Question Bank for the subject
Python Application Programming (15CS664)

Module 2 (Questions on Files):

1. What is a file handler? Discuss its usage in file handling.
2. Explain *open()* function with syntax and example.
3. Discuss pros and cons of *read()* function on files.
4. Discuss *write()* function on files with syntax and example.
5. What is the common run-time error faced while opening a file? How do you handle it? Explain with example.
6. What is the need of *close()* function?

Module 3:
Lists:

1. Define list. How do you create an empty list? Give example.
2. Discuss different ways of traversing a list.
3. Explain + and * operations on list objects with suitable examples.
4. Discuss various built-in methods in lists.
5. What are the different ways of deleting elements from a list? Discuss with suitable functions.
6. How do you convert a list into a string and vice-versa? Illustrate with examples.
7. Write a short note on:
    a. Parsing lines
    b. Object Aliasing
8. Write the differences between
    a. sort() and sorted() b) append() and extend() c) join() and split()
9. When do we encounter TypeError, ValueError and IndexError?
10. What are identical and equivalent objects? How are they identified? give examples.
11. Consider the list scores = [5, 4, 7, 3, 6, 2, 1] and write the python code to perform the following operations:
    i) Insert an element 9 at the beginning of the list.
    ii) Insert an element 8 at the index position 3 of the list.
    iii) Insert an element 7 at the end of the list.
    iv) Delete all the elements of the list.
    v) Delete an element at the index position 3.
    vi) Delete an element at the beginning of the list.

Dictionary and Tuples:

12. Define a dictionary type in Python. Give example.
13. Explain ***get()*** function in dictionary with suitable code snippet.
14. Discuss the dictionary methods *keys()* and *items()* with suitable programming examples.
15. Explain various steps to be followed while debugging a program that uses large datasets.
16. Briefly discuss key-value pair relationship in dictionaries.

17. Define a tuple. Give an example to illustrate creation of a tuple.
18. What is mutable and immutable objects? Give examples.
19. Explain List of Tuples and Tuple of Lists.
20. How do you create an empty tuple and a single element tuple?
21. Explain DSU pattern with respect to tuples. Give example
22. Store the following data pairs in a list, a tuple, and a dictionary:

India	23. 91
USA	24. 1
UK	25. 41
Japan	26. 9

23. How do you create a tuple using a string and using a list? Explain with example.
24. Explain the concept of tuple-comparison. How tuple-comparison is implemented in *sort()* function? Discuss.
25. Write a short note on:
    a. Tuple assignment
    b. Dictionaries and Tuples
26. How tuples can be used as a key for dictionaries? Discuss with example.
27. Discuss pros and cons of various sequences like lists, strings and tuples.
28. How do you debug a program containing multiple data structures with possibility of shape errors? Discuss the steps involved.

Regular Expressions:

29. What is the need of regular expressions in programming? Explain.
30. Discuss any 5 meta characters used in regular expressions with suitable example.
31. Discuss *search()* and *findall()* functions of *re* module.
32. What is the need of escape characters in regular expressions? Give suitable code snippet.

Programming Questions:

On File:

1. Write a program to count number of lines in a file and to display the line and total count.
2. Write a program to read n number of lines from the keyboard and to write them onto a file.
3. Write a program to search and print all the lines starting with a specific word (taken as keyboard input) in a file.
4. Write a Python program to count number of characters, words and lines in a file
5. Write a Python program to first 10 lines and last 10 lines in a file.
6. Write a python program to read a file and write into another file.

On Lists:

7. Write a python program to create a list and print all the items in reverse index order.
8. Write a Python function that takes two lists and returns True if they have at least one common member.
9. Write a python function to check whether the list is Ascending order or not.
10. Write a Python function that takes two sorted lists and return merged List.
11. Write a Python program to remove duplicates from a list

12. Write a Python program to check a list is empty or not.
13. Write a Python program to print a the even indexed elements.
14. A list rotation consists of taking the last element and moving it to the front. For instance, if we rotate the list [1,2,3,4,5], we get [5,1,2,3,4]. If we rotate it again, we get [4,5,1,2,3]. Write a Python function ***rotatelist(ls,k)*** that takes a list ls and a positive integer k and returns the list ls after k rotations. If k is not positive, your function should return ls unchanged. Note that your function should not change ls itself, and should return the rotated list.

    Here are some examples to show how your function should work.

    ```
    >>> rotatelist([1,2,3,4,5],1)   #output is [5, 1, 2, 3, 4]
    >>> rotatelist([1,2,3,4,5],3)  #output is [3, 4, 5, 1, 2]
    >>> rotatelist([1,2,3,4,5],12)  #output is [4, 5, 1, 2, 3]
    ```

15. Define a Python function ***ascending(ls)*** that returns True if each element in its input list is at least as big as the one before it. For empty list, it should be True. Here are some examples to show how your function should work.

    ```
    >>> ascending([])              #returns True
    >>> ascending([3,3,4])         #returns True
    >>> ascending([7,18,17,19])    #returns False
    ```

16. Define a Python function ***alternating(ls)*** that returns True if the values in the input list alternately go up and down (in a strict manner). For empty list, it should be True

    For instance:

    ```
    >>> alternating([])                  #True
    >>> alternating([1,3,2,3,1,5])       #True
    >>> alternating([3,2,3,1,5])         #True
    >>> alternating([3,2,2,1,5])         #False
    >>> alternating([3,2,1,3,5])         #False
    ```

On Dictionary and Tuples

17. Write a program to count frequency of each character in an input string, using dictionary.
18. Write a program to count frequency of words in a given file, using dictionaries. Ignore the punctuation marks attached to the words in file and treat lowercase and uppercase letters as the same.
19. Consider a dictionary with strings as keys and numbers as values. Write a program to sort the elements of this dictionary based on keys.
20. Read a string from keyboard input. Create a list containing tuples, where each tuple represents a word in the input string and length of that string. Write a program sort the words in descending order of their length.
21. Write a program to display most frequent 10 words in a text file. (Hint: use dictionary, list and tuple)
22. Write a python program to input a line of text name and find the frequency of each word.
23. Write a python program to display value and key of a dictionary using a tuples
24. Write a python program to read first name, last name and age, create a dictionary based on name as key and age as value.

On Regular Expressions:

25. Write a program to extract only email-ID's in a text file. Use suitable regular expression.
26. Write a program to validate USN (both UG and PG) of VTU students. (Hint: UG USN format: 1RN15EC001, PG USN format: 1RN15MCA01)
27. Write a program to validate a mobile number in India in the form of +91 99999 99999 ((Note the white spaces after +91 and after first 5 digits)
28. Write a program to extract string from a file where the string should start with only uppercase letters and should end with a digit, and must contain a special character ^, anywhere in-between.
29. Python program that matches a string that has an 'a' followed by anything, ending in 'b'
30. Python program that display lines that do not contain number in a text file.