23ITB202-PYTHON PROGRAMMING

QUESTION BANK

Unit 1 - Part A

1. List the various control flow structures.
2. Define algorithm.
3. Distinguish between pseudocode and flowchart.
4. Write the building blocks of algorithms.
5. Discuss different modes of operation in Python.
6. Write a simple Python program to perform the addition of two values.
7. Distinguish between string and list data types.
8. Infer how the Python interpreter works.
9. Write an algorithm for basic arithmetic operations.
10. Evaluate the order of precedence of operators in Python.
11. State Tuple Assignment.
12. Develop an algorithm to convert temperature in Celsius to Fahrenheit.
13. Define Variable.
14. Show how comments are used in Python.
15. Examine a simple pseudocode to print n integers.
16. What is a data type in Python? List the various data types.
17. Draw the flowchart for calculating simple interest.
18. Write the pseudocode for the greatest among two numbers.
19. Mention the features of Python.
20. Discuss problem-solving techniques.
21. Classify expressions by applying different operators.
22. List the basic symbols used in drawing the flowchart for sequence control structure.
23. Classify different types of statements in Python.
24. List the types of operators available in Python.

Unit 1 - Part B

1. Analyze the model of the interpreter and explain how Python works in different modes.
2. Evaluate the different values (data types) and types of values that can be used in Python.
3. Summarize the advantages and disadvantages of flowcharts.
4. List the different operators in Python and estimate the precedence of execution.

5. Explain pseudocode and its rules, and give examples for sequence, selection, and repetition-type problems.

6. List the types of operators in Python and explain the different expressions involved in Python.

7. With a neat sketch, explain the following building blocks of algorithms: Statements, Control flow.

8. Analyze the need for functions and explain with an example.

9. Develop a flowchart to check whether the given number is a prime number or not. Develop pseudocode to perform arithmetic operations.

10. What is an algorithm? List the characteristics of a good algorithm. Write an algorithm to find the square root of a number.

11. Design a flowchart and write an algorithm that calculates the salary of an employee.

12. Write an algorithm and pseudocode for the following: (i) Calculating area and circumference of a circle. (ii) Check if a given year is a leap year or not.

13. Explain sequence data types in Python with examples.

14. Discuss the various modes of the Python interpreter and explain with an example.

15. Summarize the difference between an algorithm, flowchart, and pseudocode.

16. Explain the following: Tuple assignment, Comments, Statements in Python.

17. Using a simple Python snippet, analyze different values, types, and expressions, and explain them.

Unit 1 - Part C

1. Write an algorithm, pseudocode, and draw the flowchart to find the factorial of a number n.

2. Design a calculator with Python code by defining its algorithm using different notations.

3. Write a program to explain the various operators involved in Python and how an expression is evaluated using the precedence of operators.

4. Write an algorithm, pseudocode, and draw the flowchart to check whether the given number is a palindrome or not.

5. Rate the order of execution of different expressions by evaluating them through a Python program.

Unit 2 - Part A

1. Define function and state its use.

2. What does recursive function imply?

3. Analyze the different kinds of arguments.

4. Evaluate the importance of fruitful functions.

5. Analyze the need to divide a program into functions.

6. Write a program to print n numbers iteratively using a function.

7. Using the concept of functions, calculate the area of a circle.

8. Using the concept of tuple assignment, how will you swap two values?

9. What do you mean by fruitful function?

10. Outline the scope of variables.

11. Write the syntax of if-else statements.

12. Differentiate between for loop and while loop.

13. List any three built-in functions and their usage.

14. Write a function without argument and with return type.

15. Differentiate local and global variables.

16. Write a program to find the sum of the digits of a number.

17. Illustrate the flowchart of if-elif-else statements.

18. How would you test the significance of a for loop with else in an example?

19. Present the flow of execution for a while statement.

20. Write the syntax for function definition.

21. Name the type of Boolean operators.

22. Describe the break statement with an example.

23. Write a program to find the square root of a given number.

24. Give the syntax for pass and continue statements.

Unit 2 - Part B

1. Using the concept of control structure, determine the prime numbers in a given range using Python.

2. Write a program to determine the factorial of a given number with and without the use of recursion.

3. What does a fruitful function refer to? How can it be used? Explain with an example.

4. Write the syntax and explain the concept of a recursive function with an example. Write a Python code to search for an element using linear search.

