Thumb drive No:	

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 (Lab-based) **29 Aug 2019**
2 hours 30 minutes

Additional Materials:
Electronic version of BANK.XLSX data file
Electronic version of WEIGHT.PY file
Electronic version of RIDE.PY file
Insert Quick Reference Glossary

READ THESE INSTRUCTIONS FIRST

Answer **all** 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 **6** printed pages.

Task 1

Maybey Bank uses spreadsheet software to record its customers' loan records. You are required to finish setting up the spreadsheet to record the details.

Open the file **BANK.XLSX**. You will see the following data.

MAYBEY BANK CUSTOMERS LOAN RECORDS

Customer Code	Account Type	Application Method	Loan Amount	Number of Years in Loan Plan	Interest Rate per Year	Total Amount of Payout	Free $70 Cash-back
S12930482		Offline	$ 35,000.00				
S84012493		Online	$ 78,400.00				
S95093284		Offline	$ 55,730.00				
C09842304		Offline	$ 48,000.00				
C73842394		Online	$ 53,500.00				
C90077234		Offline	$ 43,620.00				
I83482034		Online	$ 48,600.00				
I98203429		Online	$ 82,570.00				
I92349434		Online	$ 53,850.00				
T23490823		Offline	$ 6,304.00				
T92304343		Online	$ 85,012.00				
T09240234		Offline	$ 8,921.00				

Interest Rate Table

Number of Years in Loan Plan	Interest Rate per Annum
5	3.30%
7	3.50%
10	4%
15	7%

Loan Plan Term Table

Account Type	S	C	I	T
Number of Years	10	5	7	15

Save the file as **LOAN**_<your name>_<class>_<index number>

1 For the cell range **B5:B16**, use an appropriate function to extract the first letter of the **Customer Code** to represent the **Account Type**.

[1]

2 Use an appropriate function to search for the **Number of Years** in the **Loan Plan Term Table** given and use it to complete the **Number of Years in Loan Plan** column.

[2]

3 Use an appropriate function to search for the **Interest Rate per Annum** in the **Interest Rate Table** given and use it to complete the **Interest Rate per Year** column.

[2]

4 Enter a formula to calculate the total amount of money each customer will have to pay at the end of their loan plan period. This is based on the **Number of Years in Loan Plan**, **Interest Rate per Year** compounded **monthly** and **Loan Amount**. Use that value to complete the **Total Amount of Pay** column

[3]

5 Use a conditional statement, to identify those customers who have taken a loan amount of more than $45000 and whose application method was online, and put **YES** in the **Free $70 Cash-back column**. Otherwise, put **NO** in the **Free $70 Cash-back column**.

[2]

Save and close your file.

Task 2

The following program accepts the weight in kg for 15 students and prints out the highest weight and the average weight of the students. The weights are in the range of 20kg to 140kg (inclusive).

```
heaviest = 20
total_weight = 0
no_of_students = 15

for student in range(no_of_students):
    weight = float(input("Enter the weight of student in kg: "))
    if heaviest < weight:
        heaviest = weight
    total_weight = total_weight + weight
average_weight = total_weight/no_of_students

print("Heaviest weight of the students is ", heaviest)
print("Average weight of the students is ", average_weight)
```

Open the file **WEIGHT.py**

Save the file as **CLASSWEIGHT**_<your name>_<class>_<index number>

6 Edit the program so that it:

(a) Accepts weight for 36 students. [1]

(b) Prints out the lightest weight as well as the heaviest weight. [4]

(c) Test if the weight input is between 20kg and 140kg (inclusive), and if not, asks the user for input again as necessary. [3]

Save your program.

7 Save your program as **VARWEIGHT**_<your name>_<class>_<index number>

Edit your program so that it works for any number of students. [2]

Save your program.

