		Thumb drive No: STL
Class/ Index Number /	Centre Number/ 'O' Level Index Number /	Name

新加坡海星中学

MARIS STELLA HIGH SCHOOL
PRELIMINARY EXAMINATION
SECONDARY FOUR

COMPUTING **7155/02**
Paper 2 Practical (Lab-based) **26 August 2022**
2 hours 30 mins

Additional Materials:
Electronic version of LIBRARY.XLSX file
Electronic version of NUMBER.PY file
Electronic version of WORD.PY file
Electronic version of CONVERTER.PY file
Insert Quick Reference for Python

READ THESE INSTRUCTIONS FIRST

Write your class, index number, Centre number, O level index number and name in the spaces at the top of this page.

Answer **all** the questions.

All tasks must be done in the computer laboratory. You are not allowed to bring in or take out any pieces of work or materials on paper or electronic media or in any other form.

Programs are to be written in Python.
Save your work using the file name given in the question as and when necessary.

The number of marks is given in brackets [ ] at the end of each question or part question.
The total number of marks for this paper is 50.

For Examiner's Use
50

This document consists of **10** printed pages.

Task 1

The Chanel Library stores the loan records of its library users.

It uses a spreadsheet software to record the details of the loans.

You are required to finish setting up the spreadsheet to record the loan details for the users of the library.

Open the file **LIBRARY.xlsx**. You will see the following data.

CHANEL LIBRARY LOAN RECORDS

Number	First name	Last name	Book Title	Loan ID	Date Borrowed	Date Returned	Total Days of Loan	Platinum Membership	Overdue Status
1001	Atticus	Tan	Charlotte's Web		4/1/2022	15/1/2022		Yes	
1002	Peter	Lim	Brave New World		5/1/2022	16/1/2022		No	
1003	Immanuel	Lee	Tipping Point		6/1/2022	30/1/2022		Yes	
1004	Hafiz	Salleh	The Great Gatsby		7/1/2022	16/1/2022		Yes	
1005	Donald	Chin	Catch-22		8/1/2022	17/1/2022		No	
1006	Firdaus	Mansoor	Lord of the Rings		9/1/2022	18/1/2022		No	
1007	Jeremiah	Kong	Great Expectations		10/1/2022	19/1/2022		No	
1008	Adrian	Lim	Little Women		6/1/2022	2/2/2022		Yes	
1009	Chad	Lu	Grit		5/1/2022	18/1/2022		No	
1010	Ravi	Kaur	Crime and Punishment		7/1/2022	22/1/2022		No	
1011	Ravi	Kaur	Tipping Point		7/1/2022	19/1/2022		No	
1012	Zenith	Chua	Frankenstein		6/1/2022	19/1/2022		Yes	
1013	Kaylen	Lim	Interceptor		9/1/2022	4/2/2022		No	
1014	Gregory	Ho	Grit		8/1/2022	19/1/2022		No	
1015	Gregory	Ho	Brave New World		8/1/2022	16/2/2022		No	
1016	Tyrone	Lu	Dare to Lead		10/1/2022	28/1/2022		No	
1017	Huzaine	Lee	Catch-22		7/1/2022	27/1/2022		Yes	
								Total Loan Records	
								Total Overdue Records	

Genre

Book Title	Book Genre
Brave New World	FICTION
Catch-22	NON-FICTION
Charlotte's Web	CHILDREN
Crime and Punishme	THRILLER
Dare to Lead	NON-FICTION
Frankenstein	CLASSICS
Great Expectations	CLASSICS
Grit	NON-FICTION
Interceptor	THRILLER
Little Women	CLASSICS
Lord of the Rings	CLASSICS
The Great Gatsby	CLASSICS
Tipping Point	NON-FICTION

Save the file as **LOAN_<class>_<index number>_<your name>.xlsx**

1 In cell **E5**, enter a formula that uses an appropriate function to search for the **Book Genre** in the **Genre** table. The Loan ID comprises the first six characters of the **Book Genre** and all characters of the **Number** field. You may use the symbol **&** to join them. For example, ="ABCDEF" & "1001" displays the text as ABCDEF1001.

