



**FOUNDATION ASSESSMENT II MATERIAL RELEASE**

THEORY QUESTIONS

|  |  |
| --- | --- |
| **SECTION** | **MARK** |
| **Theory Questions** | 31 |
| **Concept Questions** | 19 |
| **Python Challenge** | 25 |
| **SQL Challenge** | 25 |
| **TOTAL** | **100** |

**Important notes:**

* This document shares the first section of the Foundation Assessment II which is composed of 9 Theory Questions
* It is worth just under a third of your assessment mark
* You have 24 hours before the assessment to prepare.
* If any plagiarism is found in how you choose to answer a question you will receive a 0 and the instance will be recorded. Consequences will occur if this is a repeated offence. You can remind yourself of the plagiarism policy [here](https://drive.google.com/file/d/1k9UaGOR7hx54QRZ8jvp2jtC4P-8_Rs4F/view?usp=sharing).

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

|  |  |
| --- | --- |
| * 1. **What does SDLC stand for?**   **Ans:** SDLC stands for Software Development Life Cycle. It is mainly a process where high quality software is developed by minimising cost and time in a well-structured way. SDLC has various phases- 1) Planning. 2) Analysis. 3) Decision. 4) Implementation. 5) Testing & Integration. 6) Maintenance.  . Diagram  Description automatically generated  Figure 1: Software Development Life Cycle. | **1 mark** |

|  |  |
| --- | --- |
| * 1. **What exception is thrown when you divide a number by 0?**   **Ans:** When a number is divided by zero, a “ZeroDivisionError” exception is thrown. As, dividing a number by 0 is not defined in maths.  a= **2** b = **0** c = a/b print(c)  The above program throws a ZeroDivisionError. | **1 mark** |

|  |  |
| --- | --- |
| **1.3 What is the git command that moves code from the local repository**  **to the remote repository?**  **Ans:** The git command that moves code from the local repository to the remote repository is called git push. The basic syntax of git push is  **git push remote-name branch-name** where remote-name is the name that the local repo uses for the remote repositories and branch-name is the name of the branch to push to the repository. | 1. **mark** |

|  |  |
| --- | --- |
| **1.4 What does NULL represent in a database?**  Ans: The NULL in a database represents that the value in a filed or column is absent or not known. It cannot be a blank or empty string value. It is not also a zero value for numeric or a field containing space.  Graphical user interface, application  Description automatically generated  Figure 2: Table with NULL values. | **1 mark** |

|  |  |
| --- | --- |
| **1.5 Name 2 responsibilities of the Scrum Master .**  **Ans:** The two responsibilities of the Scrum Master is given below:  1). Facilitating Scrum Process: The Scrum Master oversees enabling various Scrum events, such as the Daily Standup, Sprint Planning Sprint Review, and Sprint Retrospective. The meetings are done efficiently and productively.  2) Making sure that the Scrum practices are followed: The Scrum Master is accountable for confirming that the team is adhering to the Scrum framework and it`s practices. | **2 marks** |

|  |  |
| --- | --- |
| **1.6 Name 2 debugging methods, and when you would use them.**  **Ans: 1) Print Statements**: One of the commonly used and popular debugging method. The method is to insert print statements in the code at points to print outputs, comments and so on. It helps the users and developers to understand what the code is doing and what it is supposed to do.  The print statement can be used for simple debugging issues. It can be used to check if a variable’s value is correctly printing out or the logic is correctly working. Also, to check if a block of code is executed as it should and if a function is being called.  2) **Breakpoints**: Breakpoint is a powerful and widespread debugging method which creates a pausing point in a certain location at the code to examine the state of the code. It can be set using the debugger. It will then allow the developers to analyse the code line by line. There are different types of breakpoints which can be insert into the code like conditional, unconditional, automatic, request and so on.  The breakpoints can be set up in program before it runs, or it is stopped at a particular point in the programme. It is helpful when examining a specific problem which is difficult to find just by going through the code alone. It’s also useful for investigating complex bugs, state of the code at a definite point during runtime. Breakpoints can also be used to understand how the code is being executed. | **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.   |  | | --- | | **def can\_pay(price, cash\_given):**  **if cash\_given >= price:**  **return True**  **else:**  **return False** |   **Ans:** In the above code, the can\_pay is the function which takes two arguments and it checks the condition if the cash\_given is greater than or equals to the price. If it is , then it returns true otherwise it returns false.  **def** can\_pay(price, cash\_given):  **if** cash\_given >= price:  **return True  else**:  **return False** print(can\_pay(**10**,**12**))  This will return True.  This code will throw an error when anything other than numeric values are passed as the arguments like strings, non-numeric values.  **def** can\_pay(price, cash\_given):  **if** cash\_given >= price:  **return True  else**:  **return False** print(can\_pay(**'12'**,**12**))  **TypeError: '>=' not supported between instances of 'int' and 'str'**  This throws a type error as the price is passed as a string value ‘12’ and the cash\_given is an integer.  A try-except block can be used while handling this exception.  **def** can\_pay(price, cash\_given):  **try**:  **if** cash\_given >= price:  **return True  else**:  **return False  except** TypeError:  print(**"Error: price and cash\_given must be numeric values(integer/float)."**)  print(can\_pay(**'12'**,**12**))  **Output:**  **Error: price and cash\_given must be numeric values(integer/float).**  **None**  So, in the above code, the try-except block prints out an error message when a TypeError Occurs when the user enters values other than integer of float data types. | **5 marks** |

