Introduction

Dear Candidate,

You will find in this document a set of questions and small challenges to assess your knowledge, reasoning and coding skills.
Please answer them as much as possible. When multiple-choice answers are proposed, make sure to highlight clearly your choice of answer. Note that in such cases, there may be more than one correct answer.
Points for each question are indicated, incorrect answers will be graded negatively (-1 point), no answer will reward 0 point so guessing is not ~~recommended~~.

There is no imposed programming language for the coding part, even if our preference goes to Python.

Please send us your answers and code contributions within a week, along with the directives to execute your code.

Thanks a lot and good luck.

Name: NGUYEN DINH DO Thai-Nhiên

Date: 14/08/2025

A. Reasoning / Statistics

1. Assume a family has two children, of which at least one is a boy.
What is the probability that the family has two boys? 1 pt

⇨ a. 1/2 [✓ b. 1/3] c. 2/3 d. 5/8

2. On an analog watch, what is the angle between the big and small pointer at 3:30pm, looking at the smaller angle? 1 pt

⇨ a. 30 degrees b. 66.7 degrees [✓ c. 75 degrees] d. 90 degrees

3. Assume two players play a game, throwing a dice in turns. The first one to draw a 6 wins the game and the game ends. Assuming you start, what is the probability of winning the game in that case? 1 pt

⇨ a. 1/2 b. 3/4 [✓ c. 6/11] d. 19/36

4. A linear correlation of 0.90 between two variables necessarily indicates a causal link. 1 pt

⇨ a. True [✓ b. False]

5. Which of the techniques below can remove one of two strongly correlated variables? 1 pt

⇨ a. Quantile regression [✓ b. Residual] [✓ c. Orthogonalization] d. Variable Ranking

6. Which of the following visualizations does help to represent graphically the distribution of a variable? 1 pt

⇨ a. Galton Anamorphosis [✓ b. Boxplot] c. Scatter Plot d. Self-correlation graph

7. Which of these items can be a component in plotting a time series? 1 pt

⇨ a. Referral [✓ b. Trend] [✓ c. Periodicity] [✓ d. Noise] e. All

8. Here is a series of numbers. Could you complete the series with the missing item? 1 pt

1 | 5 | 15 | 31 | **?** | 81

⇨ a. 42 b. 49 [✓ c. 53] d. 67

9. Here are some timetable information from the local train station.

- *Journey C* lasts twice as long as *Journey D*.
- Taking *Train A* halves the duration of *Journey B*.
- *Train C* reduces the duration of *Journey D* by five hours.
- *Journey A* and *Journey B* last for eight hours.
- Taking *Train B* halves the duration of *Journey A* and *Journey C*.
- *Journey B* lasts half the duration as *Journey D*.

Based on the information above, please answer the four questions listed below.

1) Taking *Train B* will reduce the duration of *Journey C* by twelve hours. 1 pt

⇨ a. True [✓ b. False] c. Insufficient Information

2) *Journey D* lasts for 18 hours. 1 pt

⇨ a. True [✓ b. False] c. Insufficient Information

3) Taking *Train C* will reduce the duration of *Journey D* to eleven hours. 1 pt

⇨ [✓ a. True] b. False c. Insufficient Information

4) Taking appropriate Train, *Journey D* lasts longer than *Journey A*. 1 pt

⇨ [✓ a. True] b. False c. Insufficient Information

B. Algorithmic / Computer Science

1. What do you think is the best definition of D3.js? 1 pt

⇨ a. a Graphical Bookshop b. a Graphical layer c. a Data Viewing Tool

2. Propose a document-oriented data model (json) for “record 1” in the below table: 1 pt

Doc_Id	Colour	Size	X3
1	Blue	Tall	12

⇨

```
{
"Doc_Id": 1,
"Colour": "Blue",
"Size": "Tall",
"X3": 12
}
```

3. Which of the following do not represent a NoSql format? 1 pt

⇨ a. Graph b. Oriented column c. Diagonal d. Documents e. Key-Value

4. Among the following front-end web framework, which are best suited to fully managing authentication to a website: 1 pt

⇨ a. AngularsJS b. ReactJS c. Bootsrap d. None.

