USN | | | | | | | | | | |

Internal Assessment Test 4 – Feb 2022

Sub:	Application Development Using Python					Sub Code:	18CS55	Branch:	ISE	
Date:	03/02/2022	Duration:	90 min's	Max Marks:	50	Sem/Sec:	V A,B&C		OBE	

Answer any FIVE FULL Questions

		MARKS	CO	RBT
1a)	What is Dictionary in Python? How is it different from List data type? Explain how a for loop can be used to traverse the keys of the Dictionary with an example. **[1+2+2 marks]**	[5]	CO2	L2

Dictionary: A dictionary is a collection of many values. Indexes for dictionaries can use many different data types, not just integers. Indexes for dictionaries are called keys, and a key with its associated value is called a key-value pair. A dictionary is typed with braces, {}.

The dictionary is different from list data type:

- ✓ In lists the items are ordered and in dictionaries the items are unordered.
- ✓ The first item in a list exists. But there is no "first" item in a dictionary.
- ✓ The order of items matters for determining whether two lists are the same, it does not matter in what order the key-value pairs are typed in a dictionary.
- ✓ Trying to access a key that does not exist in a dictionary will result in a KeyError error message, in list "out-of-range" IndexError error message.

Traversing using for loop:

```
>>> spam = {'color': 'red', 'age': 42}
>>> for k, v in spam.items():
        print('Key: ' + k + ' Value: ' + str(v))

Key: age Value: 42
Key: color Value: red
```

1b)	What is the difference between **copy.copy( )** and **copy.deepcopy( )** functions applicable to a List or Dictionary in Python? Give suitable examples for each.	[5]	CO2	L2

[Explanation 2 Marks+ Program 3 marks]

Copy module must be imported and can be used to make a duplicate copy of a mutable value like a list or dictionary, not just a copy of a reference.

copy.copy( ): A shallow copy creates a new object which stores the reference of the original elements.So, a shallow copy doesn't create a copy of nested objects, instead it just copies the reference of nested objects. This means, a copy process does not create copies of nested objects itself.

Example:

```
import copy
old_list = [[1, 1, 1], [2, 2, 2], [3, 3, 3]]
```

	new_list = copy.copy(old_list old_list[1][1] = 'AA' print("Old list:", old_list) print("New list:", new_list) **Output:** Old list: [[1, 1, 1], [2, 'AA', 2], [3, 3, 3]] New list: [[1, 1, 1], [2, 'AA', 2], [3, 3, 3]] **copy.deepcopy( ):** A deep copy creates a new object and recursively adds the copies of nested objects present in the original elements. **Example:** import copy old_list = [[1, 1, 1], [2, 2, 2], [3, 3, 3]] new_list = copy.deepcopy(old_list) old_list[1][0] = 'BB' print("Old list:", old_list) print("New list:", new_list) **Output:** Old list: [[1, 1, 1], ['BB', 2, 2], [3, 3, 3]] New list: [[1, 1, 1], [2, 2, 2], [3, 3, 3]] **copy.deepcopy( ):** A deep copy creates a new object and recursively adds the copies of nested objects present in the original elements. **Example:** import copy old_list = [[1, 1, 1], [2, 2, 2], [3, 3, 3]] new_list = copy.deepcopy(old_list) old_list[1][0] = 'BB' print("Old list:", old_list) print("New list:", new_list) **Output:** Old list: [[1, 1, 1], ['BB', 2, 2], [3, 3, 3]] New list: [[1, 1, 1], [2, 2, 2], [3, 3, 3]]			
2a)	Differentiate mutable and immutable data types with example Python Mutable data types are those whose values can be changed in place whereas Immutable data types are those that can never change their value in place. Here are Mutable data types :[2 Marks] Lists Dictionaries Sets And Immutable data types in Python: Integers Floating-Point numbers Booleans Strings Tuples Examples : with code: **[3 marks]**	[5]	CO2	L2
2b)	Write short Note on 1. pprint Module with pprint() and pformat() functions **[Code-1.5 Marks Explanation - 1 Mark]**	[5]	CO2	L2

```
>>> import pprint
>>> cats = [{'name': 'Zophie', 'desc': 'chubby'}, {'name': 'Pooka',
'desc': 'fluffy'}]
>>> pprint.pformat(cats)
"[{'desc': 'chubby', 'name': 'Zophie'}, {'desc': 'fluffy', 'name': 'Pooka'}]"
>>> fileObj = open('myCats.py', 'w')
>>> fileObj.write('cats = ' + pprint.pformat(cats) + '\n')
83
>>> fileObj.close()
```

