**Most important topic 1.python, 2.SQL then (powerbi, ML)**

**BASIC INTRODUCTIONS**

1. Introduce yourself
2. Briefs your projects
3. What are your strengths?
4. https://www.geeksforgeeks.org/hr-interview-questions/

**EXCEL QUESTIONS**

1. What is the use case of excel
2. What is the advantage and disadvantage of excel
3. Difference count , counta, countif
4. Sum ,sumif , sumifs.
5. IF, IFERROR
6. What is referencing? Relative References, Absolute References, Mixed References, Multiple Sheet References, Consolidating Data
7. What is name ranges?
8. Autofill and custom lists?
9. Sorting using Custom List
10. What is conditional formatting? types
11. What is reference functions
    1. What lookup or vlookup or hlookup or xlookup? Usecase, advantage, disadvantage,syntax, implement
    2. index match? Usecase, advantage, disadvantage,syntax, implement
12. What is data validations? Usecase, implement
13. What is Drop-Down Lists and how to create
14. What is pivot table?

A [pivot table](https://www.simplilearn.com/tutorials/excel-tutorial/pivot-table) is like a summary table of the dataset that enables you to create reports and analyze trends. They are useful when you have long rows or columns that hold values you need to track.

1. How do you freeze panel in excel?
2. Write a function to add pass/fail to the results column based on the following criteria. If student marks > 60 and attendance > 75%, then pass else the student fails.
3. Calculate your age in years from the current date. Use the YEARFRAC() or DATEDIF() function to return the number of whole days between start\_date and end\_date
4. Using the Coronavirus dataset, create a pivot table to find the total cases in each country belonging to their respective continents.
5. Create a pivot table to find the top three countries from each continent based on the total cases using COVID data.
6. How does a Slicer work in Excel?

To filter data in a Pivot table, we can use slicers.

1. To create a slicer, go to the Insert tab, and select Slicer present under Filter.
2. Then, select the list of fields for which you want to create slicers.
3. Use the coronavirus dataset to find the percentage contribution of each country and continent to the total cases?

Create the pivot table to show the total cases by country and continent.

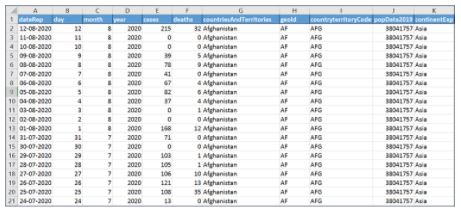
Right-click on the sum of cases column and under Show Value As, select “% of Grand Total.”

1. Using the COVID data, find the number of days in which the number of deaths in Italy has been greater than 200.-> COUNTIFS() function we use is =COUNTIFS(G2:G35777,"Italy",E2:E35777,">200")
2. Using the below-given sales data, highlight those cells where total sales > $5000.
3. Using the given table, explain how the index-match function works in Excel.
4. How do you find duplicate values in a column?1. Conditional Formatting

2. COUNTIF(range,a2)>1 COUNTIF($D$2:D2,D2) https://www.simplilearn.com/tutorials/excel-tutorial/remove-excel-duplicate

26.  How can you remove duplicate values in a range of cells?

1. To delete [duplicate values](https://www.simplilearn.com/tutorials/excel-tutorial/remove-excel-duplicate) in a column, select the highlighted cells, and press the delete button. After deleting the values, go to the ‘Conditional Formatting’ option present in the Home tab. Choose ‘Clear Rules’ to remove the rules from the sheet.
2. You can also delete duplicate values by selecting the ‘Remove Duplicates’ option under Data Tools present in the Data tab.



**# OPTIONAL**

1. How can you restrict someone from copying a cell from your worksheet?
2. How is a Formula different from a Function in Excel?
3. What is the shortcut to add a filter to a table
4. How can we merge multiple cells text strings in a cell
5. How can you split a column into 2 or more columns?
6. how to find unique values from a column?
7. How do you create a pivot chart in Excel?
8. What are macros in Excel? Create a macro to automate a task.
9. What is the What-If Analysis in Excel?