5. List the different types of arguments/parameters used in functions with suitable examples.

6. Using a Python program, analyze the different logic behind swapping the values between variables.

7. Write a program to find the square root of a number by iterative Newton's method.

8. Explain the looping statements (while and for loops) with an example.

9. List the different types of conditional control statements and explain them with a suitable example.

10. Describe the concept of binary search with a suitable example and write Python code to implement it.

11. Write a Python program to find the GCD of two given numbers. Write a Python program to find the exponent of a number using recursion.

12. Write a Python program to find the greatest among three numbers. Write a program to check if a given number is an Armstrong number or not.

13. Explain break and continue statements using the while loop.

14. Write a Python code to print all numbers in a range (a, b) divisible by a given number (n).

15. What is a function? How is a function defined and called in Python? Explain with a simple program.

16. Write a Python program to find the sum of N natural numbers. What is the use of the pass statement? Illustrate with an example.

17. Write a Python program using a function to find the sum of the first 'n' even numbers and print the result. Write a Python program to find the roots of the quadratic equation.

Unit 2 - Part C

1. Create a user-defined fruitful function to test if a given year is a leap year.

2. Write a function to determine whether a given natural number is a perfect number.

3. Write a Python program to implement a student mark system using chained conditional if control structure. Write a Python program to check if a given number is positive, negative, or zero using nested if conditional control structure.

4. Write a function that reads two numbers and evaluates whether they are co-prime or not.

5. Write a function to multiply two non-negative numbers by repeated addition and evaluate the result by normal procedure.

Unit-3

Part A

1. Define Python list.

2. Mention the list operations.

3. What are the different ways to create a list?

4. Illustrate negative indexing in a list with an example.

5. How to slice a list in Python?

6. Point out the methods that are available with the list object.

7. Show the membership operators used in a list.

8. Define Python Tuple.

9. Write a program to add two lists.

10. Classify the Python accessing elements in a Tuple.

11. Point out the methods used in Tuples.

12. How tuple as arguments to a function? Give example.

13. Write the syntax and purpose of the insert() method in a list.

14. How for loop is used in sequence objects? Give example.

15. Point out the advantages of a tuple.

16. Evaluate the difference between lists and tuples.

17. Show how Tuples are used as return values?

18. What does sorting refer to?

19. What does the term mutability refer to?

20. Write a program to create a list of even numbers in a given range.

21. Write the syntax for concatenating two lists in Python.

22. Show how Tuples are immutable?

23. List the different sorting algorithms.

24. With the help of a program, list the different methods in a list.

Part B

1. Discover an algorithm and write a Python program to sort the numbers in ascending order using insertion sort.

2. Describe the following: Creating the list, Accessing values in the lists, Updating the list, Deleting the list elements.

3. Analyze the basic list operations in detail with necessary programs. Write a Python program to add two matrices.

4. Mention the Python list methods with examples. Why is it necessary to have both the functions append() and extend()? What is the result of the following expression that uses append() where it probably intended to use extend()?

5. Demonstrate the working of +, and slice operators in Python lists and tuples.

6. What is a Python Tuple? What are the advantages of Tuple over a list? "Tuples are immutable." Explain with an example.

7. Illustrate the ways of creating the Tuple and the Tuple assignment with suitable programs.

8. What are the accessing elements in a Tuple? Explain with suitable programs. Explain how to return more than one value from a function with the help of a program.