Here, we import pprint to let us use pprint.pformat(). We have a list of dictionaries, stored in a variable cats. To keep the list in cats available even after we close the shell, we use pprint.pformat() to return it as a string. Once we have the data in cats as a string, it's easy to write the string to a file, which we'll call *myCats.py*.

The modules that an import statement imports are themselves just Python scripts. When the string from pprint.pformat() is saved to a *.py* file, the file is a module that can be imported just like any other.

2. join() and split() methods with example

[Code-1.5 Marks Explanation - 1 Mark]

Join()

- The join() method is useful when we have a list of strings that need to be joined together into a single string value.
- The join() method is called on a string, gets passed a list of strings, and returns a string. The returned string is the concatenation

```
>>> ', '.join(['cats', 'rats', 'bats'])
'cats, rats, bats'
>>> ' '.join(['My', 'name', 'is', 'Simon'])
'My name is Simon'
>>> 'ABC'.join(['My', 'name', 'is', 'Simon'])
'MyABCnameABCisABCSimon'
```

of each string in the passed-in list.

Split()

- The split() method is called on a string value and returns a list of strings.

We can pass a delimiter string to the split() method to specify a different string to

```
>>> 'My name is Simon'.split()
['My', 'name', 'is', 'Simon']
```

split upon

3(a)	Discuss the use of below Regular expression symbols with example 1. ? , *, . + 2. {n},{,n} and{n,} 3. ^(caret character) & $(dollar symbol) **Explanation: 3 marks** • The ? matches zero or one of the preceding group. • The * matches zero or more of the preceding group. • The + matches one or more of the preceding group. • The {n} matches exactly *n* of the preceding group. • The {n,} matches *n* or more of the preceding group. • The {,m} matches 0 to *m* of the preceding group.	[6]	CO3	L3

	• spam$ means the string must end with *spam*. • ^spam means the string must begin with *spam*. • The . matches any character, except newline characters. • [^abc] matches any character that isn't between the brackets. **Example code for Each: group: 3 marks**			
3(b)	Discuss how will you read content of files using open(),read() , readlines() function with example program. **Full Program- 4 marks** `>>> helloFile = open('C:\\Users\\your_home_folder\\hello.txt')` `>>> helloContent = helloFile.read()` `>>> helloContent` `'Hello world!'` `>>> sonnetFile = open('sonnet29.txt')` `>>> sonnetFile.readlines`	[4]	CO3	L3
4(a)	Write a Python Program to find an American phone number (example: 415-555-4242) in a given string using Regular Expressions. **Full Program- 4 marks** import re print("Enter a string") x = input() phoneRegex = re.compile(r'\d\d\d-\d\d\d-\d\d\d\d') phoneRegex.findall(x)	[4]	CO3	L3
4(b)	Describe the difference between Python **os** and **os.path** modules. Also, discuss the following methods of **os** module a) chdir() b) rmdir() c) walk() d) listdir() e) getcwd() os module →It is from the standard library and it provides a portable way of accessing and using operating system dependent functionality. os.path module → It is used to manipulate the file and directory paths and path names. **Difference [2 Marks]** **[each function with example 4 marks]** a) chdir( ) → This method is used to change from current working directory to the required directory. Ex: import os os.chdir("C:\\Windows\\System") b) rmdir( ) → This method is used to delete the folder at path. This folder must not contain any files or folders. c) walk( ) → This method is used to access or do any operation on each file in a folder. Ex: import os for foldername, subfolder, filenames in os.walk("C:\\Windows\\System") print("The current folder is "+foldername) d) listdir( ) → This method returns a list containing the names of the entries in the directory given by path. e) getcwd( ) → This method is used to get the current working directory.	[6]	CO3	L3

