ST. PATRICK'S SCHOOL
PRELIMINARY EXAMINATIONS 2020
SECONDARY 4 EXPRESS

NAME

CLASS | INDEX NUMBER

COMPUTING **7155/02**
Paper 2 (Lab-based) **24 August 2020**
2 h 30 min

Additional Materials:
- Electronic version of `THEMEPARK.xlsx` data file
- Electronic version of `GARGLE.py` file
- Electronic version of `INTEGERS.py` file
- Quick Reference Glossary

READ THESE INSTRUCTIONS FIRST

Write your name, class, index number in the spaces 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.

Parent's Signature : _____

Date: _____

Remarks (if any) :

For Examiner's Use	
Marks	/50
Total	**%**

ANNEX A

Quick Reference for Python

This quick reference shows some examples of the Python language constructs. The complete Python language is not limited to these examples.

1. Identifiers

When naming functions, variables and modules, the following rules must be observed:

- Names should begin with character 'a' - 'z' or 'A' - 'Z' or '_'
  and followed by alphanumeric characters or '_'
  .
- Reserved words should not be used.
- User-defined identifiers are case sensitive.

2. Comments and Documentation Strings

```
# This is a comment

"""
   This is a documentation string
   over multiple lines
"""
```

3. Input/Output

```
print ("This is a string")

s = input ("Instructions to prompt for data entry.")
```

4. Import

import <module>

e.g. **import** math

5. Data Type

Data Type	Notes
int	integer
float	real number
bool	boolean
str	string (immutable)
list	series of values

6. Assignment

Assignment Statement	Notes
a = 1	integer
b = c	variable
d = "This is a string"	string
mylist = [1, 2, 3, 4, 5]	list or array

7. Arithmetic Operators

Operator	Notes
+ -	plus, subtract
* /	multiply, divide
%	remainder or modulus
**	exponential or power
//	quotient of the floor division

8. Relational Operators

Operator	Notes
==	equality
!=	not equal to
> >=	greater than, greater than or equal to
< <=	less than, less than or equal to

9. Boolean Expression

Boolean Expression	Notes
a and b	logical and
a or b	logical or
not a	logical not

10. Iteration

while loop	for loop
while condition(s): <statement(s)>	**for** i **in** range(n): <statement(s)>
	for record **in** records: <statement(s)>

ANNEX A

11. Selection

Type 1	Type 2	Type 3
if condition(s): <statement(s)>	**if** condition(s): <statement(s)> **else:** <statement(s)>	**if** condition(s): <statement(s)> **elif** condition(s): <statement(s)> **else:** <statement(s)>

12. Built-in Functions

(a) Basic functions

abs( )	chr( )	float()	input( )	int( )
ord( )	print( )	range( )	round( )	str( )
format( )				

(b) Mathematical functions

ceil( )	exp( )	fabs( )	floor( )	log( )
max()	min()	pow( )	sqrt( )	trunc( )

(c) String functions

endswith()	find()	isalnum( )	isalpha( )	isdigit( )
islower( )	isspace( )	isupper( )	len()	lower( )
startswith()	upper( )			

13. Reserved Words

Reserved words cannot be used as identifiers. They are part of the syntax of the language.

False	None	True	and	as
assert	break	class	continue	def
del	elif	else	except	finally
for	from	global	if	import
in	is	lambda	nonlocal	not
or	pass	raise	return	try
while	with	yield		

Task 1

A holiday theme park, Theme Park Pte Ltd uses a spreadsheet software to calculate their ticket sales. You are required to finish setting up the spreadsheet to calculate the revenue gained for each transaction number.

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

Save the file as **MYTHEMEPARK**_<Class>_<Class_Index_Number>_<Your_Name>

Theme Park Pte Ltd

Ticket Sales Records

Today's Date: 17/7/2020

No	Transaction Number	Package Name	Number of Tickets Sold	Full Cost	Transaction Type	Discount	Revenue
1	2020A0997		1451		NORMAL		
2	2020C9990		662		CORPORATE		
3	2020C3289		1391		STUDENT		
4	2020A2888		670		CORPORATE		
5	2020B7597		1356		STUDENT		
6	2020A7117		1155		CORPORATE		
7	2020A7023		723		STUDENT		
8	2020C0090		867		CORPORATE		
9	2020B4393		884		STUDENT		
10	2020C4922		1125		STUDENT		

Package

Code	Name	Price
A	PASS Play Max	$ 81.90
B	2 DAY FUN PASS	$ 141.90
C	DAY FUN PASS Play 5	$ 59.90

Discount

Type	STUDENT	CORPORATE	NORMAL
Discount	10%	5%	0
Number			

1 The fifth character of each **Transaction Number** is the **Code** of the **Package Name**. In cells **C7** to **C16** enter a formula that uses an appropriate function to search for the **Package Name** in the **Package** table. Use it to display the **Package Name**. [2]

2 In cells **E7** to **E16** enter a formula that uses an appropriate function to search for the **Price** in the **Package** table. Use it to calculate the **Full Cost** in currency format. [2]