The What-If Analysis in Excel is a powerful tool to perform complex mathematical calculations, experiment with data, and try out different scenarios.

1. 1. What is the ribbon in excel?

The term "ribbon" refers to the region at the top of the application that houses the MS Excel toolbars and menu items. With CTRL+F1

1. Explain the difference between SUBSTITUTE and REPLACE in excel.
2. How do you create a column in a pivot table?

For this, you have to go to the PivotTable Analyze tab and select ‘Fields, Items & Sets’ option. Under that, you need to click ‘Calculate Field’ to create a new column.

1. Use the coronavirus dataset to find the percentage contribution of each country and continent to the total cases?
2. Create the pivot table to show the total cases by country and continent.

Right-click on the sum of cases column and under Show Value As, select “% of Grand Total.”

**POWERBI QUESTIONS**

1. What is Power BI? why use power bi
2. Difference between Power BI and Tableau
3. Difference between Power Query and Power Pivot
4. What is Power BI Desktop
5. What is Power Pivot?
6. What is Power Query?
7. Difference between Power Query and Power Pivot
8. What is DAX?
9. What are Filters in Power BI?
10. What are Custom Visuals in Power BI?
11. What is GetData in Power BI?
12. What are Filters in Power BI?
13. Mention some advantages of Power BI.
14. List out some drawbacks/limitations of using Power BI.
15. Name the different connectivity modes available in Power BI?
16. What are the various types of refresh options provided in Power BI?
17. What is a dashboard in Power BI?
18. Explain how relationships are defined in Power BI Desktop?
19. What is the CALCULATE function in DAX?
20. What are the different stages in the working of Power BI? There are three different stages in working on Power BI, as explained below. Data Integration Data Processing Data Presentation
21. What are KPIs in Power BI? KPI is abbreviated as Key Performance Indicator. Any professional organization has teams and employees follow the KPI protocols. The organizations set up KPIs for all the employees. These KPIs act as their targets. These KPIs are compared to previous performance and analyze the progress.
22. Difference between sum and sumx /
23. Difference BW average and averagea , averagex
24. Count ,counta, countx, countblank
25. What are the available views?
26. What are the major components of Power BI?
27. What are the types of visualizations in Power BI?

**TABLEAU QUESTIONS**

1. What is Tableau?
2. What are the advantages of Tableau?
3. <https://www.simplilearn.com/tableau-interview-questions-and-answers-article>
4. Difference bw tableau and powerbi
5. Limitation and advantage of tableau

**Python For Data Science QUESTIONS**

1. **Explain type casting in Python.**
2. What is the difference between a Shallow Copy and a Deep Copy?
3. What is the purpose of the pass statement?
4. How would you remove all leading Whitespace in a string?
5. How would you replace all occurrences of a Substring with a new string?
6. How do you display the contents of a text file in reverse order?
7. Differentiate between append() and extend().
8. 3way to create a list or set or dict , or tuple
9. What is the difference between a list and a tuple?
10. How do you use Print() without the newline?
11. What are \*args and \*kwargs?
12. What is slicing and indexing
13. How to reassign value in list in specific index
14. What are the essential features of Python?
15. What is lambda function ? Why Lambda is used in Python?
16. What is global and local variable?
17. What Difference between for loop and while loop in Python
18. What is oop? And what are 4 pillars
19. What is the use of self in Python?
20. What is \_init\_?
21. How does continue, break, and pass work?
22. What is arguments and types?
23. **What are the *built-in data types* in *Python*?**
24. **How do you *handle exceptions* in *Python*?**
25. Create a function to find a number is prime or not?
26. Create a function to print n odd number using while loop?
27. Write a Python program to check if a string is a palindrome.
28. Write a Python program to find the factorial of a number.
29. Write a Python program to swap two variables without using a temporary variable.
30. Write a Python program to find the length of a string without using the len() function.
31. Write a Python program to find the Fibonacci sequence up to a given number.
32. Write a Python program to check if a string is a Anagram or not.
33. Write a Python program to count the number of positive values given in the list.

