ASSESSMENT

Python and MySQL
assessment test 2 hours

NO	TASK	POINTS
1	Theory questions	20
2	Concept question	8
3	Coding question	8
4	Coding question	8
5	Concept question	8
6	Concept with practical example	8
7	Concept question	8
8	SQL practical question	10
9	Coding question	22
	TOTAL	**100**

1. **Python theory questions**	**20 points**

1. What is the program?
   A program is a set of ordered instructions that can be executed by a computer. They are written by a human in a programming language. This is one that is readable by humans. They are then converted to machine code (binary) by the compiler or interpreter and then executed by the computer.

2. What is the process?
   A process is an instance of a computer program that is currently being executed. It consists of the program itself, plus all of the resources that it requires to run.

3. What is Cache?
   The cache is a small amount (ranging from several KB to a few MB) of fast memory that sits either within the CPU or between the CPU and the RAM. It holds data that is frequently being required. The CPU will check if the data it needs is in the cache first before checking the RAM.

4. What is Thread and Multithreading?
   A thread is a unit of execution within a process. There can be just a single thread or numerous threads, the latter being know as multithreading. If there are multiple threads, the threads share the resources that are allocated to the process.

5. What is GIL in Python and how does it work?
   GIL stands for Global Interpreter Lock. It is a lock that is put on the Python interpreter that ensures only a single thread is being run at any time. A thread must acquire the GIL in order to be able to run. The reason for having a GIL is that you might have multiple threads running that are accessing the same python objects. If one thread no longer has any references pointing to an object, it might remove that object from memory. But if there is another thread also accessing this object, then it would be a problem if that object was removed from memory! It basically ensures there are no conflicting threads adding and removing things from memory and causing problems.

6. What is Concurrency and Parallelism and what are the differences?
   Concurrency and parallelism refer to how the CPU handles the running of multiple threads. A processor can only run one thread at once. With concurrency, only a single thread is being run at any one time. However, the CPU will run the threads by switching back and forth between them so quickly that it is not noticeable that it is only ever executing one at any one moment in time. If a computer has multiple processors (as most modern machines do these days) then multiple threads can be run truly at once.

7. What do these stand for in programming: DRY, KISS, BDUF
   DRY – don't repeat yourself

Don't write the same code again and again. Write reusable blocks of code (e.g. functions) that can be called upon later with a small amount of code.

KISS – keep it simple stupid
Don't write code that is overly complicated. Write the simplest solution that you can.

BDUF – big design up front
When a whole program is designed and finished before it is presented/deployed. This approach is used most commonly when a team follows a waterfall methodology.

8. What is Garbage collector? How does it work?
   The garbage collector is part of Python's memory management. Python keeps count of the number of references that have been made to an object. When that reference count reaches zero, the garbage collector can remove the object from memory, thereby freeing up space in the memory that can be used to store something else.

9. What are 'deadlock' and 'livelock' in a relational database?
   A deadlock is when a transaction is trying to access data that another transaction holds an exclusive lock to. But then the transaction that is holding that lock is trying to access the table that the first transaction currently holds a lock on. It is like a circle of locks. There can be numerous transactions in the deadlock, not just the two used in this example here. It results in a standstill with no transaction being able to move forward because it can't acquire the lock it needs.

   A livelock is when two (or more) transactions each hold a read lock on a piece of data but want to be able to write to the piece of data that the other is holding a read lock. They can't obtain a write lock while something else has a read lock on it. So again, it comes to a standstill.

10. What is Flask and what can we use it for?
    Flask is a lightweight web framework for Python. This means it contains tools that we can use to make building web applications easier. We can use it for creating web applications such as websites and API endpoints.

2. Discuss the difference between Python 2 and Python 3	8 points

Python 2 is an older version of Python, while Python 3 is the latest one (we are now on Python 3.9 or something like that). Normally each new release of a version of some software would include just minor alterations but the difference between Python 2 and Python 3 was such that the code that was written in Python 2 does not work if you were to try and run it with Python 3. While Python 3 is actively being updated and new libraries are being made and the security of it is being monitored, Python 2 has stopped being actively updated (as of Jan 2020 if I remember right!). There are never going to be any new versions of Python 2. Although lots of programs have by now been ported over to Python 3, there are many that still run on Python 2. As it is still around, it is worth knowing the differences in case you come across these programs. Here are some examples:

- In Python 2 you can print without parentheses, e.g. print 3
  In Python 3 you need parentheses, e.g. print(3)
- In Python 2 the range function returns an entire list
  In Python 3 the range function works like the Python 2 xrange function and returns a generator object, saving memory
- In Python 2 / is floor division
  In Python 2 / is normal division (i.e. returns decimals if required) and // is floor division

3. Write a function that can define whether a word is a Palindrome or not (a word, phrase, or sequence that reads the same backwards as forwards, e.g. *madam*)	**8 points**

```
def is_palindrome(s):
      # check for 1 character strings first
      if len(s) == 1:
            return True
      else:
            s.lower() # make sure all of string is in lower case
            return s == s[::-1]
```

NOTE: can also be done by using indexes to check if the first letter is the same as the last and iterating over ever closer characters to the centre (we did this in Python refresher class!). This other way is more efficient but don't have time right now to think how to write it out!

