**Question 1: Theory Questions**

1. What is the program?

A program is a collection of coded instructions that are run by a computer. They are created using a programming language with the purpose of performing a task when executed.

1. What is the process?

A process refers to the set of instructions being executed as part of a program. The executing of a program triggers a process in which the contained tasks are performed.

1. What is Cache?

Cache is a type of memory contained within the Central Processing Unit (CPU). A frequently used process/instruction can be temporarily held in cache memory which is quicker to access than other types of memory storage (e.g. RAM) since it is local.

1. What is Thread and Multithreading?

A thread is a component within a process that can be executed independent of other code. It is the smallest sequence of programmed instructions that can be programmed in an OS.

Multithreading refers to the action of executing multiple threads in parallel i.e., multiple threads execute a task rather than a single thread. Each thread within a process will contain its own register and own slack memory storing their local variables but will all share the program code and global variables.

1. What is GIL in Python and how does it work?

GIL (or Global Interpreter Lock) is a lock that prevents the Python interpreter from executing multiple threads at a time. This lock restriction means processes are run as single threads, helping to prevent issues of overwriting data, deadlocks etc.

1. What is Concurrency and Parallelism and what are the differences?

Concurrency refers to the execution of multiple tasks at the same time i.e., in an overlapping period, more than one task may be in processing.

Meanwhile, parallelism refers to the processing of one task, though the splitting of these can be processed in parallel across multiple CPU cores/processors (while concurrency is associated with a single CPU).

1. What do these stand for in programming: DRY, KISS, BDUF

DRY- Don't Repeat Yourself- aims to reduce repetition and serves to remind programmers to try to divide code into reusable segments when possible.

KISS- Keep It Simple Stupid- a reminder to keep code simple, concise and as easy to understand as possible to help others gauge its purpose, and to find and debug errors.

BDUF- Big Design Up Front- this principle suggests programmers complete a satisfactory overview design for a website, app, software etc, before the implementation begins.

1. What is Garbage collector? How does it work?

GarbageCollector is a tool used in the automatic deletion of unwanted objects, such as class instances, as a way to free memory space. The garbage collector is accessed in the gc module and has a threshold of objects (which can be changed). When the number of objects exceeds this threshold, a collection process is triggered by the GarbageCollector. While it is typically automatic, it can be disabled or triggered manually through on a time basis (after a fixed time) or on an event basis (e.g., exit application). In reference counting, once references reach 0, garbage collection is triggered.

1. What are ‘deadlock’ and ‘livelock’ in a relational database?

A deadlock occurs when multiple transactions/processes are having to wait for locks to be released upon the resource they require. Neither transaction/process can continue, unless there is an abort or rollback, as they have a lock on the demanded resource while waiting for another locked one.

A livelock occurs when requesting an exclusive lock fails due to overlapping shared locks interfering with each other. The processes change to adapt to each other. While they do not indefinitely wait, like in a deadlock, this does prevent any progress being made.

1. What is Flask and what can we use it for?

Flask is a Python web framework which contains the tools, modules etc allowing a programmer to create a web application (e.g. blogs, commercial websites, social networks, other web pages).

**Question 2:** Discuss the difference between Python 2 and Python 3

Python 2 and Python 3 are two versions of the programming language, with the latter being the more recent, developed version which is more like to receive additional features.

They differ in some ways regarding their functions and syntax, e.g:

Strings- It is not compulsory to use parentheses in Python 2, unlike Python 3

Int division- Integer division in Python 3 can return a float, while Python 2 will return an integer unless the given values are floats.

Unicode- Strings in Python 3 are unicode by default while Python 2 requires the letter 'u' to preface the string.

Input- this function reads inputs as strings in Python 3, while they are considered numbers in Python 2 unless in quote marks.

(Q3 and Q4 in py. file)

**Question 5:** Agile methodology, Scrum: name at least 3 types of meetings that are exercised by Agile teams and describe the objective of each meeting.

One type of meeting is the **daily scrum**- these are short meetings (approx. 15 mins) which are held daily and attended by the scrum master, the product owner and the team. These are held with the objective of acquiring an overview of the team’s progress thus far in the sprint. It ensures the team is updated regarding the scheduled tasks and next steps.

Another meeting is the **sprint review**- these are held at the end of a sprint and are attended by the developers, the product owner, scrum master, and stakeholders. The objectives are to gain an understanding of the scrum team’s accomplishments and setbacks against the sprint goals, which can be gauged through a showcasing of a demo. This also provides the opportunity to gain feedback from the stakeholders, representing the customer’s expectations. Consequently, there will be a collaborative effort to discuss moving forward, which will consider a review of timelines, budgets, markets etc.

A third type of meeting is the **sprint retrospective**- similarly to the sprint review, this meeting takes place after a sprint but it is centered primarily on the scrum team’s methods rather than the product. The methods of working implemented throughout the sprint are discussed and reviewed. The objective is to gain an insight into what the team thought went well, what did not go as well, and what can be done to improve these weaknesses in order to be a more efficient team in a future sprint.

Question 6: Exception handling in Python, explain what each of the following blocks means in the program flow: Try, except, else, finally

Try- this block will contain the initial code to be executed, which will contain the possibility of running into errors. If there are no errors,, the code will run as normal and the desired output will be produced.

Except- this code will be executed when an exception is raised, allowing the error to be handled. If there is still code to be run after this is raised in the try block, it will be skipped and will reach this except block, returning an error to the user notifying them of the issue. E.g. a ValueError or KeyError. .

Else- this code block is positioned after the exception blocks if implemented, and runs when no exceptions have been raised.

Finally- this code will be positioned at the end of the code and will run irrespective of the code run before it in the program flow and their outputs.

**Question 7:** How can we connect a Python program (process) with a database? Explain how it works and how do we fetch / insert data into DB tables from a python program.

This typically requires a connection between Python and an SQL database (DB) which will have already been created. An initial configuration script will have to be run containing details of the local host, a username and password for a programmer’s SQL database. MYSQL connector would have to be installed for this step. Creating queries will also require a cursor to be established. Another script representing the API/application would be required which would contain end point details and the ‘get request’ and ‘put request’- these will be the processes which fetch and insert data into DB tables respectively.

**Question 8**: SQL practical

SELECT DISTINCT a.author\_name, SUM(sold\_copies)

FROM

AUTHORS a

INNER JOIN #return all that match

BOOKS b ON b.book\_name = a.book\_name #share a column in common

GROUP BY a.author\_name #some authors have more than one book

ORDER BY SUM(sold\_copies) DESC #starting from the largest num sold, downwards.

LIMIT 3; #limit it to only 3 rows

(Q9 in py. file)