9. (i) Explain the basic Tuple operations with examples. (ii) Illustrate a program to check whether an element ‘y’ and ‘a’ belong to the tuple mytuple = ('p', 'y', 't', 'h', 'o', 'n') and after printing the result, delete the Tuple.

10. Write a program to perform the following matrix operations: (i) Addition of two matrices (ii) Transpose of a matrix.

11. How is a list used in loops? Give an example for a while loop and a for loop used in lists.

12. Given a tuple test_tup = (4, 5, 4, 5, 6, 6, 5), write a program to find the frequency of each element.

13. Describe the built-in functions with Tuples and write a program to use Max(), Min(), and sorted() methods in Tuple.

14. (i) Discuss a) Tuples as return values b) Variable Length Argument Tuples. (ii) Write a program to illustrate the comparison operators in Tuple.

15. Write a Python program to store 'n' numbers in a list and sort the list using selection sort.

16. Using functions and methods, analyze the differences and similarities of lists and tuples with examples for each.

17. Write a program to perform the logic of the quicksort algorithm.

Part C

1. Write a function that takes a list of numbers as input from the user and produces the corresponding cumulative list.

2. Write a function to perform sorting of given numbers and present a list of odd and even numbers separately.

3. Write a Python program to add a tuple to a list and vice-versa.

4. Write a function 'leftCirculate' that takes a list as an input and left circulates the values in the list so that in the final list, each value is left shifted by one position and the leftmost value in the original list now appears as the rightmost value.

5. Write a program to delete all the duplicate elements in a list.

UNIT – 4

Unit 4 - Part A

1. Point out different modes of file opening.

2. Define the access modes.

3. Distinguish between files and modules.

4. Define read and write file.

5. Describe renaming and deleting a file in Python.

6. Discover the format operator available in files.

7. Examine the need for exceptions using an example.

8. Explain Built-in exceptions.

9. Difference between built-in exceptions and handling exception.

10. Write a program to write data in a file for both write and append modes.

11. How to import statements?

12. Find the error in the code given: `while True print('Hello world')`.

13. Define package.

14. What are packages in Python?

15. Write the difference between `read()` and `read(n)` functions?

16. Accept five names from the user and write in a file “name.txt”.

17. What is ‘value error’ in Python?

18. What is base exception?

19. Examine buffering.

20. What do you mean by `fileisatty()` method?

21. Discover `except` clause with multiple exceptions.

22. Create a Python script to display the current date and time.

23. Analyze the object as return values.

24. Discuss a modular design.

Unit 4 - Part B

1. Write a Python program to demonstrate file I/O operations.

2. Discuss the different modes for opening and closing a file.

3. Discover a program to catch a divide by zero exception. Add a finally block too. Write a function to print the hash of any given file in Python.

4. Describe the use of try and except blocks in Python with syntax. Describe exceptions with arguments in Python with an example.

5. Explain with an example of writing to a file. Discover syntax for reading from a file.

6. Structure renaming a file. Explain file-related methods.

7. Describe Python modules. Describe Python packages.

8. Write a program that will prompt the user for a string and a file name, and then print all lines in the file that contain the string. Also, interpret the obtained result.

9. Identify the various methods used to delete elements from the dictionary.

10. Describe in detail exception handling with a sample program.

11. Illustrate a program to find the one’s complement of a binary number using a file.

12. Write a program to display a pyramid.

13. Explain the terminology of raising an exception concept with a sample program.

14. Write a program to count the total number of uppercase characters in a file in Python.

15. Explain the following: (i) Predefined Modules (ii) User-defined Modules.

16. Write a program to find the number of instances of different digits in a given number.

17. Explain with an example to copy the contents of one file to another.

Unit 4 - Part C

1. Create a program to compute the price per unit weight of an item using try–except–else block.

2. Write a program that reads the contents of the file `text.txt` and counts the number of alphabets, blank spaces, lowercase letters, uppercase letters, the number of words starting with a vowel, and the number of occurrences of the word ‘is’ in the file.

