Bukit View Secondary School

Secondary Four Express
Preliminary Examination 2022

CANDIDATE NAME			
CLASS		INDEX NUMBER	

COMPUTING **7155/2**

Paper 2 (Lab-based)

13 September 2022
2 hour 30 minutes

Additional Materials: Electronic version of `CAFE.xlsx` data file
Electronic version of `ENCRYPT.py` file
Electronic version of `TIME.py` file
Insert Quick Reference Glossary

READ THESE INSTRUCTIONS FIRST

Write your candidate name, class and index number in the spaces provided at the top of this page.
Write in dark blue or black pen.

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
Marks
50

Setter: Mrs Leong Wee Choo **Parent's Signature: ________________**

This question paper consists of **9** printed pages, **1** blank page and **1** insert.

Task 1

Amy runs a cafe. Due to the increase in prices of ingredients and utilities, she has to adjust the prices of her food items.

To help her decide on the percentage of increase, she decides to use a spreadsheet to keep track of her sales from Friday to Sunday.

You are required to finish setting up the spreadsheet for the required details of her business.

Open the file **CAFE.xlsx**. You will see the following data. Cells **C4** to **E12** show the number of each item sold from Friday to Sunday.

	A	B	C	D	E	F	G	H
1	**BV Cafe**							
2								
3	**Food items**	**Price**	**Friday**	**Saturday**	**Sunday**	**Average sold**	**Status**	**New Price**
4	Hot chocolate	$ 4.50	35	34	45			
5	Cappuccino	$ 4.30	22	18	32			
6	Latte	$ 3.90	13	15	10			
7	Tuna Sandwich	$ 9.50	32	37	23			
8	BBQ Chicken	$ 9.90	26	25	45			
9	Seafood pasta	$ 13.90	42	23	43			
10	Fish N Chip	$ 12.90	27	19	30			
11	Aglio Olio	$ 12.50	31	29	42			
12	Chicken Cutlet	$ 12.00	26	38	25			
13								
14	Range of average sold							
15								
16	**Percentage Increase**							
17	**Average sold**	0	15	25	30			
18	**Description**	0 up to but not including 15	15 up to but not including 25	25 up to but not including 30	30 and above			
19	**Increment (%)**	0	2	5	10			

Save the file as **MYCAFE_2022**_<your name>_<index number>**.xlsx**

1 In the **Average sold** column, enter a formula that uses a function to calculate the average number of items sold. The average number must be rounded off to the nearest integer. [2]

2 In cell **C14**, enter a formula to calculate the range of average sold. [1]

3 In the **Status** column, enter a formula that uses functions to find the top two food items based on average sold and display the text **Favourite** for these food items. The other cells in the column must be empty. [3]

4 Use an appropriate function to search for the **Increment (%)** in the **Percentage Increase** table, and use it to complete the **New Price** column. [2]

5 In cells **A4** to **H12**, use a conditional formatting tool to fill the row blue if the first character of the food item is greater than "P". [2]

Save and close your file.

Task 2

The following program reads in a message and prints out the encrypted message by replacing each character with the character 3 positions behind it based on the ASCII table. You can assume the message is in lower case.

```
message = input("Enter a message: ")
encrypted_m = ""
for i in range(len(message)):
    encrypted_m = encrypted_m + chr(ord(message[i])+3)
print("Encrypted message:", encrypted_m)
```

Open the file **ENCRYPT.py**

Save the file as **MYENCRYPT_2022_**<your name>_<index number>**.py**

6 Edit the program to implement the following changes to the encryption algorithm:

- replace lower case vowel ("a", "e", "i", "o", "u") with the same vowel in upper case instead
- replace space with "&" instead [4]

Save your program.

7 Write a function `decrypt(s)` which takes in the encrypted message and returns the original message. [5]

Call the function to take in the encrypted message and print out the original message. [1]

Save your program.

Task 3

The program on the next page asks the user to enter time in 24-hour format (HHMM), validates the input and outputs the time in 12-hour format.

The program displays a suitable message to indicate why the input is invalid and repeatedly takes in input until the user enters a valid time.

A time is invalid if it:

- does not contain 4 digits only
- hour (HH) is not within the range 0 to 23 inclusive
- minute (MM) is not within the range 0 to 59 inclusive

The program calls the function `correct_range()` to validate hour and minute.

Function `correct_range(num, limit)` takes in two integers as parameters. It returns a boolean value `True` if `num` is within the range of 0 and `limit` (inclusive); otherwise it returns `False`.

If the time is valid, the program prints the time in 12-hour format.

Some correct sample output is shown here:

- `12:00AM` if the input is `0000`
- `9:36AM` if the input is `0936`
- `12:00PM` if the input is `1200`
- `11:33PM` if the input is `2333`

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

```
def correct_range(num, limit):
    return 0 <= num < limit

print("Time in 24-hour format should be entered as 4 digits")
valid = False
while valid == True:
    time = input("Enter a time in 24-hour format: ")
    if len(time) = 4:
        print("Length of input must be 4")
    elif time.hasdigit() == False:
        print("Input must be digits only")
    else:
        hour_int = int(time[:2])
        minute_str = time[2:]
        if not correct_range(hour_int):
            print("Hour must be in the range 0 to 23 inclusive")
        elif correct_range(int(minute_str), 59):
            print("Minute must be in the range 0 to 59 inclusive")
        else:
            valid = False
if 0 <= minute_str <= 11:
    indicator = "AM"
else:
    indicator = "PM"
new_hour = hour_int
if hour_int == 0:
    new_hour = hour_int + 12
elif hour_int >= 13:
    new_hour = hour_int + 12
converted_time = new_hour + ":" + minute_str + indicator
print(converted_time)
```

Open the file **TIME.py**

Save the file as **MYTIME_2022_**<your name>_<index number>**.py**

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

Save your program. [10]

Task 4

You have been asked to write a program for a teacher to enter the students' names and their respective test marks. The test marks are whole numbers only.

The program must:

- allow the user to enter student's name and mark and store the input in 2 different lists. The program does not need to validate student's name. However, it must ask for another mark each time the user enters a mark that is not between 0 and 100 inclusive.
- ask the user if there are any more students' details to be input. The program does not need to validate this input.
- calculate the average mark.

When all the input has been entered, the program must display:

- `"There are [insert student count] students"`.
  The `[insert student count]` must be replaced with the number of students entered.
- `"The average mark is [insert average]"`.
  The `[insert average]` must be replaced with the average mark, rounded to 1 decimal place.

The program must include appropriate input and output messages.

9 Write a program to meet the requirements. Save your program as **MARKS_2022_**<your name>_<index number>**.py** [9]

10 When your program is working, use a set of test data covering normal, error and boundary conditions to show your result.

Take a screenshot of your result and save it as:
MARKSTEST_2022_<your name>_<index number>

Save your file in either **.jpg** or **.png** format. [3]

11 Save your program as **REMEDIAL_2022 _**<your name>_<index number>**.py**

A student who scored lower than the average mark (before rounding) will need to attend remedial class. The teacher wants to arrange these students into groups of three. If the number of students is not a multiple of 3, the last group will have fewer than 3 students. Each group will then be assigned a peer tutor.

Extend your program to:

- store the names of students who need to attend remedial class in a list
- include an appropriate output message and display the names of these students. If no students have been identified, display `"There are no students for remedial".`
- calculate and display with appropriate output message the number of peer tutors needed if there are students identified for remedial.

Save your program. [8]

- End of Paper -

BLANK PAGE