NOTE 2: file saved as palindrome.py

4. **Write tests for the newly created Palindrome function. Provide a brief explanation for your test case options.**	**8 points**

```
from unittest import TestCase
from palindrome import is_palindrome

class TestIsPalindrome(TestCase):

      # test if/else logic on single character strings
      def test_single_char(self):
            self.assertTrue(is_palindrome('a'))

      # test it correctly identifies a multi-character palindrome
      def test_multi_char(self):
            self.assertTrue(is_palindrome('wow')

      # test is doesn't incorrectly ID a palindrome when it shouldn't
      def test_not_palindrome(self):
            self.assertFalse(is_palindrome('hello')

      # test it correctly IDs a palindrome with different case letters
      def test_palindrome_variable_case(self):
            self.assertTrue(is_palindrome('Wow')
```

5. Agile methodology, Scrum: name at least 3 types of meetings that are exercised by Agile teams and describe the objective of each meeting.	**8 points**

Sprint planning – Happens at the start of the sprint. The objective is to set the goals and decide upon the tasks that will be completed during the sprint.

Daily scrum – The team meets each day (at roughly the same time, probably in the morning). The objective is to brief each other on how their work is going and to flag any problems they are having.

Sprint review – Happens at the end of a sprint. The objective is to describe and present the work that was completed during the sprint.

Sprint retrospective – Happens at the end of a sprint. The team discusses how the sprint went in terms of their efficiency and effectiveness. They discuss how they worked well and how they can improve their working processes.

6. Exception handling in Python, explain what each of the following blocks means in the program flow: Try, except, else, finally	**8 points**

try – The code in this block is run first.

except – The code in this block is run if an exception (i.e. an error) was raised by the code in the try block. The point of this is so that the program can keep running and is not halted abruptly with an exception.

else – This block of code is run if the try block ran successfully and did not raise and exception.

Finally – This block is always run regardless of whether an exception was raised or not.

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.	**8 points**

We need to use a library that can connect Python and the database (DB). Different libraries are available to do this for different DBs. But as we are using MySQL, we use the mysql.connector library. So we first import this library, e.g. import mysql.connector

Once the library is imported, we need to create a connection. This acts as a 'pipeline' between the Python program and the DB. The connection method takes several arguments that are required to configure the connection. These include things like the name of the DB you are connecting to but also things needed for authentication like the username and password. We put these authentication parameters in a config file for security reasons. These are imported at the top of the script, e.g. from config import HOST, USER, PASSWORD. (In the examples provided here I have assigned the connection to the variable cnx.)

Once the connection is established (i.e. the connection object is created), we need to create a cursor. This is like the messenger that travels back and forth along the connection 'pipeline'. The cursor is an object that is created by using the .cursor() method on a connection object, e.g. cur = cnx.cursor()

We can then use the cursor methods to query the DB. The .execute() method acting on a cursor takes a string as an argument. This string contains MySQL statements that retrieve or modify data in the DB. e.g. cur.execute("""SELECT * FROM customers;""")

In order to print out the results, we need to use the .fetchall() method, e.g. cur.fetchall()

When we are done with the DB connection, we close it. We must close both the cursor and the connection, e.g. cur.close(), cnx.close()

8. Given two SQL tables below: authors and books. ● **The authors dataset has 1M+ rows** ● **The books dataset also has 1M+ rows** Create an SQL query that shows the TOP 3 authors who sold the most books in total!	**10 points**

AUTHORS

author_name	book_name
author_1	book_1
author_1	book_2
author_2	book_3
author_2	book_4
author_2	book_5
author_3	book_6

BOOKS

book_name	sold_copies
book_1	1000
book_2	1500
book_3	34000
book_4	29000
book_5	40000
book_6	4400

```
SELECT
      a.author_name,
      SUM(b.sold_copies) AS total_copies_sold
FROM
      AUTHORS AS a
JOIN
      BOOKS AS b ON
      a.book_name = b.book_name
GROUP BY
      a.author_name
ORDER BY
      SUM(b.sold_copies) DESC
LIMIT 3;
```

9. TWO NUMBER SUM: ● Write a function that takes in a non-empty array of distinct integers and an integer representing a target sum. If any two numbers in the input array sum up to the target sum, the function should return them in an array, in any order. If no to numbers sum up to the target sum, the function should return an empty array. ● Note that the target sum has to be obtained by summing two different integers in the array. You cannot add a single integer to itself in order to obtain the target sum. ● You can assume that there will be at most one pair of numbers summing up to the target sum. **Sample Input:** numbers = [3, 5, -4 ,8, 11, 1, -1, 6] target_sum = 10 **Sample Output:** [-1, 11] the numbers can be in any order, it does not matter.	**22 points**

```
def two_number_sum(number_array, target_sum):

      output = []

      for i in number_array:
            if target_sum - i in number_array and target_sum - i != i:
                  output = [i, target_sum - i]

      return output
```