SERANGOON SECONDARY SCHOOL
PRELIMINARY EXAMINATION
SECONDARY 4 EXPRESS

CANDIDATE NAME	( )	CLASS	
CENTRE NUMBER	S	INDEX NUMBER	

COMPUTING

7155/02
Paper 2
29 August 2022
2 hours 30 minutes

Additional Material: Quick Reference for Python

Setter(s): Mr Adrial Tan

READ THESE INSTRUCTIONS FIRST

Write your name, class and index number on all the work you hand in.
Write in dark blue or black pen.

All tasks must be done in the computer laboratory.
Programs are to be written in Python. A Quick Reference for Python is provided.

Approved calculators are allowed.

Answer **all** questions.

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.

Retrieve the template files from the COMPEXAM folder.

Save your work inside the COMPEXAM folder using the file names given in the question as and when necessary.

For Examiner's Use
50

Task 1

Serangoon Secondary School's basketball team has several members.

The teacher-in-charge uses spreadsheet software to record the details of the members.

You are required to finish setting up the spreadsheet to record the details for the members in the team.

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

	A	B	C	D	E
1	**SSS Basketball Team 2022**				
2					
3	**Name**	**Class**	**Age**	**Height (m)**	**Remarks**
4	Jeremiah	2H1	14	1.58	
5	Anuar	3I3	15	1.75	
6	Wei Jie	2H1	14	1.60	
7	Aiden	4N3	16	1.82	
8	Jordan	2H4	14	1.62	
9	Michael	1SD8	13	1.42	
10	Jun Kai	4N2	17	1.75	
11	Brendon	4N8	16	1.72	
12	Zhafir	4N3	16	1.79	
13	Arun	1SD6	13	1.45	
14	Jenson	4N4	16	1.75	
15	Anston	3I5	15	1.69	
16	Sanjay	4N6	16	1.78	
17	Nizam	4N1	16	1.74	
18	Nicholas	1SD9	13	1.45	
19					
20	**Total members:**				
21	**Most common height:**				
22	**Average age:**				

Save the file as **SSSBBMEMBERS_2022**_<Class>_<Index_Number>_<Your_Name>**.xlsx**

1 In cell **B20** enter a formula that uses a function to show how many members are there in the basketball team. [1]

2 In cell **B21** enter a formula that uses a function to find the height of the members that appear the highest number of times. [1]

3 In cell **B22** enter a formula that uses functions to find the average age of the members. The average age must be rounded up to the nearest integer. [2]

4 In the **Remarks** column enter a formula that uses functions to:

- find the tallest member and display the text **Captain** for that member
- find the second tallest member and display the text **Vice Captain** for that member

All other cells in the column must display the text **Member**. [4]

5 In cells **A4** to **E18** use a conditional formatting tool to make the text red in the row that contains the shortest member. [2]

Save and close your file.

Task 2 begins on the next page.

Task 2

The minimum height a rider can be to ride on a pony at a zoo attraction is 1.4 metres.

The following program checks whether a rider meets the minimum height requirement.

The program allows the user to enter the names and heights of 10 riders. It will display a message after each rider's details are input, to indicate if the rider is tall enough or not.

```
num_rider = 10
min_height = 1.4
for p in range(num_rider):
    name = input("Enter name of rider:  ")
    height = float(input("Enter height of rider: "))
    if height >= min_height:
        print("Rider is tall enough to ride a pony.")
    else:
        print("Rider is not tall enough to ride a pony.")
```

Open the file **CHECKHEIGHT.py**

Save the file as **PONYCHECKHEIGHT_2022_**<Class>_<Index_Number>_<Your_Name>**.py**

6 Edit your program so that it will work for any number of riders. The program must display a suitable input message.

Save your program. [1]

7 If a child shorter than 1.4 m wants to take the pony ride, an adult needs to accompany the child. All children shorter than 1 m will not be allowed to take the pony ride.

Edit the program so that it checks if a child taller than 1 m but shorter than 1.4 m has an adult accompanying him/her to ride the pony.

The program must display the following messages:

- `"The rider is tall enough to ride a pony."` if the rider's height is at least 1.4 m.
- `"The rider is not tall enough to ride a pony."` if the rider's height is shorter than 1 m.
- `"Is the rider accompanied by an adult? (Y/N)"` and gets an input from the user if the rider's height is at least 1 m but shorter than 1.4 m.
- `"The rider is not tall enough but is accompanied by an adult to ride the pony."` if the rider's height is taller than 1 m but shorter than 1.4 m, and is accompanied by an adult.

- `"The rider is not tall enough and is not accompanied by an adult to ride a pony."` if the rider's height is taller than 1 m but shorter than 1.4 m, and is not accompanied by an adult.

Save your program. [5]

8 Save your program as **CHECKADULT_2022**_<Class>_<Index_Number>_<Your_Name>**.py**

Edit the program so that it stores the name of the riders in a list, only if the rider is tall enough or is accompanied by an adult. Each name must be stored in the next available element in the list.

Output the list after all riders have been entered.

Save your program
. [4]

Task 3 begins on the next page.

Task 3

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

A word is invalid if it:

- contains characters other than letters
- does not start with a capital letter

The program displays a 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 'python' is entered, the error reported is that it does not begin with a capital letter.

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 8 characters in length
- **long** if the word is more than 8 characters in length.