[3,2,4,-4,-5]

1. Write a Python program to find the sum of odd values from the given list.

[3,2,4,-4,-5]

1. Write a program to check the frequency of occurrence of 4 in the given list.

[4,5,6,7,8,94,4]

1. Write a Python class to represent a BankAccount with methods to deposit, withdraw, and check the balance.
2. Write a Python class to represent a Student with attributes like name, roll\_number, and marks. Include a method to calculate the student's grade based on marks.
3. Write a Python class to demonstrate inheritance. Create a base class Animal and a derived class Dog.
4. Write a program to create a class named `Product`. The `Product` has the following attributes:

- Unique identification number: `id`

- Product Name: `name`

- Product Price: `price`

- Product Quantity: `qty`

Define the following methods and constructor:

1. Constructor to initialize the product ID.

2. Method to get product details such as name, price, and quantity.

3. Method to display a product.

**OPTIONAL**

1. What are the *key features* of *Python*?

**Python** is a versatile and popular programming language known for its simplicity, **elegant syntax**, and a vast ecosystem of libraries. Let's look at some of the key features that make Python stand out.

Key Features of Python

1. Interpreted and Interactive

Python uses an interpreter, allowing developers to run code **line-by-line**, making it ideal for rapid prototyping and debugging.

2. Easy to Learn and Read

Python's **clean, readable syntax**, often resembling plain English, reduces the cognitive load for beginners and experienced developers alike.

3. Cross-Platform Compatibility

Python is versatile, running on various platforms, such as Windows, Linux, and macOS, without requiring platform-specific modifications.

4. Modular and Scalable

Developers can organize their code into modular packages and reusabale functions.

5. Rich Library Ecosystem

The Python Package Index (PyPI) hosts over 260,000 libraries, providing solutions for tasks ranging from web development to data analytics.

**3. What is *PEP 8* and why is it important?**

**PEP 8** is a style guide for Python code that promotes code consistency, readability, and maintainability. It's named after Python Enhancement Proposal (PEP), the mechanism used to

**5.** **What are the *built-in data types* in *Python*?**

Python offers numerous **built-in data types** that provide varying functionalities and utilities.

Immutable Data Types

1. int

2. float

3. complex

4. bool

5. str

6. tuple.

7. frozenset

8. bytes

9. bytearray

10. NoneType

Mutable Data Types

1. list

2. set

3. dict

**7.** **How do you *handle exceptions* in *Python*?**

**Exception handling** is a fundamental aspect of Python, and it safeguards your code against unexpected errors or conditions. Key components of exception handling in Python include:

Components

* **Try**: The section of code where exceptions might occur is placed within a try block.
* **Except**: Any possible exceptions that are raised by the try block are caught and handled in the except block.
* **Finally**: This block ensures a piece of code always executes, regardless of whether an exception occurred. It's commonly used for cleanup operations, such as closing files or database connections.

Generic Exception Handling vs. Handling Specific Exceptions

It's good practice to **handle** specific exceptions. However, a more **general** approach can also be taken. When doing the latter, ensure the general exception handling is at the end of the chain, as shown here:

try:  
    risky\_operation()  
except IndexError:  # Handle specific exception types first.  
    handle\_index\_error()  
except Exception as e:  # More general exception must come last.  
    handle\_generic\_error()  
finally:  
    cleanup()

def divide(a, b):  
    if b == 0:  
        raise ZeroDivisionError("Divisor cannot be zero")  
    return a / b  
  
try:  
    result = divide(4, 0)  
except ZeroDivisionError as e:  
    print(e)

**8. What is the difference between *list* and *tuple*?**

**Lists** and **Tuples** in Python share many similarities, such as being sequences and supporting indexing.

However, these data structures differ in key ways:

Key Distinctions

* **Mutability**: Lists are mutable, allowing you to add, remove, or modify elements after creation. Tuples, once created, are immutable.
* **Performance**: Lists are generally slower than tuples, most apparent in tasks like iteration and function calls.
* **Syntax**: Lists are defined with square brackets [], whereas tuples use parentheses ().