5. What is the mechanism for isolating an application without redefining an operating system layer in a deployed server? 1 pt

⇨ a. Container b. VM c. Name Space

6. Docker is a widespread container system, what system makes it possible to organise an application deployed with such a mechanism? 1 pt

⇨ a. Scheduler b. Orchestrator c. Moderator

7. What is the mechanism for monitoring and evaluating an application in Prod? 1 pt

⇨ a. Monitoring b. Scheduling c. Logging

8. The following assignments are considered. It is assumed that assignments on the same line are parallel and isolated, and sequential from one line to another:

```
x, z, y =  10, 30, 5 ;
x, z, y = x*y,y*x*z, x*z ;
z = z==15*10**2 ;
```

What is the value of z at the end of the execution?

⇨ True

1 pt

9. The time complexity of an algorithm depends on the

⇨ a. programming language b. number of processors [✓ c. number of items to be processed]

1 pt

10. Let A and B, two algorithms performing the same function, if A has a complexity smaller than B, then A is always faster than B whatever the number of processors.

⇨ a. True [✓ b. False]

1 pt

11. Let look at a data table (200 million lines) on which we want to retrieve lines that meet at least one of the three criteria on the columns (A, B, C), criterion A (1 million cases passing), criterion B (10 million cases passing), criterion C (100 million cases passing). In what order would you run the tests to select the data?

1 pt

⇨ C then B then A
The reason would be to make less computations by stopping as early as possible
(C has most cases passing, then B then A)

12. Consider the code below.

```
def fibo(n):
                print(n)
                if n <= 1:
                                return n
                else:
                                return fibo(n-1) + fibo(n-2)
```

If we call 'fibo(12)', how many times will you see the number "8" displayed?

1 pt

⇨ a. 1 b. 2 [✓ c. 5] d. 8

13. What is the output of the following python code?

1 pt

```
sum([x*x for x in range(10) if x%2 == 0])
```

⇨ [✓ a. 120] b. 255 c. 20 d. 220

14. What is a recursive function?

a. A function that is called many times from the main code
b. A function that returns a function object as a result
[✓ c. A function that calls itself]
d. A function without any arguments
e. A function that takes one or more functions as arguments.

1 pt

⇨

15. If we execute the following R commands (written below), what will be the output? 1 pt

```
f = function(x) {
        g = function(y) {
                y + z
        }
        z = 4
        x + g(x)
}
z = 10
f(4)
```

⇨ a. 12 b. 7 c. 4 d. 16

16. What is the output of the following Python command: `Print "Hello World"[::-1]` 1 pt

⇨ a. dlroW olleH b. Hello Worl c. d d. Error

17. Indexing data into a database is a technique for creating time series. 1 pt

⇨ a. True b. False

18. Which of these control lines are equivalent?

A) `df['status2'] = numpy.where(df.status == True, 1, 0)`

B) `df['status2'] = df['status'].map(lambda s: 1 if s==True else 0)`

C) `df['status2'] = 1 if df['status2']  == True else 0`

D) `df['status2'] = df['status'].map(lambda s: 1 if s==False else 0)`

⇨ A and B

1 pt

19. What method makes it possible to compare several models of recommendation in production? 1 pt

⇨ a. Double Testing b. A/B Testing c. Cross-validation

20. Which of the following are systems for establishing a CI/CD mechanism? 1 pt

⇨ a. Jenkins b. Gitlab CI c. Git d. TFS

21. Write the fastest code as possible to execute the tasks below.

- create 100 randomly selected values between 0 and 10. 1 pt

⇨
```
import numpy as np
vals = np.random.rand(100) * 10
```

- calculate the sum of the squares of these values. 1 pt

⇨
```
result = np.sum(vals**2)
```

22. Answer briefly (1-2 sentences) the following questions:

a) if you had to build a website, which language and tools would you use?

⇨ For the front-end: React/Next.js + TypesScript
For the back-end: Node.js/Express
For BDD: PostGres + Prisma
CI/CD: GitLab

2 pts

b) How are containers different to virtual machines?

⇨ Containers are lighter than VMs.
They share a light OS whereas VM get the entire OS.

2 pts

c) What is CRUD?

⇨ Create, Read, Update, Delete which correspond to basic operations on resources

2 pts

d) How would you measure the latency/performance of a piece of software?

⇨ I would measure the response time, use load tests and profile for bottlenecks.

2 pts

e) How would you measure the quality of a piece of software?