	Ex: import os os.getcwd( ) os module →It is from the standard library and it provides a portable way of accessing and using operating system dependent functionality.			
5 (a)	Define assertions. What does an assert statement in python consists of? Explain how assertions can be used with Python code snippet. **[Explanation 2 marks + code example 2 marks]** ✓ **Assertion:** An assertion is a sanity check to make sure the code isn't doing something obviously wrong. ✓ If the sanity check fails, then an AssertionError exception is raised. ✓ In python, an assert statement consists of the following: 1. The assert keyword 2. A condition (that is, an expression that evaluates to True or False) 3. A comma 4. A string to display when the condition is False ✓ In plain English, an assert statement meaning is, "I assert that this condition holds true, and if not, there is a bug somewhere in the program." ✓ **Traffic Light Simulation:** The data structure representing the stoplights at an intersection is a dictionary with keys 'ns' and 'ew', for the stoplights facing north-south and east-west, respectively. The values at these keys will be one of the strings 'green', ' yellow', or 'red'. `market_2nd = {'ns': 'green', 'ew': 'red'}` `mission_16th = {'ns': 'red', 'ew': 'green'}` These two variables will be for the intersections of Market Street and 2nd Street, and Mission Street and 16th Street. The switchLights() function, will take an intersection dictionary as an argument and switch the lights. **This example can also be given** >>> ages = [26, 57, 92, 54, 22, 15, 17, 80, 47, 73] >>> ages.sort() >>> ages [15, 17, 22, 26, 47, 54, 57, 73, 80, 92] >>> assert ages[0] <= ages[-1] # Assert that the first age is <= the last age.	[4]	CO3	L2
5 (b)	USN is University Seat Number which contains 1 digit region code, followed by 2 characters College code, 2 digit year and 2 character branch code and 3 digit roll number Write a Python program to read n number of strings and print all the strings which are USN. **Full Program [6 marks]** Regex: `import re` `n=int(input())` `i=0` `lst=[]` `while i<n:` `  item=input()` `  lst=lst+ [item]` `  i=i+1` `USNRegex = re.compile(r'\d\w\w\d\d\w\w\d\d\d')` `for i in lst:`	[6]	CO3	L3

	`` if USNRegex.search(i)!=None: print(i) ```			
6(a)	Explain the way how extracting ZIP file can be done in Python Program using code snippet **Extracting ZIP Files: [Explanation 2Marks Example Code 4 Marks]** ✓ The extractall() method for ZipFile objects extracts all the files and folders from a ZIP file into the current working directory. ✓ After running the below code, the contents of example.zip will be extracted to C:\. ``` >>> import zipfile, os >>> os.chdir('C:\\') # move to the folder with example.z >>> exampleZip = zipfile.ZipFile('example.zip') >>> exampleZip.extractall() >>> exampleZip.close() ``` ✓ The extract() method for ZipFile objects will extract a single file from the ZIP file. ``` >>> exampleZip.extract('spam.txt') 'C:\\spam.txt' >>> exampleZip.extract('spam.txt', 'C:\\some\\new\\folders') 'C:\\some\\new\\folders\\spam.txt' >>> exampleZip.close() ```	[6]	CO3	L2
6(b)	What is send2trash module? How to safely delete files using this module? **[2 +2 Marks] Explanation and code required.** Using send2trash, we can send files to the Trash or Recycle Bin instead of permanently deleting them. The OS module's unlink(), remove() and rmdir() functions can be used to delete files or folders. But, these functions delete the files permanently. The operations cannot be undone if there were any accidental deletions performed. This can be prevented using **send2trash** ``` import send2trash send2trash.send2trash("/location/to/file") ```	[4]	CO3	L2

HoD Signature CCI signature Course Instructor