Roll N0. | | | | |

Internal Assessment Test II – August 2023

Sub:	Introduction to Python Programming-Scheme and Solution					Sub Code:	BPLCK105B	Branch:	Chemistry Cycle
Date:	10-08-2023	Duration:	90 min's	Max Marks:	50	Sem / Sec:	I A,B,C,E,F		OBE

	Answer any FIVE FULL QUESTIONS	MARKS	CO	RBT
1 (a)	Compare List with Tuple. Also Explain the following operations in Lists i) List Concatenation ii) List Replication **Any 2 points with example [2 marks]** **Each of List concatenation and List replication contains 1 mark.**	2+2	CO2	L2

List	Tuple
List is mutable. List values modified, appended, or removed.	Tuple is immutable. Tuples cannot have their values modified, appended, or removed.
Lists are typed with square brackets, []. spam = ['cat', 'dog', 'bat']	Tuples are typed with parentheses (). eggs = ('hello', 42, 0.5)

List Concatenation : The + operator combines two lists to create a new list value.

```
>>> [1, 2, 3] + ['A', 'B', 'C']
[1, 2, 3, 'A', 'B', 'C']
```

List Replication: The * operator can be used with a list and an integer value to replicate the list.

```
>>> ['X', 'Y', 'Z'] * 3
['X', 'Y', 'Z', 'X', 'Y', 'Z', 'X', 'Y', 'Z']
```

(b)	Explain about following string methods with example code snippets i) join() and endswith() ii) strip methods **(1+1) =2M x 3=6 Marks** **i) join()** is an inbuilt string function in Python used to join elements of the sequence separated by a string separator. > ', '.join(['cats', 'rats', 'bats']) 'cats, rats, bats' >>> ' '.join(['My', 'name', 'is', 'Simon']) 'My name is Simon' ii) The endswith() method return True if the string value they are called on ends with the string passed to the method; otherwise, it return False. Example: 'Hello, world!'.endswith('world!')	6	CO2	L2

Course Instructor Chief Course Instructor

	True
	'abc123'.endswith('12')
	False
	iii) The string strip() method helps to remove the whitespaces or specific characters from the string at the beginning and end of the string.
	`>>> spam = '  Hello World  '`
	`>>> spam.strip()`
	`'Hello World'`
	`>>> spam.lstrip()`
	`'Hello World  '`
	`>>> spam.rstrip()`
	`'  Hello World'`			

2 (a)	ABC Electronics has 5 outlets in Cities Bangalore, Mumbai, Delphi, Chennai, and Kolkotta. They sell different combinations of electronics items such as Labtops, Desktops, Mobile, Televisions, Tablets and Monitors. Write a python program to store these items list in Nested Dictionary. Have city's names as the keys and dictionary (inner) which contains item lists as values. Write a function totalitems() that would take item name either 'Laptops', or 'Desktops' or 'Tablets' as argument and display the total count of the item passed in all 5 cities..
	• **Correct logic [3 marks]**
	• **Correct syntax[2 marks]**
	
	outlets={"Bangalore":{"Laptops":45,"Desktops":80,"Mobiles":55,"Televisions":80, "Tablets":55},
	"Mumbai":{"Laptops":67,"Desktops":85,"Televisions":80,"Tablets":55},
	"Delhi":{"Laptops":34,"Desktops":80,"Mobiles":55,"Televisions":80,"Tablets":25},
	"Chennai":{"Laptops":25,"Desktops":80,"Mobiles":155,"Tablets":55},
	"Kolkata":{"Desktops":80,"Mobiles":58,"Televisions":80,"Tablets":90},
	}
	
	def totalitems(out,item):
	total=0
	for k,v in out.items():
	total=total+v.get(item,0)
	return total
	
	print(totalitems(outlets,"Televisions"))
	print(totalitems(outlets,"Mobiles"))
	print(totalitems(outlets,"Laptops"))
		5	CO2	L3

(b)	Explain the use of get() and setdefault() methods related to dictionary with suitable code snippet. • **Correct definition and syntax[3 marks]** • **Correct example[2 marks]** get() Method return the value for the given key if present in the dictionary. If not, then it will return None (if get() is used with only one argument). Syntax : Dict.get(key, default=None)	5	CO2	L2