⇨ I would do test coverage, compute regression rates, bugs in production, resolution times,
code review and user satisfaction

2 pts

23. Here is the description of the FizzBuzz problem:

Given an integer n, return a string array answer (1-indexed) where:

answer[i] == "FizzBuzz" if i is divisible by 3 and 5.

answer[i] == "Fizz" if i is divisible by 3.

answer[i] == "Buzz" if i is divisible by 5.

answer[i] == i (as a string) if none of the above conditions are true.

Example: Input: n = 5

Output: ["1","2","Fizz","4","Buzz"]

And a proposed implementation of the solution:

```
def fizzBuzz(num):
        for i in range(1, num):
                if i % 3 == 0:
                        print(i, "Fizz")
                elif i % 5 == 0:
                        print(i, "Buzz")
                elif i % 3 == 0 and i % 5 == 0:
                        print(i, "FizzBuzz")
                else:
                        print i
```

What are the three most important issues with the proposed implementation of the fizzBuzz() function above?

⇨

The three most important issues are:
1) The order of tests is false. We should test the last condition (divisible by 3 and 5) first, otherwise we will never reach this condition
2) The range loop should go from range(1,num+1) otherwise number n will not go into the loop.
3) The function print each subresults instead of printing a list as expected. We should append each subresults within a list.

1 pt

24. How might you improve this function?

```
def crawl_directory(crawl_path):
        """
        Crawl a directory and return the list of files
        :param crawl_path: dir path to crawl
        :return: int list of files in path
        """
        return len(list(Path(crawl_path).glob('*')))
```

⇨

1 pt

25. How could you reduce the following list to a list of unique elements?
names = [1, 2, 2, 2, 3, 4, 5, 6, 6, 7, 9, 8, 8, 9, 8, 9, 9, 8 ,8 ,1, 2, 2, 3, 4, 5, ...]

⇨ In python:
list(set(names))

1 pt

26. Describe how the try/except statement works in python?

⇨ In python, we first execute the Try statement. In the Try statement, if and error occurs it goest directly within the appropriate Except (ie the Except statement that handles the error) statement and execute it.

2 pts

How would you use it to code a function doing:

- Divide a per b, then add 1, and apply square power,
- If b is 0, then divide a per b+1 and apply square power,
- Add 1 to the result

⇨

```
def compute(a, b):
    try:
        res = a / b
    except ZeroDivisionError:
        res = a / (b + 1)
    res = (res + 1) ** 2
    return res + 1
```

1 pt

C. SQL

For this set of questions around SQL, you will first need to create some tables in a SQL database and fill them with data.
You find below the instructions covering these different steps.
Once your database is set up, you will have a set of questions to answer. We are expecting both the output data generated by the query and the SQL query itself.

Table Creation

1. Create the country table

~~~~
```sql
CREATE TABLE if not exists countries (
        country_id VARCHAR (2) PRIMARY KEY,
        country_name VARCHAR (40) DEFAULT NULL,
        region VARCHAR(10) NOT NULL
);```
~~~~

2. Create the employee table

~~~~
```sql
CREATE TABLE if not exists employees (
        employee_id SERIAL PRIMARY KEY,
        first_name VARCHAR (20) DEFAULT NULL,
        last_name VARCHAR (25) NOT NULL,
        email VARCHAR (100) NOT NULL,
        phone_number VARCHAR (20) DEFAULT NULL,
        manager_id INT DEFAULT null REFERENCES employees (employee_id),
        department_id INT DEFAULT NULL,
        country_id VARCHAR (2) REFERENCES countries (country_id)
);```
~~~~

3. Filling the country table first

~~~~
```sql
INSERT INTO countries VALUES ('AR','ARGENTINA', 'lamr');
INSERT INTO countries VALUES ('BR','BRAZIL', 'lamr');
INSERT INTO countries VALUES ('ES','SPAIN', 'euro');
INSERT INTO countries VALUES ('FR','FRANCE', 'euro');
INSERT INTO countries VALUES ('GB','UNITED KINGDOM', 'euro');
INSERT INTO countries VALUES ('HK','HONG KONG', 'asia');
INSERT INTO countries VALUES ('IT','ITALY', 'euro');
INSERT INTO countries VALUES ('US','UNITED STATES', 'namr');
```
~~~~