3 In cells **G7** to **G16** enter a formula that uses an appropriate function to search for the **Discount** in the **Discount** table. Use it to calculate the **Discount** given in currency format. [2]

4 In cells **H7** to **H16** enter a conditional statement to calculate the **Revenue**. If the **Transaction Type** is **CORPORATE**, apply an additional 2% of the **Revenue**. [2]

5 In cells **F36** to **H36** enter a conditional statement to calculate the number of each Transaction Type. [2]

Task 2

The following program creates a Gargle Suite account for a user. It creates the username by taking the first letter of the user's name and combining it with the user's ID. It will also allow the user to enter a password.

```
name = input("Please enter your full name: ")
id = input("Please enter your ID: ")
username = name[0] + id
print("Your username is " + username)
password = input("Please enter a password: ")
```

Open the file **GARGLE.py**.

Save the file as **MYGARGLE**_<Class>_<Class_Index_Number>_<Your_Name>**.py**

6 Edit the program so that the username is created using the first 5 characters of the user's name and combining with the last 5 characters of the user's ID. The program should also ignore any spaces in the user's name when creating the username. [3]

7 The program needs to validate both the password and whether the user has correctly re-entered their password.

(a) Edit the program to:

- test whether the user has entered a password of eight characters or more and that it consists of at least 1 capital letter, 1 small letter and 1 numeral.
- output a suitable error message that asks the user to enter a password again if the password does not meet the above criteria, and repeat this until the user enters a valid password. [3]

(b) Edit the program to:

- ask the user to re-enter their password (i.e. second entry of password)
- output a suitable error message that asks the user to enter a new password if the second entry of the password does not match the first entry, and repeat this until the user enters a valid password and the second entry matches the first entry of the password.
- output the user's username and email address in this format:

```
Your account has been set up successfully.
Your username is ElgooG2020
Your email address is elgoog2020@gargle.com
```

[4]

Save your program.

Task 3

The following program asks user to input a series of integers and checks whether the integer is a positive, negative or zero. The program also outputs the list of integers in ascending order. The program quits when the user enters "q".

There are several syntax and logic errors in the program.

```
data
negatives = []
positives = []
zeroes = []
num_zeroes = 0

num = int(input("Enter an integer (q to quit): "))
while num != q:
    num = num
    data.extend(num)
    if num < 0:
        negatives.append(num)
    elif num > 0:
        positives.append(num)
    else
        zeroes.append(num)
num = input("Enter an integer (q to quit): ")

data = sort()
print("\nThe integers, sorted in ascending order, are: {}\n".format(data))

print("\nThe negative values are: ")
for num in negatives:
    print(num)

print("\nThe positive values are: ")
for num in positives:
    print(num)

num_zeroes = count(zeroes)
  print("\nThere are {} zeroes.", num_zeroes)
```

Open the file **INTEGERS.py**.

Save the file as **MYINTEGERS**_<Class>_<Class_Index_Number>_<Your_Name>**.py**

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

Save your program.

Task 4

You have been asked to create an unscrambling word game program. This program simulates a two-player game. Player 1 enters a word. The program then scrambles it for Player 2 to figure out Player 1's word.

The program should:

- allow Player 1 to input a word of length between 3 and 45, for Player 2 to unscramble. Instruct Player 1 to only enter lowercase letters from the English alphabet. There must be a validation present to check that the word entered is within the above characters limit and that it is made up of only lowercase letters from the English alphabet.
- allow Player 2 to have three attempts to correctly unscramble the word input by Player 1. You do not need to validate the input for Player 2.
- output an appropriate message when Player 2 manages to unscramble correctly and inputs the same word entered by Player 1. The game ends when Player 2 unscrambles correctly.
- output an appropriate message when Player 2 does not input the same word entered by Player 1.
- output an appropriate message when Player 2 has 3 incorrect attempts. The game must also end here.

9 Write your program and test that it works.
Save your program as **MYWORD1**_<Class>_<Class_Index_Number>_<Your_Name>**.py** [10]

10 When your program is complete, test it for the following:

- Test 1: Player 1 inputs the word "ISOSCELES"
- Test 2: Player 1 inputs the word "an"
- Test 3: Player 1 inputs the word "triangle" and player 2 enters "traingle" and "triangle".
- Test 4: Player 1 inputs the word "hen" and player 2 enters "neh", "enh", "nhe".

Take a screenshot of:

- Test 1, 2, 3 and 4. Save this **single** screenshot as:
  TEST1234_<Class>_<Class_Index_Number>_<Your_Name>

Save your file in either **.png** or **.jpg** format. [4]

11 Save your program as **MYWORD2**_<Class>_<Class_Index_Number>_<Your_Name>**.py**

Extend your program to keep track of the number of attempts left for Player 2. Outputs a suitable message on the number of attempts left before asking Player 2 to enter his word.

Save your program. [2]

12 Save your program as **MYWORD3**_<Class>_<Class_Index_Number>_<Your_Name>**.py**

Extend your program to allow Player 2 to choose an easy, medium or hard game. An easy game allows five attempts, a medium game allows three attempts and a hard game allows two attempts.

Save your program. [4]

End of Paper