Course Instructor Chief Course Instructor

	Example:
	picnicItems = {'apples': 5, 'cups': 2}
	>>> 'I am bringing ' + str(picnicItems.get('cups', 0)) + ' cups.'
	'I am bringing 2 cups.'
	
	setdefault() returns the value of a key (if the key is in dictionary). Else, it inserts a key with the default value to the dictionary.
	Syntax: dict.setdefault(key, default_value)
	Example:
	>>> spam = {'name': 'Pooka', 'age': 5}
	>>> spam.setdefault('color', 'black')
	'black'
	spam {'color': 'black', 'age': 5, 'name': 'Pooka'}
	>>> spam.setdefault('color', 'white')
	'black'			

3 (a)	Illustrate with example how the copy.copy() is different from copy.deepcopy() which is relevant to
	lists or dictionaries in Python.
	• **Correct definition and syntax[3 marks]**
	• **Correct example[2 marks]**
	Python provides a module named copy that provides both the copy() and deepcopy() functions. The first of these, copy.copy(), can be used to make a duplicate copy of a mutable value like a list or dictionary, not just a copy of a reference.
	Enter the following into the interactive shell:
	>>> import copy
	>>> spam = ['A', 'B', 'C', 'D']
	>>> cheese = copy.copy(spam)
	>>> cheese[1] = 42
	>>> spam ['A', 'B', 'C', 'D']
	>>> cheese ['A', 42, 'C', 'D']
	Syntax of Shallow copy: copy.copy(x)
	Syntax of Deep copy: copy.deepcopy(x)
	Example: import copy
	>>> import copy
	>>> spam = ['A', 'B', 'C', 'D']
	>>> cheese = copy.copy(spam)
	>>> cheese[1] = 42
	>>> spam
	['A', 'B', 'C', 'D']
	>>> cheese
	'A', 42, 'C', 'D']
	If the list you need to copy contains lists, then use the copy.deepcopy() function instead of copy.copy(). The deepcopy() function will copy these inner lists as well.
	import copy
	l1=[1,2,[3,4],5,6]
	l2=copy.deepcopy(l1)
	l1[2][1]=50
		5	CO2	L2

Course Instructor Chief Course Instructor

	print(l1)
	print(l2)
	Output:
	`[1, 2, [3, 50], 5, 6]`
	`[1, 2, [3, 4], 5, 6]`			

(b)	Write a Python program to input 'n' strings as input and count the number of strings which contain all the vowels
	• Correct **definition and syntax[3 marks]**
	• Correct **logic [2 marks]**
	l=[]
	n=int(input())
	wc=0
	for i in range(n):
	s=input()
	l=l+[s]
	for st in l:
	st=st.lower()
	flag={'a':0,'e':0,'i':0,'o':0,'u':0}
	
	for c in st:
	if c=='a':
	flag['a']+=1
	
	elif c=='e':
	flag['e']+=1
	
	elif c=='i':
	flag['i']+=1
	
	elif c=='o':
	flag['o']+=1
	
	elif c=='u':
	flag['u']+=1
	
	if flag['a'] >0 and flag['e']>0 and flag['i'] >0 and flag['o'] >0 and flag['u'] >0 :
	wc=wc+1
	print(wc)
		5	CO2	L3

Course Instructor Chief Course Instructor

4(a)	**Explain the use of 'in' and 'not in' operators with example code snippets**
	Definition/Description [2 Marks]
	Example code snippets [3 Marks]
	
	To check whether a value is or isn't in a list, the in and not in operators are used. Both in and not in are used in expressions and connect two values: a value to look for in a list and the list where it may be found. These expressions will evaluate to a Boolean value.
	`>>> 'howdy' in ['hello', 'hi', 'howdy', 'heyas']`
	`True`
	`>>> spam = ['hello', 'hi', 'howdy', 'heyas']`
	`>>> 'cat' in spam`
	`False`
	`>>> 'howdy' not in spam`
	`False`
	`>>> 'cat' not in spam`
	`True`
		[5]	CO2	L2

(b)	**Write a Python program to remove all duplicate words in a given inputted sentence.**
	Correct logic [3 marks]
	Correct syntax[2 marks]
	
	`s1=input("enter a string")`
	`s1=s1.split(' ')`
	`l1=[]`
	`for words in s1:`
	`  if words not in l1:`
	`  l1.append(words)`
	`s2=" ".join(l1)`
	`print(s2)`
		[5]	CO2	L3

