Kaffa - Pre-qualification test (2019/2S)

Objective

Complete **at least 2** of the exercises below.

Important: it's NOT necessary to complete all exercises, but you can do as many as you want.

Instructions

- Choose any programming language, library, tool or framework, as long it's open source and freely available;
- Every exercise must be done in a ready to execute format, so that is possible to build, run and see actual results;
- You can develop the code of the exercises in any form or platform (Desktop App, Web, Android, iOS, command line);
- You can merge multiple exercises in one single application/executable, as long as it's easy to identify the individual exercises;
- The code should come with instructions on how to build and run, including any dependency or setup instruction;
- You **can write** your comments and documentation **in portuguese** (writing in english is a plus);
- Submit the results as a Github repository, created for this exam;
- Use the same repository for all the exercises;
- Complete the tasks yourself without help and without copying and pasting code from the internet;
- Don't submit code you don't understand, you'll be asked in the interview about your solution;
- You can ask any question by email (in portuguese or english).

Deadline

One week (7 days).

The sooner, the better.

Evaluation criteria

- Correctness of the solution
- Presentation of the content
- Code organization
- Techniques and practices
- Complexity of the exercises
- Time to completion

Exercises

1) Validate CNPJ Format (Mask)

Given a string, check if it looks like a CNPJ, considering two formats:

- Formatted:
  `"00.000.000/0001-00"`
- Number only:
  `"00000000000100"`

2) Validate CNPJ Digits

Given a string validate if it's a well-formed CNPJ, considering the "check digits" as defined by *Receita Federal*.

3) Compute area of intersection between two rectangles

Considering two rectangles, defined by its boundaries, compute the area of intersection between the two.

Example:

```
.
.  +--------+
10 |        |
.  |   A    |
.  |        |
.  |   ######-+
.  |   ###### |
5  +---###### |
.      |  B   |
.      |      |
.      +------+
.    .    .     .
0....5....10....15
```

```
A = (3, 5; 11, 11)
B = (7, 2; 13, 7)

areaOfIntersection(A, B) = 18
```

4) Simple Todo list

Todo list application that permits the creation and deletion of tasks (texts).

- The application must persist the tasks between executions;
- Use any storage you want: database, files, PaaS backends (Firebase, etc.);

5) Rest Client - World Clock

Application that queries a server and displays the current date/time hour in local and UTC timezones.

Server URL: http://worldclockapi.com/api/json/utc/now

6) Rest Server - World Clock

REST server returning a JSON like:

```
{
    "currentDateTime":"2019-08-12T14:40Z"
}
```

7) Entity Relationship Diagram - Simple Order Manager

Design the model of a simple Order Manager System.

The system consists of:

- Clients
- Products
- Orders

You can draw, describe, or list the tables as SQL.

Extras:

- SQL: list ORDERS with number of items
- Which indexes should be created in this model?

Attention: this exercise is documentation only - there's no executable to run in this case.