3. Examine the following function `percentage`:

```python
def percentage(marks, total):
    try:
        percent = (marks / total)  100
    except ValueError:
        print('ValueError')
    except TypeError:
        print('TypeError')
    except ZeroDivisionError:
        print('ZeroDivisionError')
    except:
        print('any other error')
    else:
        print(percent)
    finally:
        print('Function percentage completed')
```

Determine the output for the following function calls:

- a) `percentage(150.0, 200.0)`
- b) `percentage(150.0, 0.0)`
- c) `percentage(‘150.0’, ‘200.0’)`

4. Write a function that reads a file `file1` and evaluates and displays the number of words and vowels in the file.

5. Write a program to count the total number of lines and count the total number of lines starting with ‘A’, ‘B’, and ‘C’.

UNIT 5

Unit: Data Handling and Visualization

Part A

1. Describe the purpose and use of NumPy arrays.
2. Differentiate between NumPy arrays and Python lists.
3. Define basic operations on NumPy arrays.
4. Explain the concept of broadcasting in NumPy.
5. List and describe common functions for data manipulation in Pandas.
6. Illustrate how to select and filter data in a Pandas DataFrame.
7. Discuss the importance of data visualization in data analysis.
8. Explain the basic plotting functions available in Matplotlib.
9. Compare line plots, bar charts, and scatter plots in Matplotlib.
10. How do you read data from a CSV file into a Pandas DataFrame?
11. Describe the process of exporting a Pandas DataFrame to an Excel file.
12. Identify different data formats supported by Pandas for import and export.
13. How can you handle missing data in Pandas?
14. Define the concept of indexing and slicing in NumPy arrays.
15. Explain the use of Matplotlib for creating multi-line plots.
16. Describe the method to import an Excel file using Pandas.
17. What is the significance of using `groupby` in Pandas?
18. List the steps to create a histogram using Matplotlib.
19. How do you handle large datasets in Pandas efficiently?
20. Discuss the role of the `agg` function in data aggregation with Pandas.

Part B

1. Write a Python program to perform basic arithmetic operations on NumPy arrays.
2. Explain with an example how to manipulate data using Pandas DataFrame.
3. Write a Python script to visualize a dataset using a line plot in Matplotlib.
4. Create a Pandas DataFrame from a CSV file and demonstrate data cleaning techniques.
5. Discuss the process of merging and joining data in Pandas with examples.
6. Write a Python program to read data from an Excel file and plot a bar chart using Matplotlib.

7. Demonstrate the use of NumPy's `reshape` and `ravel` functions with an example.

8. Explain the concept of pivot tables in Pandas and write a program to create one.

9. Write a Python script to handle missing data in a dataset using Pandas.

10. Illustrate the process of creating a scatter plot with multiple datasets in Matplotlib.

11. Discuss the methods for reading data from different file formats (CSV, Excel, JSON) in Pandas.

12. Write a Python program to perform group-wise operations on a Pandas DataFrame.

13. Explain the use of subplots in Matplotlib with a code example.

14. Write a script to export a cleaned and processed dataset to a new CSV file using Pandas.

15. Discuss the significance of data visualization and create a pie chart using Matplotlib.

16. Write a Python program to normalize a dataset using NumPy.

17. Explain the process of filtering data in a Pandas DataFrame with logical conditions.

Part C

1. Create a program to load a large CSV file, manipulate the data using Pandas, and visualize key insights using Matplotlib.

2. Write a Python script to analyze a dataset by handling missing values, performing aggregation, and visualizing the results.

3. Design a Python function that imports data from an Excel file, cleans it, and visualizes the distribution of a key variable using Matplotlib.

4. Write a comprehensive program to demonstrate the process of reading, manipulating, and exporting data using Pandas, with visualization steps using Matplotlib.

5. Create a program that performs data manipulation on a dataset using Pandas and creates a dashboard of visualizations using Matplotlib subplots.