5(a)	**Consider below expressions and identify type of errors occurs. Justify Your Answer.** **i) spam = ['hello', 'hi', 'howdy', 'heyas']** **spam.index('howdy howdy howdy')** **ii) x=[1,2,3,4, 'Alice', 'Bob']** **x.sort()** **iii) y={'man': 'abc', 'woman': 'xyz'}** **y['person']** **iv) spam= "hello world!"** **spam.islower()** **Correct Description [4X1 Marks]** **Explanation [1 Marks]** i) ValueError: 'howdy howdy howdy' is not in list. In index() method a single value can be passed, and if that value exists in the list, the index of the value is returned. ii) TypeError: '<' not supported between instances of 'str' and 'int'. Lists that have both number values and string values cannot be sorted iii) KeyError: 'person'. The key must be present in the dictionary. iv) It is correct. No error	[5]	CO3	L2

Course Instructor

Chief Course Instructor

(b)	**What is Shelve Module? How to open/create a file using this module and write content into and read the content from file using this module.**
	Definition/Description [2 Marks]
	Example code snippets [3 Marks]
	
	Variables in Python programs can be saved to binary shelf files using the shelve module. This way, the program can restore data to variables from the hard drive. The shelve module aloows to add Save and Open features to program.
	To read and write data using the shelve module, you first import shelve. Call shelve.open() and pass it a filename, and then store the returned shelf value in a variable. You can make changes to the shelf value as if it were a dictionary. When you're done, call close() on the shelf value. Here, our shelf value is stored in shelfFile.
	
	`>>> import shelve`
	`>>> shelfFile = shelve.open('mydata')`
	`>>> cats = ['Zophie', 'Pooka', 'Simon']`
	`>>> shelfFile['cats'] = cats`
	`>>> shelfFile.close()`
	
	`>>> shelfFile = shelve.open('mydata')`
	`>>> type(shelfFile)`
	`<class 'shelve.DbfilenameShelf'>`
	`>>> shelfFile['cats']`
	`['Zophie', 'Pooka', 'Simon']`
	`>>> shelfFile.close()`
		[5]	CO2	L3

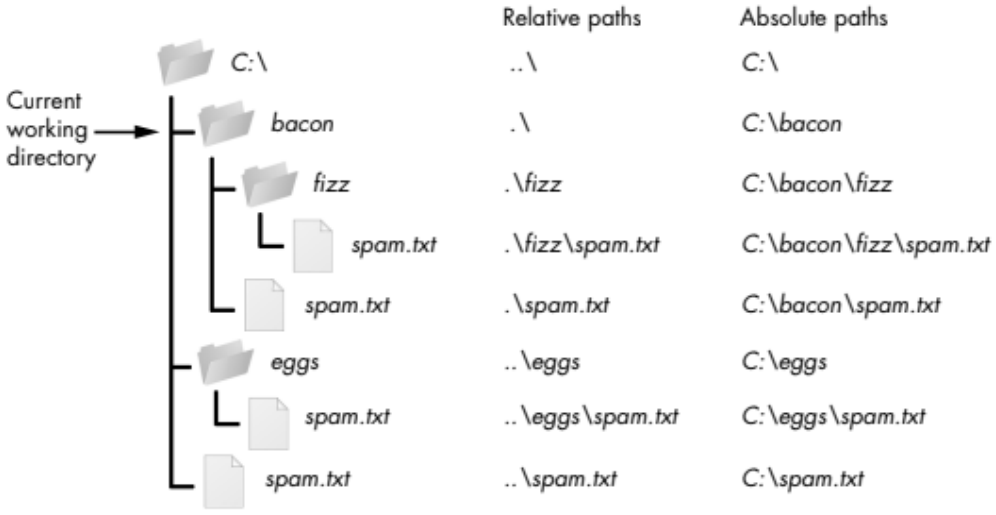
6 (a)	**Explain about the use(at least 2 methods)of following modules in Python with code snippets**
	1) os 2) pathlib
	Definition/Description [2 Marks]
	Example code snippets [3 Marks]
	
