Name: ..... Register no: ..... Class: .....

NGEE ANN SECONDARY SCHOOL

O

PRELIMINARY EXAMINATION

COMPUTING **7155/02**

Paper 2 (Lab-based) **30 August 2019**

2 hr 30 min

Additional materials:
Electronic version of CARLOAN.py file
Electronic version of VENDING.py file
Electronic version of WORDS.py file
Electronic version of PARK.py file
Insert Quick Reference Glossary

Instructions to Candidates

Write your name, register number and class at the top of this page.

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

Checked by student: _____ Date: _____

This document consists of 9 printed pages and 1 blank page.

Task 1

A car dealer company uses a spreadsheet to calculate the car loans and payment details of its customers. The spreadsheet provides the loan amount and annual interest rates tagged to different loan periods. You are required to finish setting up the spreadsheet to calculate the monthly payment amounts of each customer.

Open the file **CARLOAN.xlsx**. You will see the following data.
Save the file as **LOANPAYMENT**_<your name>_<centre number>_<index number>.xlsx

	A	B	C	D	E	F	G	H	I	J	K
1	**Car Loan Payment Records**										
2	**Customer Ref. No.**	**Purchase Date**	**Year of Purchase**	**Loan Amount ($)**	**No. of Years of Loan**	**Month of Repayment**	**Annual Interest Rate (%)**	**Monthly Principal Payment ($)**	**Monthly Interest Payment ($)**	**Total Monthly Payment ($)**	**Credit Status**
3	129128	4/12/2017		78000	10	2					
4	329233	16/5/2018		25000	3	15					
5	554532	23/4/2012		120000	10	17					
6	135818	18/5/2010		115000	10	9					
7	928281	1/12/2014		35000	5	22					
8	284949	19/5/2012		96000	10	36					
9	282919	21/12/2013		50000	5	30					
10	948181	24/4/2011		135000	10	8					
11	847710	19/3/2018		80000	3	10					
12	719292	15/2/2012		93000	10	7					
13											
14	**No. of Years of Loan**	**Annual Interest Rate (%)**									
15	3	2.2									
16	5	2.5									
17	10	2.9									
18											

1 In cell **C3** enter an appropriate formula to extract the year of purchase of the car from the purchase date given, and use it to complete the **Year of Purchase** column. [1]

2 Use an appropriate function to search for the **Annual Interest Rate (%)** and use it to complete the **Annual Interest Rate (%)** column. [2]

3 In cell **H3** enter an appropriate formula to calculate the monthly principal payment of the car, and use it to complete the **Monthly Principal Payment ($)** column. [2]

4 In cell **I3** enter an appropriate formula to calculate the monthly interest payment of the car, and use it to complete the **Monthly Interest Payment ($)** column. [2]

5 In cell **J3** enter an appropriate formula to calculate the total monthly payment of the car, and use it to complete the **Total Monthly Payment ($)** column. [1]

6 The company uses a credit status system to classify its customer.

There are 3 groups of customers:

A: total monthly payment is less than $1000

B: total monthly payment is less than $2000

C: total monthly payment is $2000 or more

Use a conditional statement in cell **K3** to indicate if the customer belongs to group **A**, **B** or **C**, and use it to complete the **Credit Status** column. [2]

Save and close your file.

Task 2

The following program accepts 20 words to be input, and prints out the shortest word.

```
word = input ("Enter a word: ")
shortest_word = word
num_words = 19

for count in range (num_words):
    word = input ("Enter a word: ")

    if len(shortest_word) > len(word):
        shortest_word = word

print ("Shortest word is " + shortest_word)
```

Open the file **WORDS.py**

Save the file as **MYWORDS**_<your name>_<centre number>_<index number>

7 Edit the program so that it:

(a) Accepts only 10 words. [1]

(b) Prints out the longest word. [3]

(c) Tests if a word is a palindrome. A palindrome is a word or sequence that reads the same backwards as forwards, e.g. madam, level, rotor.

Program should count the number of palindromic words and print out the number at the end of the program. [4]

8 Save your program as **VARWORDS**_<your name>_<centre number>_<index number>

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

Save your program.

Task 3

A company recently purchased a drink dispenser for its staff. The dispenser is programmed such that it is able to dispense multiple drinks at any one time using the following rules:

- User is able to select the type of drink: 1: Coffee, 2: Coke, 3: Tea, 4: Milo.
- User then chooses the quantity of the selected drink to dispense.
- Selecting the same type of drink again will allow user to re-enter its quantity to be dispensed.
- Quantity selected for a drink cannot exceed its stock. Output a statement if this happens.

