    kind: 'reductionVoucher',
    savings: 300,
    savingsType: 'total',
  },
  {
    price: undefined,
    kind: 'sales',
    savings: 10,
    savingsType: 'percent',
  },
  {
    price: 3000,
    kind: 'sales',
    savings: 15,
    savingsType: 'percent',
  }
]
transform(input)
// output :
// [
//   {
//     basePrice: '3 000 €',
//     description: 'You are saving 300 € related to the initial price',
//     price: '2 700 €',
//     savings: '300 €',
//     title: Flat voucher',
//   },
//   {
//     basePrice: '3 000 €',
//     description: 'You are saving 450 € related to the initial price',
//     price: '2 550 €',
//     savings: '15 %',
//     title: 'Sales',
//   }
// ]
```

Exercise 2:

Question 1: Write a function to generate a random genealogical tree, using the "faker" library. Each person of the genealogical tree should be represented by the property "name" (a string) and the property "children" (a list of persons). Each person should have between 0 and 2 children.

Ex:
```
{
  "name": "Aurora Kirby",
  "children": [
    {
      "name": "Reyansh Finney",
      "children": []
    },
    {
      "name": "Caris Flowers",
      "children": [
        {
          "name": "Aedan Foster",
          "children": []
        }
      ]
    }
  ]
}
```

Question 2: What would happen if each person can have between 0 and 3 children instead of between 0 and 2 children in the exercise? Why?

Exercise 3:

Question 1: Make a function which generate a random number of person, each person having a name and a country. The name and the country can be generated by the faker library.

Question 2: Make a function which display the list of names inside each country, as follow:

```
Citizen of Mauritania: Vicente Hodkiewicz,Halle Beatty
Citizen of Trinidad and Tobago: Kelvin Cronin,Keagan Jakubowski
…
```

The function should be able to order 1 000 000 persons in less than a second

Exercise 4 (bonus):

You will write a function which takes in input an object and returns a list of strings.

The input object has between 1 and 3 properties which can be a leaf (a string in this exercise) or a node (an object in this exercise). A node has between 1 and 3 properties which can be a leaf or a node, and so on. The depth of the object cannot exceed 3 (in the 3rd level, there is no more nodes, only leaves).

Question 1: Represent the tree in a JSON format.
if you are totally lost at this point, ask me to give you the solution so that you can continue the exercise

Question 2: Write the list of possible paths through the object you designed for the question 1. Those paths must be a string starting by the different nodes contained in the object concatenated by underscore "_" and finishing by the leaf, as followed: node1_node2_leaf
if you are totally lost, ask me and I'll give you the answer to help you for the rest

Question 3: Write the function which takes the response of the question 1 as input and that returns the answer of the question 2 (nb: this question can be treated the same time as the question 4 if you prefer)

Question 4: Write a test for the function, taking the response of the question 1 as input and giving the response of the question 2 as output. You will use the library "jest" with the "expect" statement, and the method "toEqual" to compare the string.