	1)**os module** 4 functions given. **Any two they can explain**
	os.chdir()
	to change to a directory that is existing directory: 'C:\\Windows\\System32':
	>>> os.chdir('C:\\Windows\\System32')
	Python will display an error if you try to change to a directory that does not exist.
	>>> os.chdir('C:/ThisFolderDoesNotExist')
	
	2) os.mkdir()
	
	To make a directory from a Path object, call the mkdir() method. For example, this code will create a spam folder under the home folder on my computer:
	
	>>> from pathlib import Path
	>>> Path(r'C:\Users\Al\spam').mkdir()
	
	Note that mkdir() can only make one directory at a time; it won't make several subdirectories at once like os.makedirs().
		[5]	CO3	L2

Course Instructor Chief Course Instructor

	3) os.path.abspath(path) will return a string of the absolute path of the argument. This is an easy way to convert a relative path into an absolute one.
	`>>> os.path.abspath('.')`
	`'C:\\Users\\Al\\AppData\\Local\\Programs\\Python\\Python37'`
	`>>> os.path.abspath('.\\Scripts')`
	`'C:\\Users\\Al\\AppData\\Local\\Programs\\Python\\Python37\\Scripts'`
	4) Calling os.path.isabs(path) will return True if the argument is an absolute path and False if it is a relative path.
	`>>> os.path.isabs('.')`
	`False`
	`>>> os.path.isabs(os.path.abspath('.'))`
	`True`
	
	2) Pathlib Module **Any two they can explain**
	
	Path() function in the pathlib module will take the string values of individual file and folder names in a path as parameters, and will return a string with a file path.
	`>>> from pathlib import Path`
	`>>> Path('spam', 'bacon', 'eggs')`
	`WindowsPath('spam/bacon/eggs')`
	
	You can get the current working directory as a string value with the Path.cwd() function
	`>>> from pathlib import Path`
	`>>> import os`
	`>>> Path.cwd()`
	`WindowsPath('C:/Users/Al/AppData/Local/Programs/Python/Python37')'`
	3. The Home Directory
	
	All users have a folder for their own files on the computer called the home directory or home folder. You can get a Path object of the home folder by calling Path.home():
	
	>>> Path.home()
	WindowsPath('C:/Users/Al')			

(b)	**Write a Python program to find the total size of the MSWORD files only in the folder 'C:\\Windows\\System32'. Assume that Folder contains only files of different types.** Correct logic **[3 marks]** Correct syntax**[2 marks]** p = Path(**'C:\\Windows\\System32'**) s=0 for textFile in p.glob('*.docx'):	[5]	CO3	L2

Course Instructor Chief Course Instructor

	print(textFile) s=s+os.path.getsize(textFile) print(s)			
7 (a)	Explain the difference between Absolute path and relative path with example codes snippets **Definition/Description [2 Marks]** **Example code snippets [3 Marks]** • An absolute path, which always begins with the root folder • A relative path, which is relative to the program's current working directory There are also the *dot* (.) and *dot-dot* (..) folders. These are not real folders but special names that can be used in a path. A single period ("dot") for a folder name is shorthand for "this directory." Two periods ("dot-dot") means "the parent folder." 2 Example for Absolute path and Relative path form the given diagram is sufficient:  	[5]	CO3	L3

(b)	Explain with program how a text file can be opened, written with content, read the content from the file and append the content into the file.
	Definition/Description [2 Marks]
	Example code snippets [3 Marks]
	To Open a File :
	>>> helloFile = open('/Users/your_home_folder/hello.txt')
	
	The read() method returns the string that is stored in the File.
	>>> helloContent = helloFile.read()
	>>> helloContent
	'Hello, world
	
	To Write into File :
	>>> baconFile = open('bacon.txt', 'w')
	>>> baconFile.write('Hello, world!\n')
	Example :
	>>> baconFile = open('bacon.txt', 'w')
		[5]	CO3	L2

Course Instructor Chief Course Instructor

	>>> baconFile.write('Hello, world!\n')
	13
	>>> baconFile.close()
	>>> baconFile = open('bacon.txt', 'a+')
	>>> baconFile.write('Bacon is not a vegetable.')
	25
	>>> baconFile.close()
	>>> baconFile = open('bacon.txt')
	>>> content = baconFile.read()
	>>> baconFile.close()
	>>> print(content)
	
	Output:
	Hello, world!
	Bacon is not a vegetable			

Course Instructor Chief Course Instructor