|  |  |
| --- | --- |
| **1.8 What is git branching?** Explain how it is used in Git.  Ans: Developers can build a unique history of code changes within a Git repository using the functionality known as "git branching." A branch that has been generated is a snapshot of the codebase at that particular moment. Once that branch is created, developers can work on it without affecting the main codebase, making modifications, adding features, or correcting bugs.  Multiple developers can work on various branches of the same repository at once when branching allows for parallel feature development. When a branch is finished, it can be merged back into the main codebase, including the branch's modifications into the mainline of the codebase.  Git branching comes in handy for a number of tasks, including beta testing new features, addressing issues, experimenting with code modifications, and producing stable releases. Branching enables developers to work in a more organised and productive manner, lowering the possibility of disputes and mistakes by keeping the development of new features separate from the main codebase.  Git offers a variety of methods for managing branches, including branch creation, renaming, merging, and deletion. Understanding branching tactics, merging strategies, and how to resolve conflicts when integrating branches back into the main codebase are necessary for effectively using Git branching.  A picture containing diagram  Description automatically generated  Figure 3: Git Branch and its Operations | **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?   Ans:  The list of key requirements for the restaurant ordering system:   1. Easy -to-use system: The ordering system or interface should be inquisitive to the customers and pose no problem to understand and operate it. They should be able to look at menu items, order food and make payments. 2. Real-time update: Both parties -the customers and the restaurant staffs should be able to see any updates instantaneously to know how long it will take for their order to be ready and delivered. 3. Incorporation between the kitchen staff and delivery team: Working closely with kitchen staff and delivery workers to make sure orders are processed and delivered without any problems. 4. Handling multiple orders: Being able to take care of many orders at the same time and organize them in a line. 5. Inventory Management: By connecting the menu with the inventory system it can be made sure that the things on the menu are available to prepare and ready to be sold to the customers. 6. Custom menu: Being able to fulfill unique and personalized customer needs according to their dietary requirements , religious belief, allergies and so on. 7. Reward System: combining with reward systems and advertising efforts. 8. Secure Payment: The payment method whether online or in-person like cash on delivery should be secure and trusted.   The main consideration of the system:  1.Online payments should be protected and free from harm.  2.The system needs to work all the time so that customers can order things whenever they want without problems.  3.The system should be able to handle more orders when lots of people are ordering at once.  4.The system should be able to work together with other systems like the kitchen staff and delivery workers, and also with inventory management software.  5.The system needs an easy-to-use interface for customers to easily order things.  The system should let customers make changes and have their own special requests.  The components or tools to use:   1. A front end: Customers can use a website to see the menu, order food, and pay online. 2. APIs: An API helps different systems like inventory management and payment processing systems to work together. Also, an API for the geolocation system to track the delivery and to calculate how much time to get the food. 3. A back-end system: A system at the back-end that helps to handle orders and keep track of what they have in stock. It also lets them change the menu and check how much money they're making. 4. A database: A safe place to keep details about customers and orders and other inventories. 5. Encrypted system: A system that keeps customer information and payment details safe using a code. 6. Notification System: A way to send messages to customers and restaurant workers telling them about their orders and how long they have to wait. Also, to promote any new product and give discounts and offers to the customers. | **10 marks** |