When to Use Each

* **Lists** are ideal for collections that may change in size and content. They are the preferred choice for storing data elements.
* **Tuples**, due to their immutability and enhanced performance, are a good choice for representing fixed sets of related data.

Syntax

List: Example

my\_list = ["apple", "banana", "cherry"]  
my\_list.append("date")  
my\_list[1] = "blackberry"

Tuple: Example

my\_tuple = (1, 2, 3, 4)  
# Unpacking a tuple  
a, b, c, d = my\_tuple

**9. How do you create a *dictionary* in *Python*?**

**Python dictionaries** are versatile data structures, offering key-based access for rapid lookups. Let's explore various data within dictionaries and techniques to create and manipulate them.

Key Concepts

* A **dictionary** in Python contains a collection of key:value pairs.
* **Keys** must be unique and are typically immutable, such as strings, numbers, or tuples.
* **Values** can be of any type, and they can be duplicated.

Creating a Dictionary

You can use several methods to create a dictionary:

1. **Literal Definition**: Define key-value pairs within curly braces { }.
2. **From Key-Value Pairs**: Use the dict() constructor or the {key: value} shorthand.
3. **Using the dict() Constructor**: This can accept another dictionary, a sequence of key-value pairs, or named arguments.
4. **Comprehensions**: This is a concise way to create dictionaries using a single line of code.
5. **zip() Function**: This creates a dictionary by zipping two lists, where the first list corresponds to the keys, and the second to the values.

Examples

Dictionary Literal Definition

Here is a Python code:

# Dictionary literal definition  
student = {  
    "name": "John Doe",  
    "age": 21,  
    "courses": ["Math", "Physics"]  
}

10. What is the difference between *==* and *is operator* in *Python*?

Both the **==** and **is** operators in Python are used for comparison, but they function differently.

* The **==** operator checks for **value equality**.
* The **is** operator, on the other hand, validates **object identity**,

In Python, every object is unique, identifiable by its memory address. The **is** operator uses this memory address to check if two objects are the same, indicating they both point to the exact same instance in memory.

* **is**: Compares the memory address or identity of two objects.
* **==**: Compares the content or value of two objects.

**11. How does a *Python function* work?**

**Python functions** are the building blocks of code organization, often serving predefined tasks within modules and scripts. They enable reusability, modularity, and encapsulation.

Key Components

* **Function Signature**: Denoted by the def keyword, it includes the function name, parameters, and an optional return type.
* **Function Body**: This section carries the core logic, often comprising conditional checks, loops, and method invocations.
* **Return Statement**: The function's output is determined by this statement. When None is specified, the function returns by default.
* **Local Variables**: These variables are scoped to the function and are only accessible within it.

**12. What is a *lambda function*, and where would you use it?**

A **Lambda function**, or **lambda**, for short, is a small anonymous function defined using the lambda keyword in Python.

While you can certainly use named functions when you need a function for something in Python, there are places where a lambda expression is more suitable.

Distinctive Features

* **Anonymity**: Lambdas are not given a name in the traditional sense, making them suited for one-off uses in your codebase.
* **Single Expression Body**: Their body is limited to a single expression. This can be an advantage for brevity but a restriction for larger, more complex functions.
* **Implicit Return**: There's no need for an explicit return statement.
* **Conciseness**: Lambdas streamline the definition of straightforward functions.

Common Use Cases

* **Map, Filter, and Reduce**: Functions like map can take a lambda as a parameter, allowing you to define simple transformations on the fly. For example, doubling each element of a list can be achieved with list(map(lambda x: x\*2, my\_list)).
* **List Comprehensions**: They are a more Pythonic way of running the same map or filter operations, often seen as an alternative to lambdas and map.
* **Sorting**: Lambdas can serve as a custom key function, offering flexibility in sort orders.
* **Callbacks**: Often used in events where a function is needed to be executed when an action occurs (e.g., button click).
* **Simple Functions**: For functions that are so basic that giving them a name, especially in more procedural code, would be overkill.