The Loan ID of the first record is CHILDR1001. Use it to calculate the **Loan ID** in column E. [3]

2 In cells **H5** to **H21**, enter a formula to find the total days of the loan using the **Date Borrowed** and **Date Returned** fields. [1]

3 In cells **J5** to **J21**, enter a conditional statement to identify if the loan is overdue. If the total days of loan exceeds 21 days, display the text “Overdue”. Otherwise, display the text “Not Overdue”. [2]

4 In cell **J22**, enter a function to calculate the total number of loan records.

[1]

5 In cell **J23**, enter a function to calculate the total number of overdue records.

[1]

6 In cells **B5** to **B21**, use a conditional formatting tool to make the cell yellow in the rows that contain platinum members with overdue records. These members need not pay a fine due to their platinum membership.

Save and close your file. [2]

Task 2

The following program allows 10 integers to be entered. The numbers must be within 1 to 100 inclusive, before being classified and printed as Low, Medium or High.

```
for count in range(10):
    number = int(input('Enter a positive integer: '))

    if number >= 1 and number <= 40:
        print(number, '- Low')
    elif number > 40 and number <= 70:
        print(number, '- Medium')
    elif number > 70 and number <= 100:
        print(number, '- High')
```

Open the file **NUMBER.py**

Save the file as **NUMBER1_<class>_<index number>_<your name>.py**

7 Edit the program so that it will accept any number of inputs. Assume that only digits will be entered. [2]

8 Edit the program to test whether the user has entered a number in the correct range. Output a suitable error message that asks the user to enter the number again, and repeat this until the user enters a valid number. [2]

9 Edit the program to store numbers that are classified as high in a list. Each number must be stored in the next available element in the list.

Output the list after all the numbers have been entered.

Save your program. [3]

10 Save your program as **NUMBER2_<class>_<index number>_<your name>.py**

Edit the program to find the average of the total numbers entered.

Output the average.

Save your program. [3]

Task 3

A program asks the user to enter a word. The program converts the word entered to lower case. The program checks if the word entered is valid or invalid.

A word is invalid if it:

- begins with the letter 'p'; or
- ends with the letter 'h'; or
- contains the letter 'e'.

The program displays a suitable message to indicate why the word is invalid. This will be for the first error that is found and not for all errors. For example, if the word 'phish' is entered, the error reported is that it begins with the letter 'p'.

If the word is valid, the program categorises the word as:

- **short** if the word is 3 characters or less in length
- **medium** if the word is 4 to 10 characters in length
- **long** if the word is more than 10 characters in length.

The program displays a suitable message to show the category of the word.

There are several syntax errors and logic errors in the program.

```
word = input("Please enter your word: ")
word = word.islower()
begin_p = word.beginswith("p")
end_h = word.endswith("h")
contain_e = word("e")
word_length = word.length()
if not begin_p and not end_h and contain_e = -1:
    if word_length < 3:
        print("The length of the word is short.")
    elif word <= 10:
        print("The length of the word is medium.")
    elif:
        print("The length of the word is long.")
if begin_p:
    print("Invalid. You entered a word that begins with 'p'.")
elif end_h:
    print("Invalid. You entered a word that ends with 'h'.")
elif not contain_e:
    print("Invalid. You entered a word that contains 'a'.")
```

Open the file **WORD.py**

Save the file as **MYWORD_<class>_<index number>_<your name>.py**

11 Identify **and** correct the errors in the program so that it works according to the requirements given.

Save your program. [10]

Task 4

You have been asked to create a network address generator program. This program will create IPv4, IPv6 and M.A.C. addresses using user-defined functions.