The program finishes when user inputs 0 to exit. The program then computes the quantity for each drink and displays that information to the user.

There are several syntax and logic errors in the program.

```
vending_list = ["Coffee", 23, "Coke", 12, "Tea", 19, "Milo", 18]

total_order = [0,0,0,0]
message = "Please enter your choice (1: Coffee, 2: Coke, " \
"3: Tea, 4: Milo, 0: Confirm Selection): "

order = int(input(message))
if order != 0:
    qty = input("Enter the quantity to dispense: ")

while order != 0:
    if order == 0:
        if vending_list[1] - qty >= 0:
            total_order [0] = qty
        else:
            print ("Quantity unavailable, " \
                   + str(vending_list[1]) + " remaining")

    elif order == 2:
        if vending_list[3] - qty >= 0:
            total_order [1] == qty
        else:
            print ("Quantity unavailable, " \
                   + str(vending_list[3]) + " remaining")

    elif order == 3:
        if vending_list[5] - qty >= 0
            total_order [2] = qty
        else:
            print ("Quantity unavailable, " \
                   + str(vending_list[5] + " remaining")
```

```
    if order == 4:
        if vending_list[7] - qty >= 0:
            total_order [3] = qty
        else:
            print ("Quantity unavailable, " \
                   + str(vending_list[7]) + " remaining")

    else:
        print ("Input error.")


    order = int(input (message))
    if order == 0:
        qty = int(input ("Enter the quantity to dispense: "))

vending_list[1] = vending_list[1] - total_sold[0]
vending_list[3] = vending_list[3] - total_sold[1]
vending_list[5] = vending_list[5] - total_sold[2]


print ("You have bought the following items: ")

i = 1
while i in range (len(total_order)):
    if total_order[i] > 0:
        print (vending_list[i+2] + ": " + str(total_order[i]))
    i += 1
```

Open the file **VENDING.py**

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

9 Identify and correct the errors in the program so that it works according to the rules given. [10]

Save your program.

Task 4

A theme park determines the price of admission based on the age of the guest.

Category of Guests	Admission Fee
Infants (less than 3 years old)	Free
Children (3 to 12 years old)	$25.00
Adults	$37.00
Seniors (at least 60 years old)	$17.00

You have been asked to write a program that will help the theme park to calculate its daily revenue.

Assuming that there are only five groups, each comprising four guests who purchased tickets today, the program should allow you to:

- Enter data in the format a, b, c, d where a, b, c, d are the ages (in years) of the respective guests in the group.
- An example is:
  `12, 16, 49, 50`
- Only allow data entry of 0 to 120 as the age (in years) of any guest.
- Calculate the breakdown of the number of guests who entered the theme park from each of the categories: infants, children, adults and seniors.
- Calculate the average age (rounded down to the nearest year) of guests for the day.
- Calculate the daily revenue collected from each of the categories: children, adults and seniors.
- Calculate the total daily revenue collected.

- Display this on the screen. Your output **must** look like this:

```
Guest statistics
Number of Infants: 2
Number of Children: 6
Number of Adults: 9
Number of Seniors: 3

Average age of guests is 28 years old

Revenue by category
Children: $336.00
Adults: $684.00
Seniors: $114.00

Collected total revenue of $1134.00
```

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

Save your program as **PARK**_<your name>_<centre number>_<index number>

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

```
12, 16, 49, 50
10, 13, 38, 1
66, 65, 59, 3
32, 60, 11, 4
30, 32, 10, 5
```

Take a screen shot of your results and save it as **PARKRESULTS**_<your name>_<centre number>_<index number> [5]

Save your files in either **.png** or **.jpg** format.

12 Save your program as **PARK2**_<your name>_<centre number>_<index number>
Extend your program to allow any number of guests (minimum 1) to be entered per group. [3]

Use the following test data to show your test results:

```
12, 16, 49, 50
10, 13, 38
66, 65, 59, 3, 1
30, 32, 60, 11, 10, 4, 5
32
```

Your output **must** look like this:

```
Guest statistics
Number of Infants: 2
Number of Children: 6
Number of Adults: 9
Number of Seniors: 3

Average age of guests is 28 years old

Revenue by category
Children: $336.00
Adults: $684.00
Seniors: $114.00

Collected total revenue of $1134.00
```

Save your program.

13 Save your program as **PARK3**_<Class>_<Index_Number>_<Your_Name>
Extend your program to work for any number of groups. [2]
Save your program.

-- End of Paper –

BLANK PAGE