**13. Explain *\*args* and *\*\*kwargs* in *Python*.**

In Python, \*args and \*\*kwargs are often used to pass a variable number of arguments to a function.

\*args collects a variable number of positional arguments into a **tuple**, while \*\*kwargs does the same for keyword arguments into a **dictionary**.

Here are the key features, use-cases, and their respective code examples.

**\*args**: Variable Number of Positional Arguments

* **How it Works**: The name \*args is a convention. The asterisk (\*) tells Python to put any remaining positional arguments it receives into a tuple.
* **Use-Case**: When the number of arguments needed is uncertain.

Code Example: "\*args"

def sum\_all(\*args):  
    result = 0  
    for num in args:  
        result += num  
    return result  
  
print(sum\_all(1, 2, 3, 4))  # Output: 10

**\*\*kwargs**: Variable Number of Keyword Arguments

* **How it Works**: The double asterisk (\*\*) is used to capture keyword arguments and their values into a dictionary.
* **Use-Case**: When a function should accept an arbitrary number of keyword arguments.

Code Example: "\*\*kwargs"

def print\_values(\*\*kwargs):  
    for key, value in kwargs.items():  
        print(f"{key}: {value}")  
  
# Keyword arguments are captured as a dictionary  
print\_values(name="John", age=30, city="New York")  
# Output:  
# name: John  
# age: 30  
# city: New York

**14. What are *decorators* in *Python*?**

In Python, a **decorator** is a design pattern and a feature that allows you to modify functions and methods dynamically

**15. How can you create a *module* in *Python*?**

You can **create** a Python module through one of two methods:

* **Define**: Begin with saving a Python file with .py extension. This file will automatically function as a module.
* **Create a Blank Module**: Start an empty file with no extension. Name the file using the accepted module syntax, e.g., \_\_init\_\_, for it to act as a module.

Next, use **import** to access the module and its functionality.

Code Example: Creating a math\_operations Module

Module Definition

Save the below math\_operations.py file :

def add(x, y):  
    return x + y  
  
def subtract(x, y):  
    return x - y  
  
def multiply(x, y):  
    return x \* y  
  
def divide(x, y):  
    return x / y

Module Usage

You can use math\_operations module by using import as shown below:

import math\_operations  
  
result = math\_operations.add(4, 5)  
print(result)  
  
result = math\_operations.divide(10, 5)  
print(result)

**NUMPY**

* 1. What is NumPy? Why should we use it?
  2. Write the steps to create 2D and 3D arrays with output.
  3. What's the difference between a Python list and a NumPy array?
  4. How do you check the shape and size of a NumPy array?
  5. How do you reshape a NumPy array?
  6. How do you create an array of all zeros or all ones?
  7. What is broadcasting in NumPy?
  8. How do you find the mean, median, and standard deviation of a NumPy array?
  9. How can you handle and manipulate arrays with missing or infinite values in NumPy?
  10. Is there a way to quickly and easily apply functions to each row and column of a 2D array?
  11. What are some ways we can easily sort
  12. How do you concatenate 2 NumPy arrays?
  13. array indexing and slicing
  14. What are ways of creating 1D, 2D and 3D arrays in NumPy?
  15. What do you understand by Vectorization in NumPy?
  16. Function Vectorization technically means that the function is applied to all elements in the array.
  17. What are the main features of Numpy?
  18. What are some of the limitations of NumPy.

