



**FOUNDATION EXAM - 2 HOURS**

|  |  |
| --- | --- |
| **SECTION** | **MARK** |
| **Theory Questions** | 31 |
| **Concept Questions** | 19 |
| **Challenges: students should choose two out of the three to answer** | |
| **Python Challenge** | 25 |
| **SQL Challenge** | 25 |
| **Javascript Challenge** | 25 |
| 1. **TOTAL** | **100** |

**Important notes:**

* Any code files written **must be submitted via a Pull Request to your marker**.
* You can submit theory questions through an edited version of this document on Slack, or on the Pull Request by adding python comments into a new file, or using a text or markdown file.
* You are allowed to submit everything on Slack if it is close to the deadline, as long as you work on getting a pull request up soon after.
* It is an open book exam.
* You are allowed to use PyCharm, MySQL Workbench, and/or CodePen for this assessment.
* If you are using CodePen, please add your CodePen URL in a text or markdown file.
* You only need to complete **two coding challenges**. Please choose two coding challenges of the three provided (Python, SQL and Javascript) to answer.

**Section 1: Theory Questions [31 marks]**

|  |  |
| --- | --- |
| * 1. **What does SDLC stand for?**   The Software Development Life Cycle (SDLC) is a structured process that enables the production of high-quality, low-cost software, in the shortest possible production time. The goal of the SDLC is to produce superior software that meets and exceeds all customer expectations and demands. The SDLC defines and outlines a detailed plan with stages, or phases, that each encompass their own process and deliverables. Adherence to the SDLC enhances development speed and minimizes project risks and costs associated with alternative methods of production. | **1 mark** |

|  |  |
| --- | --- |
| * 1. **What exception is thrown when you divide a number by 0?**   ArithmeticException | **1 mark** |

|  |  |
| --- | --- |
| **1.3 What is the git command that moves code from the local repository**  **to the remote repository?**  Push | **1 mark** |

|  |  |
| --- | --- |
| **1.4 What does NULL represent in a database?**  Data is missing, not filled-in. Unknown, which is different from 0. | **1 mark** |

|  |  |
| --- | --- |
| **1.5 Name 2 responsibilities of the Scrum Master** Scrum Master’s service to the development team, eg :  * Ensuring the development team is working effectively * Facilitating Scrum events as requested or needed  Scrum Master’s service to the organisation, eg :  * Acting as a change agent that increases the productivity of the team * Leading and guiding the organization in successful Scrum adoption, by streamlining processes and ensuring that every member on the team follows the Scrum framework | **2 marks** |

|  |  |
| --- | --- |
| **1.6 Name 2 debugging methods, and when you would use them.**   1. ***Breakpoint***. Regularly at a few steps checking correct running of the code at different stages to smoothly check every step of code. Stop the code at some selected point, where I understand what the output should be. Check the actual output against the expected correct. If there is an error, and not yet realized where, run the code again and stop at the earlier point and check again. 2. ***Debugger window view***. Useful to look for the exception messages, which might occur often, and more easily verified / viewed in a debugger window. | **4 marks** |

|  |  |  |
| --- | --- | --- |
| **1.7 Looking at the following code, describe a case where this function**  **would throw an error when called.** Describe this case and talk about  what exception handling you’ll need.  Possible errors :   * Incompatible datatype (integers, decimals) * Negative value of any of the parameters  |  | | --- | | **def can\_pay(price, cash\_given):**  **if cash\_given >= price:**  **return True**  **else:**  **return False** | | **5 marks** |

|  |  |
| --- | --- |
| **1.8 What is git branching?** Explain how it is used in Git.  It is working in smaller separate parts of code as if in parallel ‘branch’, which allows not to affect the main ‘branch’ and later update the main ‘branch’ with this part of job. It allows to work parallel on a different parts of project and not to put at risk the whole project incidentally, and allow to keep clear history of changes. | **6 marks** |

|  |  |
| --- | --- |
| **1.9 Design a restaurant ordering system.**  You do not need to write code, but describe a high-level approach:   * 1. Draw a list of key requirements   2. What are your main considerations and problems?   3. What components or tools would you potentially use?  1. List of requirements :   *Requirements to food/drinks :*   * Groups of food and drinks by types, at a detailed segregation to allow customers select. Eg. * Savory / Meat-fish-vegetarian / * Deserts * Cuisine style (Italian, Georgian, etc) * Drinks * Propose suggestion for combination (eg. Cuisine style and suggested drink) * Sauces, side dishes suggestion * List of ingredients to be used for each dish : name and quantity per standard dish * Options of different dish volume ? * Group suggestions * Adults + children combination suggestion   *Requirements to the clients’ ordering process :*   * Options to select dishes subject to available grouping above, related to clients * Stay in or take-away * Timing of ordering * Payment options  1. Main considerations :  * Match dishes available with the ingredients required on stock * Prognose time of table reservation to maximize tables being ordered effectively (of course subject to reasonable assmptions) * Estimate the popularity of ordered dishes to plan for future menu structure * I would offer suggested combination where possible upon of selection of some dish * It seems interesting to offer some informational tips about selected food as an additional ‘entertainment’ | **10 marks** |

