**Python Basics and Data Structures:**

1. Difference between List and Tuple?

|  |  |  |
| --- | --- | --- |
| Criteria | List | Tuple |
| |  |  |  | | --- | --- | --- | | **Mutability** |  |  | | Mutable (elements can be changed) | Immutable (elements cannot be changed) |
| **Syntax** | Square brackets [] | Parentheses () |
| **Performance** | Slower due to mutability | Faster due to immutability |
| **Use Cases** | Dynamic collections (e.g., tasks | Fixed collections (e.g., coordinates) |
| **Methods** | Many (e.g., append(), insert()) | Few (e.g., count(), index()) |
| **Memory** | Higher (due to dynamic nature) | Lower(fixed size) |
| **Functionality** | Suitable for frequent updates | Suitable for constant data |

1. Different methods of string concatenation?

Using the + Operator: result = str1 + " " + str2

Using the .join() Method: result = " ".join(words)

Using Formatted String Literals (f-strings): result = f"My name is {name} and I am {age} years old."

Using .format() Method: result = "My name is {} and I am {} years old.".format(name, age)

Using Concatenation in Loops: for sentence in sentences: result += sentence + " "

1. Explain List comprehension.

List comprehension is a concise and expressive way to create lists in Python.

It provides a compact syntax for creating lists by iterating over an iterable object (like lists, tuples, strings, etc.) and applying an expression to each element.

List comprehensions are often favored for their readability and the reduction in lines of code compared to traditional methods like using loops.

numbers = [1, 2, 3, 4, 5]

squared\_numbers = [num \*\* 2 for num in numbers]

print(squared\_numbers) # Output: [1, 4, 9, 16, 25]

1. What is set?

set is an unordered collection of unique elements.

It is defined by curly braces {} and can contain various immutable data types such as integers, floats, strings, and tuples.

Sets are primarily used to eliminate duplicate entries from a sequence and to perform mathematical set operations such as union, intersection, difference, and symmetric difference.

1. What is Shallow Copy?

shallow copy is a copy of an object that references the original elements.

It creates a new object but inserts references into it to the objects found in the original.

In other words, the copied object itself is new, but the elements it contains are references to the same objects as the original.

Using copy() Method: shallow\_copy\_list = original\_list.copy()

Using Slicing ([:]) for Lists: shallow\_copy\_list = original\_list[:]

Using copy Module for General Objects: shallow\_copy\_list = copy.copy(original\_list)

1. What is a deep copy?

deep copy is a complete copy of an object and all of its nested objects, recursively.

It means that a deep copy duplicates not just the object itself, but also all objects found within the object, and all objects found within those objects, and so on.

In essence, it creates a fully independent copy where changes to the original object will not affect the copied object, and vice versa.

Using deepcopy() Function: deep\_copy\_list = copy.deepcopy(original\_list)

1. What is mutable and immutable in Python?

**Immutable Objects:**

Immutable objects are objects whose state cannot be modified after they are created. When you modify an immutable object, a new object is created in memory. Immutable objects in Python include:

1. **Integers (int)**: e.g., x = 5
2. **Floats (float)**: e.g., y = 3.14
3. **Complex numbers (complex)**: e.g., z = 2 + 3j
4. **Strings (str)**: e.g., s = "Hello"
5. **Tuples (tuple)**: e.g., t = (1, 2, 3)
6. **Frozen sets (frozenset)**: e.g., fs = frozenset({1, 2, 3})
7. **Boolean (bool)**: e.g., b = True
8. **NoneType (None)**: e.g., n = None

**Mutable Objects:**

Mutable objects, on the other hand, are objects whose state can be modified after they are created. Modifications to mutable objects directly change their state without creating a new object. Mutable objects in Python include:

1. **Lists (list)**: e.g., my\_list = [1, 2, 3]
2. **Dictionaries (dict)**: e.g., my\_dict = {'key': 'value'}
3. **Sets (set)**: e.g., my\_set = {1, 2, 3}
4. **Byte arrays (bytearray)**: e.g., ba = bytearray(b'hello')
5. **Custom Objects (instances of classes)**: e.g., instances of a class defined by the user
6. What are iterators and generators?

|  |  |  |
| --- | --- | --- |
| **Aspect** | **Iterators** | **Generators** |
| **Definition** | Objects that implement \_\_iter\_\_() and \_\_next\_\_() methods. | Functions that use yield statements to produce a sequence of values. |
| **Protocol** | Requires manual implementation of \_\_iter\_\_() and \_\_next\_\_() methods. | Automatically implements iterator protocol (\_\_iter\_\_() and \_\_next\_\_()) through yield. |
| Syntax | Typically implemented as a class with \_\_iter\_\_() and \_\_next\_\_() methods. | |  | | --- | |  |  |  | | --- | | Defined using def keyword with yield statements. | |
| Memory Usage | Efficient for iterating over large datasets, one item at a time. | Generates values lazily, minimizing memory usage. |
| Usage | Commonly used for custom iterators over containers (lists, tuples, etc.). | Useful for generating sequences dynamically and lazily. |

1. Explain lambda function with example.

A lambda function in Python is a small anonymous function defined with the lambda keyword.