Task 3

The following program is used to check if a customer is allowed on the rollercoaster ride based on age and height. The program is created to check based on the following rules:

- It accepts up to a maximum of 20 customer entries or the checker inputs a "0,0".
- All entry data are in the format: "age,height(in cm)" e.g 14,174
- Customers below 5 years old or above 55 years old are not allowed entry
- Customers below the height of 130 or above the height of 180 are not allowed entry

You can assume that all entries will be in the format age,height(in cm)

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

```
customer = 1
data = input (("Enter details of customers: ")
date=data.split(",")
age = int(data[1])
height = int(data[0])


while customer != 20 or not(age != 0 and height != 0):

    if age<5 or age>55 or height<130 or height>=180:
        print("Entry denied, you do not meet the requirements")

    else:
        print("Entry allowed. Please enjoy your ride.")
        customer = customer + 1

    if customer == 5:
        break

    data = input ("Enter details of next customers: ")
    date=data.split(",")
    age = int(data[0])
    height = int(data[1])

    print("Please check safety belts of customers and get ready to start ride."
```

Open the file **RIDE.py**

Save the file as **ROLLERRIDE**_<your name>_<class>_<index number>

8 Identify and correct the errors in the program so that it works correctly according to the rules above.

[10]

Save your program.

Task 4

You have been asked to write a program to verify the check digit of the Singapore NRIC number. The last letter at the end of the NRIC number is called a checksum. The checksum allows us to check if the NRIC number has been entered correctly.

The algorithm for generating the checksum is as follows:

- Multiply the seven digits against a given weight [2,7,6,5,4,3,2].
  For NRIC number S1234567D, the sum is

(1 x 2) + (2 x 7) + (3 x 6) + (4 x 5) + (5 x 4) + (6 x 3) + (7 x 2) = 106

- If the first letter of the NRIC starts with T or G, add 4 to the total.
- Then you divide the number by 11 and get the remainder.
  106/11 = 9 remainder 7
- The letter you get depends on the first letter in the IC using the code below:

First letter is either S or T											
Remainder	0	1	2	3	4	5	6	7	8	9	10
Equivalent Letter	J	Z	I	H	G	F	E	D	C	B	A

First letter is either F or G											
Remainder	0	1	2	3	4	5	6	7	8	9	10
Equivalent Letter	X	W	U	T	R	Q	P	N	M	L	K

The checksum D for S1234567D is correct.

The program should allow you to:

- Enter an NRIC number in the format L0000000L.
  (L represents a letter, 0 represents a number)
- Ensure that only characters in the correct format is accepted.
- Store the weights as a constant in an array.
- Calculate the weighted sum of the NRIC number.
- Calculate the remainder of the weighted sum.
- Calculate the checksum and verify if the NRIC entered is correct.
- Display the output on the screen. Your output **must** look like this:

```
The NRIC no. to check is:       T4235285Z
Weighted sum is:                111
Calculated checksum:            Z
The NRIC number is correct.

The NRIC no. to check is:       S4232265Z
Weighted sum is:                86
Calculated checksum:            B
The NRIC number is incorrect.
```

9 Write your program and test that it works. [10]

Save your program as **MYNRIC**_<your name>_<class>_<index number>

10 When your program is working, use the following test data to show your test results:

S9374728J
T0005923J
G7730493R

Take a screen shot of your results and save it as a jpeg.
NRICRESULT_<your name>_<class>_<index number> [5]

11 Save your program as **NRICCON**_<your name>_<class>_<index number>
Extend your program to continuously check the numbers until "q" is entered

Save your program. [2]

12 Save your program as **NRICCHANGE**_<your name>_<class>_<index number>
Extend your program to allow the user to enter values for new weights(in the event Singapore decides to change the formula to obtain a new checksum) and calculate the checksum based on the new weights.

The user should be asked to key in weights in the format [x,x,x,x,x,x,x] (should only be 7 digits) then ask for the NRIC number to be checked. You can assume that the values used for weights will be single digits only. [3]

Save your program

-End of Paper-