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

```
word = input("Please enter your word: ")
word = word.lower()
begin_cap = word.isupper()
has_symbols = not word.isalnum()
has_digits = True
for c in words:
   if c.digit():
      has_digits = True
      break
word_len = word.length()
if not begin_cap and not has_symbols and not has_digits:
   if word_len < 3:
      print("You entered a short word.")
   elif word_len <= 8:
      print("You entered a medium word.")
   elif:
      print("You entered a long word.")

if begin_cap:
   print("Error. You entered a word that does not start with a capital letter.")
```

```
elif has_symbols:
   print("Error. You entered a word that contain special characters.")
elif has_digits:
   print("Error. You entered a word that contain digits.")
```

Open the file **WORDCHECK.py**

Save the file as **MYWORDCHECK_2022**_<Class>_<Index_Number>_<Your_Name>**.py**

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

Save your program. [10]

Task 4 begins on the next page.

Task 4

A cipher is an algorithm that encodes a message so that only the intended recipient is able to read it. The Caesar Shift Cipher is a type of cipher where each letter in the original message is shifted down the alphabet sequence by a fixed number of positions.

The table below shows an example where the letters have been shifted down the alphabet sequence by 5 positions. The letters "wrap-around" at the end. Using the table, the message 'Computing' would be encoded as 'Htruzynsl'.

Original	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
Shifted down by 5	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e
Original	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
Shifted down by 5	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E

The process of encoding the message is called encryption and the encoded message is called the ciphertext. In order to read it, the recipients need to perform an opposite process called decryption to change the ciphertext back to the original message.

Open the file **CIPHER.py**. You will see the following function which shifts a letter down the alphabet sequence by one position (i.e. a → b → … y → z → a, and A → B → … Y → Z → A) and does nothing to other characters.

```
def shift_down(c):
    # Shifts letter down by one position:
    # i.e. a → b → ⋯ y → z → a, A → B → ⋯ Y → Z → A
    # Does nothing to other characters.
    if c == "z":
        return "a"
    elif c == "Z":
        return "A"
    elif c.isalpha():
        return chr(ord(c)+1)
    else:
        return c
```

Sample executions:

```
>>>   shift_down('a')
b
>>>   shift_down('z')
a
>>>   shift_down('B')
C
>>>   shift_down('2')
2
>>>   shift_down('!')
!
```

10 Use the `shift_down(c)` function provided to write a function `encrypt(message, pos)`, which encrypts the `message` argument by shifting all the letters in the message down the alphabet sequence by the number of positions given in the `pos` argument. The function should ignore all other characters. [7]

Save your program as **ENCRYPT_2022**_<Class>_<Index_Number>_<Your_Name>**.py**

Sample executions:

```
>>>  encrypt('Python!', 0)
'Python!'
>>>  encrypt('Python!', 2)
'Ravjqp!'
>>>  encrypt('Python!', 30)
'Tcxlsr!'
```

11 Save your program as **SHIFTUP_2022**_<Class>_<Index_Number>_<Your_Name>**.py**

In the same program, write a function `shift_up(c)`, which shifts a letter up the alphabet sequence by one position (i.e. z → y → … b → a → z, and Z → Y → … B → A → Z) and does nothing to other characters. [3]

Sample executions:

```
>>>  shift_up('a')
z
>>>  shift_up('z')
y
>>>  shift_up('B')
A
>>>  shift_up('2')
2
>>>  shift_up('!')
!
```

Save your program

12 Save your program as **DECRYPT_2022**_<Class>_<Index_Number>_<Your_Name>**.py**

Use the `shift_up(c)` function to write a function `decrypt(ciphertext, pos)` which decrypts the `ciphertext` argument by shifting all the letters up the alphabet sequence by the number of positions given in the `pos` argument. The function should ignore all other characters. [4]

Sample executions:

```
>>>  decrypt('Mjqqt Btwqi!',5)
'Hello World!'
>>>  decrypt ('Gsqtyxmrk mw jyr!',30)
'Computing is fun!'
```

Save your program.

13 Save your program as **ENCRYPTBOT_2022_**<Class>_<Index_Number>_<Your_Name>**.py**

Extend your program by creating a simple user interface. The program should:

- request user to enter `'E'` to encrypt a message or `'D'` to decrypt a ciphertext (case insensitive). Prints an error message and request user to re-enter if the input is not `'E'`, `'e'`, `'D'` or `'d'`.
- request user to enter the message or the ciphertext.
- request user to enter number of positions to shift the letters. Prints an error message and request user to re-enter if input is not a positive integer. Store this input.
- output the encrypted message or decrypted ciphertext. [6]

Sample execution 1:

```
Please enter 'E' to encrypt or 'D' to decrypt: e

Please enter message: Python!

Please enter number of positions: 2

The encrypted text is 'Ravjqp!'
```

Sample execution 2:

```
Please enter 'E' to encrypt or 'D' to decrypt: D

Please enter ciphertext: Gsqtyxmrk mw jyr!

Please enter number of positions: 30

The decrypted ciphertext is 'Computing is fun!'
```

Sample execution 3:

```
Please enter 'E' to encrypt or 'D' to decrypt: X

Error! Please enter 'E' to encrypt or 'D' to decrypt: D

Please enter ciphertext: Mjqqt Btwqi!

Please enter number of positions: n

Error! Please enter number of positions: 5

The decrypted ciphertext is 'Hello World!'
```

Save your program.

END OF PAPER