**Section 2: Concept Questions [19 marks]**

|  |  |
| --- | --- |
| **2.1 Write a function that takes in an input and checks to see if it’s an**  **isogram. The function should return True or False.**    An isogram is a word where no letter is repeated.  Examples include:   * "isogram" * "uncopyrightable" * “ambidextrously”   The function can work as either of 2 options :   1. Check if each next letter (i+1) = to each of the prior ones, and as soon as some letter matches prior one return “false” 2. Assign each letter ‘Alphabetic ID’, then for each element of ‘Alphabet’ ID list count number times this ID is mentioned in the given word. If at least one ‘Alphabetic ID’ counts >1, return “false” | **7 marks** |

|  |  |
| --- | --- |
| **2.2 Make a new test file and write comprehensive unit tests for the**  **function you wrote in 2.1**  For each test case add a comment stating why you chose that case. | **12 marks** |

**Section 3: Python Challenge [25 marks]**

You are tasked with calculating the minimum classes we need to have so we know how many people to employ. Write a function which when given a number of students, calculates and prints out a string for your proposed number of classes, and a dictionary showing the allocation.

***Key Constraints:***

* The maximum size of a class is 30
* There needs to be a minimum of 2 classes
* The distribution of each class should be as even as possible.
* We want to hire as little people as possible - so where possible focus on bigger classes, and less of them!

***Inputs/Outputs***:

* If 31 was the input, the output would be:

|  |
| --- |
| Proposed Allocation: 2 classes  {'Class 1': 16, 'Class 2': 15} |

* If 59 was the input, the output would be:

|  |
| --- |
| Proposed Allocation: 2 classes  {'Class 1': 30, 'Class 2': 29} |

* If 87 was the input, the output would be:

|  |
| --- |
| Proposed Allocation: 3 classes  {'Class 1': 29, 'Class 2': 29, 'Class 3': 29} |

**Section 4: SQL Challenge [25 marks]**

In this section you will be fleshing out a database and performing queries.

**Starter Code:**

CREATE DATABASE foundation\_exam;

USE foundation\_exam;

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **4.1 Write (and execute) syntax to create the following tables:**  Example data is included to help you choose suitable data types  **A] *movie\_info*** *Table*     |  |  |  |  | | --- | --- | --- | --- | | Movie\_ID | Movie\_Name | Movie\_Length | Age\_Rating | | 1 | The Movie | 1:35:00 | 12A |   **B] *screens*** *Table*     |  |  | | --- | --- | | Screen\_ID | Four\_K | | 1 | False |   **C] *showings*** *Table*     |  |  |  |  |  | | --- | --- | --- | --- | --- | | Showing\_ID | Movie\_ID | Screen\_ID | Start\_Time | Available\_Seats | | 1 | 1 | 1 | 12:00:00 | 23 | | **10 marks** |

|  |
| --- |
| **Populate the database!**  Use the file*foundation\_exam.sql* to fill your tables with the needed data.  You may need to change the names of the tables in the SQL file if yours don’t align. |

|  |  |
| --- | --- |
| **4.2 Write a query to return the name and time of all movies that play after**  **12:00 given there is at least 1 available seat. Display the results in time**  **order.**  **SELECT** m.movie\_name**,** s.start\_time  **FROM** showings s  **JOIN** movie\_info m  **ON** s.movie\_ID=m.movie\_ID    **WHERE** s.start\_time > 12:01:00  **AND** s.available\_seats >0  **ORDER** **BY** s.start\_time | **6 marks** |

|  |  |
| --- | --- |
| **4.3 Return the name of the movie with the most showings.**  **SELECT** m.movie\_name  **FROM** movie\_info m  **JOIN** showings s  **ON** s.movie\_ID=m.movie\_ID  **ORDER** **BY** **COUNT** (s.Showings\_ID) **DESC**  **LIMIT** 1 | **9 marks** |

**Section 5: JavaScript Challenge [25 marks]**

Create a simple To-Do List web application using HTML, CSS, and JavaScript. The application should have the following features:

1. A text input field for entering tasks.
2. A "Add Task" button to add tasks to the list.
3. A list to display added tasks.
4. Each task should have a checkbox to mark it as complete.
5. A "Delete" button to remove completed tasks.
6. Style the application with CSS to make it visually appealing
   1. Make all the items centered on the page
   2. Display the list below the text input

Provide a clear and organized code for this web application.