4. Filling the employee table

~~~
```sql
INSERT INTO employees (first_name, last_name, email, phone_number, manager_id, department, country_id)
VALUES ('Jean','AAA', 'jean.aaa@mycompany.fr', '32359949', NULL, 'Sales', 'FR');
INSERT INTO employees (first_name, last_name, email, phone_number, manager_id, department, country_id)
VALUES ('Michel','BBB', 'michel.bbb@mycompany.es', '81945947', '1', 'HR', 'ES');
INSERT INTO employees (first_name, last_name, email, phone_number, manager_id, department, country_id)
VALUES ('Juliane','CCC', 'juliane.ccc@mycompany.br', '87622749', '1', 'Sales', 'BR');
INSERT INTO employees (first_name, last_name, email, phone_number, manager_id, department, country_id)
VALUES ('Patrice','DDD', 'patrice.ddd@mycompany.hk', '13931578', '1', 'HR', 'HK');
INSERT INTO employees (first_name, last_name, email, phone_number, manager_id, department, country_id)
VALUES ('Louis','EEE', 'louis.eee@mycompany.fr', '68154273', '1', 'Sales', 'FR');
INSERT INTO employees (first_name, last_name, email, phone_number, manager_id, department, country_id)
VALUES ('Albert','FFF', 'albert.fff@mycompany.hk', '63142944', NULL, 'IT', 'HK');
INSERT INTO employees (first_name, last_name, email, phone_number, manager_id, department, country_id)
VALUES ('Ludivine','GGG', 'ludivine.ggg@mycompany.fr', '63280299', '6', 'IT', 'FR');
INSERT INTO employees (first_name, last_name, email, phone_number, manager_id, department, country_id)
VALUES ('Sophie','HHH', 'sophie.hhh@mycompany.br', '66811818', '6', 'IT', 'BR');
INSERT INTO employees (first_name, last_name, email, phone_number, manager_id, department, country_id)
VALUES ('Camille','III', 'camille.iii@mycompany.gb', '24205021', '6', 'Sales', 'GB');
INSERT INTO employees (first_name, last_name, email, phone_number, manager_id, department, country_id)
VALUES ('Frank','JJJ', 'frank.jjj@mycompany.it', '83318015', NULL, 'Sales', 'IT');
INSERT INTO employees (first_name, last_name, email, phone_number, manager_id, department, country_id)
VALUES ('Thomas','KKK', 'thomas.kkk@mycompany.ar', '20430282', '9', 'Sales', 'AR');
INSERT INTO employees (first_name, last_name, email, phone_number, manager_id, department, country_id)
VALUES ('Eric','LLL', 'eric.lll@mycompany.us', '86227391', '9', 'HR', 'US');
INSERT INTO employees (first_name, last_name, email, phone_number, manager_id, department, country_id)
VALUES ('Thierry','MMM', 'thierry.mmm@mycompany.gb', '93130273', '9', 'Sales', 'GB');
INSERT INTO employees (first_name, last_name, email, phone_number, manager_id, department, country_id)
VALUES ('Nathalie','NNN', 'nathalie.nnn@mycompany.es', '41576574', '9', 'IT', 'ES');
```
~~~

Questions

1. Select all employees (all columns) sorted by country in alphabetically order.

```
SELECT e.*
FROM employees e
LEFT JOIN countries c ON e.country_id = c.country_id
ORDER BY c.country_name;
```

1 pt

2. Select all employees having a manager.

1 pt

```
SELECT * FROM employees WHERE manager_id IS NOT NULL;
```

3. Select all managers' details (being manager of at least 1 person).

1 pt

```
SELECT DISTINCT m.*
FROM employees m
JOIN employees e ON e.manager_id = m.employee_id;
```

4. Select all employees having a manager, with the "first_name" and "last_name" of the respective manager.

1 pt

```
SELECT e.*, m.first_name AS mgr_first, m.last_name AS mgr_last
FROM employees e
JOIN employees m ON e.manager_id = m.employee_id;
```

5. Do the same request as 4. but including people with no managers.

1 pt

```
SELECT e.*, m.first_name AS mgr_first, m.last_name AS mgr_last
FROM employees e
LEFT JOIN employees m ON e.manager_id = m.employee_id;
```

6. Return the number of employee per country, sorted by descending order, and using the "countries" table to display the country name and region.