Open the file **CONVERTER.py.**

You will see the following function which converts a denary number into a binary number.

```
import random
def denary_to_bin(decimal):
   conversion_table = ['0', '1']
   binary = ''

   while decimal>0:
        remainder = decimal %2
        binary = conversion_table[remainder]+ binary
        decimal = decimal //2

   return binary
```

Sample executions:

```
>>> denary_to_bin(0)
0

>>> denary_to_bin(44)
101100

>>> denary_to_bin(255)
11111111
```

12 Save your program as
HEXADECIMAL_<class>_<index_number>_<your name>.py

Write a new function called **`denary_to_hex(number)`** to convert a denary number into a hexadecimal number. You may use **`denary_to_bin(number)`** function as reference. This function should take in an integer as an argument and return a hexadecimal number. No validation is required.

```
def denary_to_hex(number):
    pass
```

Sample executions:

```
>>> denary_to_hex(15)
F

>>> denary_to_hex(22511)
57EF
```

[4]

13 Save your program as **RANDOM_<class>_<index_number>_<your name>.py**

Write a new function called **`randomnumber(start, end)`** to create a random number using the **`randint`** function. This function should take in two integers as arguments, of which both numbers are inclusive, and return an integer. No validation is required.

```
def randomnumber(start, end):
    pass
```

Sample executions:

```
>>> randomnumber(0, 255)
115

>>> randomnumber(0, 65535)
17326
```

[2]

14 Save your program as
IPV4_<class>_<index_number>_<your name>.py

Use the **`randomnumber`** function from part 13 to write a function called **`createIPv4().`** This function should return an IPv4 address, which consists of 4 bytes. Each byte is a randomly generated number between 0 and 255, both inclusive. There is a dot separating two numbers.

```
def createIPv4():
    pass
```

The format of an IPv4 address is as follows:

175.182.38.197

There are no arguments in this function.

[2]

15 Save your program as
STORE_<class>_<index_number>_<your name>.py

Use the **`createIPv4`** function from part 14 to write a function called **`storeIPv4(n).`** This function should return a list of IPv4 addresses based on n integers.

```
def storeIPv4(n):
    pass
```

Sample executions:

```
>> storeIPv4(3)
New IPv4 addresses have been created: ['202.127.14.231',
'211.78.153.201', '193.254.252.16']

>> storeIPv4(5)
New IPv4 addresses have been created: ['120.241.187.1', '5.76.0.143',
'219.82.192.200', '229.124.114.193', '188.55.89.62']
```

[3]

16 Save your program as **IPV6_<class>_<index_number>_<your name>.py**

Use the **denary_to_hex** and **randomnumber** functions from earlier parts to write a function called **createIPv6().** This function should return an IPv6 address, which consists of 8 segments of 2 bytes. Each segment corresponds to a randomly generated number between 0 and 65535, both inclusive. There is a colon separating two numbers.

```
def createIPv6():
    pass
```

The format of an IPv6 address is as follows:

4F6E:57EF:51DD:994B:DB36:1D43:E418:5232

There are no arguments in this function.

[3]

17 Save your program as **MAC_<class>_<index_number>_<your name>.py**

Use the **denary_to_hex** and **randomnumber** functions from earlier parts to write a function called **createMAC().** This function should return a MAC address, which consists of 6 bytes of data separated using colons. Each byte corresponds to a randomly generated number between 0 and 255, both inclusive.

```
def createMAC():
    pass
```

The format of a MAC address is as follows:
2C:58:C9:81:C8:8D

[2]

18 Save your program as **NETWORK_<class>_<index_number>_<your name>.py**

Extend your program by creating a simple user interface.

The program should:

- Print the 3 options available to the user with one option per line.
    - Option 1: Create an IPv4 address
    - Option 2: Create an IPv6 address
    - Option 3: Create a MAC address
- Request user to select option 1 or 2 or 3.
- Request user to re-input if the option is not 1 or 2 or 3.
- Output the network address created by calling the functions created earlier.

Sample execution 1:

```
Option 1: Create an IPv4 address
Option 2: Create an IPv6 address
Option 3: Create a MAC address
Please select options 1/2/3: 1
74.252.155.209
```

Sample execution 2:

```
Option 1: Create an IPv4 address
Option 2: Create an IPv6 address
Option 3: Create a MAC address
Please select options 1/2/3: 3
F8:6B:12:30:82:14
```

Sample execution 3:

```
Option 1: Create an IPv4 address
Option 2: Create an IPv6 address
Option 3: Create a MAC address
Please select options 1/2/3: 4
Re-enter options 1/2/3: 2
6918:6A60:E094:BA9A:D683:3933:BFE1:9140
```

[4]

-End of Paper-