**Pandas Interview Questions and Answers**

* 1. What is pandas in Python?
  2. How do you quickly access the top 5 rows and last 5 rows of a pandas DataFrame?
  3. What is the difference between a Series and a DataFrame?
  4. What is an index in pandas?
  5. What is multi indexing in pandas?
  6. Merge the two DataFrames.
  7. Find the average
  8. How do you sort a DataFrame based on columns?
  9. Show two different ways to filter data
  10. How do you aggregate data and apply some aggregation function like mean or sum on it?
  11. How can you create a new column derived from existing columns?
  12. How do you handle null or missing values in pandas?
  13. Difference between fillna() and interpolate() methods
  14. What is resampling?
  15. How do you perform one-hot encoding using pandas?--> get\_dummenis
  16. What is the pandas method to get the statistical summary of all the columns in a DataFrame?
  17. How to Read Text Files with Pandas?
  18. How are iloc() and loc() different?
  19. How will you sort a DataFrame?
  20. fillna() in Pandas?
  21. How to reset the index in a Python Pandas DataFrame?
  22. How to rename the index in a Pandas DataFrame?
  23. How to delete a column in Pandas DataFrame?
  24. How to delete a row in Pandas DataFrame?
  25. How can we convert DataFrame into a NumPy array?
  26. How to add a column to a Pandas DataFrame?
  27. How to add a row to a Pandas DataFrame?
  28. groupby()
  29. How do you plot a histogram using seaborn?
  30. How do you plot a barplot using seaborn?
  31. How do you plot a categorical plot using seaborn?
  32. How do you plot a heatmap using seaborn?
  33. What are Python lists and tuples?
  34. How do you identify and deal with missing values?
  35. Can you remove duplicates from a sorted array?

**SQL**

1. What is SQL?
2. What is a database?
3. What are the main types of SQL commands?
4. What is a primary key?
5. What is a foreign key?
6. What is the purpose of the DEFAULT constraint?
7. What is normalization in databases?
8. Normalization is the process of organizing data in a database to reduce redundancy and improve data integrity. This involves dividing large tables into
9. smaller, related tables and defining relationships between them to ensure consistency and avoid anomalies.
10. What are the different operators available in SQL?
11. What are the different types of joins in SQL?
12. What is the purpose of the GROUP BY clause?
13. What are aggregate functions in SQL?
14. What is a subquery?
15. What is the difference between the WHERE and HAVING clauses?
16. What is the difference between DELETE and TRUNCATE commands?
17. What is the purpose of the SQL ORDER BY clause?
18. What are the differences between SQL and NoSQL databases?
19. What is the purpose of the SQL SELECT statement?
20. What is the difference between DDL and DML commands?
21. how to perform join query,
22. how to perform where, having, aggrigate, group by ,limit, between, in , like

**ML Interview Questions and Answers**

1. What is machine learning
2. ML Modeling Flow or life cycle
3. What is Prametric and Non Parametric Algorithm
4. Types of ML
5. What is EDA?
6. How to Handling Missing Values
7. How to Handling Outliers
8. How to Handling Skewness
9. How to do Data Encoding
10. What is Feature Scaling -Normalization and Standardization
11. What is Feature Engineering
12. What is linear regression
13. What are the assumptions of Linear Regression?
14. What is Measuring Performance metrics-Lost and Cost Function (MAE,MSE,RMSE,R2 Score)
15. What is evaluation metrix?
16. What is regularization? L1 and L2?
17. What is cross validations
18. What is Logistic Regression?
19. What evaluations metrics use in classifications definition(Performance Metrics :Confusion Matrix, Precision ,Recall,F1 score ,ROC AUC etc.)
20. Bias and Variance Trade off
21. Overfitting and Underfitting
22. What is Decision tree? And how it works?

**Optional**

1. What is Entropy and Information Gain, gini index in DT?
2. Pruning Techniques in DT
3. What isEnsemble Techniques( bagging and boosting)?
4. What is random forest?
5. What is KNN and how to choose k?
6. What is SVM? Working of SVM
7. What are support vectors & Hyperplace
8. What is Kernels and Types of Kernel
9. What is Hard Margin & Soft Margin
10. What is Naïve Bayes?
11. What is PCA
12. What is unsupervised Learning?
13. What is K-Means Clustering?
14. Choosing the optimum K value (Elbow Method)
15. Various Distance Measure
16. What is Hirarchical Clustering ?