1 pt

```
SELECT c.country_name, c.region, COUNT(e.employee_id) AS nb_employees
FROM countries c
LEFT JOIN employees e ON e.country_id = c.country_id
GROUP BY c.country_name, c.region
ORDER BY nb_employees DESC;
```

7. Do the same request, but only display the countries for which there are more than one employee

1 pt

```
SELECT c.country_name, c.region, COUNT(e.employee_id) AS nb_employees
FROM countries c
JOIN employees e ON e.country_id = c.country_id
GROUP BY c.country_name, c.region
HAVING COUNT(e.employee_id) > 1
ORDER BY nb_employees DESC;
```

D. Coding Challenges

1. Same Type Concatenation

7 pts

Create a function that concatenates lists of elements depending of their types as follows:

- int / float --> should return a sum,
- str --> should return a concatenated string separated by pipes (‘|’),
- bool --> should return a string of 1(if true)/0(if false) concatenated,
- list --> concatenated list,
- dict --> concatenated dict.

If items have different types, then raise an exception

2. Balanced Symbols

12 pts

Implement a boolean function that checks that an input string is properly balanced with respect to '(' and ')', '[' and ']' and '{' and '}'.

In addition, the string may contain comments, which start with '/*' and end with '*/'. (The '*' cannot be shared between the start comment token and the end comment token).
Any symbol within a comment should be ignored.
A string with an unclosed comment is considered unbalanced.

Symbols can be nested, e.g. "({ [() []] })" is balanced.
However, comments cannot, i.e. in "/* abcd /* efgh */ ijkl */", the comment is simply “/* abcd /* efgh */”, and the rest of the string is not within the comment. (this string is balanced by the way…)

A few examples of unbalanced strings : "( ]", or "([ )]"

Null and empty strings are considered balanced.

3. Cost prorating

12 pts

Implement a function that takes a cost and a set of weights and returns the distribution of the cost proportionally to the weights, with the initial cost and all the prorated costs being integers.

The order of elements in the returned array corresponds of the order of the weights

The total amount distributed must be exactly the initial cost.

- For a weight equal to 0, prorated cost must be 0.
- For a weight > 0, rounded prorated cost must differ from the exact prorated cost by strictly less than 1. In that case we define the relative rounding error as Abs(roundedProrata - exactProrata) / weight. (So the relative error for a rounding difference of 0.5 on a weight of 1 is 0.5; and the relative error for a rounding difference of 0.6 on a weight of 2 is 0.3).

The algorithm should minimize the deviation, which is the sum of all relative rounding errors.

The cost is greater than or equal to 0.
The set of weights is non null and contains at least one non-zero weight.
Each weight is greater than or equal to 0.

4. Water Jugs

12 pts

You have N (N > 0) jugs of water, with a given capacity (say in liters), a faucet and a sink.

You are asked to find a sequence of operations that will yield a state where at least one jug contains a target volume of water.

You have three types of pouring operations:
- Fill a jug from the faucet
- Empty a jug in the sink
- Pour a jug into another jug. In that case, the operation ends when the source jug is empty, or when the destination jug becomes full, whichever comes first.

(… It might ring a bell => https://www.youtube.com/watch?v=BVtQNK_ZUJg)

Implement a function that, given a target volume and a set of jugs' capacities, returns a minimal sequence of pouring operations to reach a state where at least one jug has the target volume of water in it.
If no such sequence exists, it should return null.

The set of jugs' capacities defines the maximum number of liters of water each jug can contain.
This set should be non-null and non-empty. Each capacity is strictly greater than zero.
The target volume is strictly greater than zero.

A pouring step in function output should contain two parts:
- the pouring operation itself,
- the state of each jug, i.e. its volume of water, after the pouring operation.

You could define the pouring operation as a 2-tuple (Source, Destination), with
- Source: the jug to pour from, or -1 to fill from the faucet,
- Destination : the jug to pour into, or -1 to empty into the sink

For example, like in "Die Hard", such function with input (target = 4, jugs' capacities = [3, 5]) should yield a 6 operations sequence as a result:

```
-1 ->  1 : (0,5)
 1 ->  0 : (3,2)
 0 -> -1 : (0,2)
 1 ->  0 : (2,0)
-1 ->  1 : (2,5)
 1 ->  0 : (3,4)
```