It is a way to create functions without a formal def statement. Lambda functions can have any number of arguments but only one expression.

They are typically used for short, simple operations where defining a full function using def would be overkill or less readable.

square\_lambda = lambda x: x \*\* 2 print(square\_lambda(5)) # Output: 25

1. Explain polymorphism.

Polymorphism is a fundamental concept in object-oriented programming (OOP) that allows objects of different classes to be treated as objects of a common superclass.

It enables the same method or function to behave differently based on the object it is operating on.

The word "polymorphism" is derived from Greek, where "poly" means many and "morph" means forms, indicating the ability to take many forms.

Compile-Time Polymorphism (Method Overloading)

Run-Time Polymorphism (Method Overriding)

1. What is Zip concept?

zip() function is used to combine elements from multiple iterable objects (like lists, tuples, etc.) into tuples.

It aggregates elements from each iterable into a single iterator of tuples where the first tuple contains the first element from each of the iterables, the second tuple contains the second elements, and so on.

list1 = [1, 2, 3]

list2 = ['a', 'b', 'c']

zipped = zip(list1, list2)

print(list(zipped)) # Output: [(1, 'a'), (2, 'b'), (3, 'c')]

**Statistics and Analytics:**

1.**What is P-value?**

* **P-value** is a measure in statistical hypothesis testing that helps determine the significance of results.
* It represents the probability of obtaining an effect at least as extreme as the one observed, assuming the null hypothesis is true.
* A smaller p-value (typically less than 0.05) suggests stronger evidence against the null hypothesis, leading to its rejection.

2.**Difference between z-test and t-test?**

* **Z-test:** Used when the population standard deviation is known and sample size is large (>30).

Computes a z-score to test hypotheses about the population mean.

* **T-test:** Used when the population standard deviation is unknown and sample size is small (<30).

Computes a t-statistic based on sample data to test hypotheses about the population mean. T-tests are more commonly used in practice due to smaller sample sizes and unknown population parameters.

3.**What is the F-statistic?**

* **F-statistic** is used in analysis of variance (ANOVA) to compare the variability between groups with the variability within groups.
* It assesses whether there are significant differences between the means of two or more groups.
* A high F-statistic indicates that the differences between group means are larger than expected by chance, suggesting that the groups are statistically different.

4.**What is the P-Statistic?**

* There isn't a standard statistical term "P-Statistic." It might refer to a specific statistic or parameter in a particular context, but it's not widely recognized in statistical theory.

5.**What is ANOVA?**

* **ANOVA (Analysis of Variance)** is a statistical technique used to analyze the differences among group means in a sample.
* It tests whether the means of several groups are equal or not by comparing the variation within and between groups.
* ANOVA is often used when comparing means across multiple groups or treatments in experimental designs.

6.**What are outliers, whether it should be treated in healthcare domain data?**

* **Outliers** are data points that significantly differ from other observations in a dataset. In healthcare, outliers could represent rare medical conditions, errors, or extreme values. Whether to treat outliers depends on the specific context and goals of analysis:
  + **Identification:** Outliers should be identified and understood to determine their cause and potential impact on analysis.
  + **Treatment:** Extreme outliers might be genuine data points (e.g., rare diseases) and should be carefully analyzed before deciding to treat or remove them. Treatment decisions should consider clinical significance and data integrity.

7.**If the dataset is skewed what kind of treatment will you do?**

* **Skewed data** can bias statistical analyses and model performance. Treatment options include:
  + **Logarithmic or Power Transformation:** For positively skewed data (right-skewed).
  + **Square Root or Cube Root Transformation:** For data with moderate skewness.
  + **Box-Cox Transformation:** An adaptive transformation that selects the best transformation parameter to reduce skewness.
  + **Model Selection:** Using models that are robust to skewness, such as decision trees or random forests.

8.**For categorical data which type of imputation?**

* **Most Frequent Imputation:** Replace missing categorical values with the most frequent category (mode) in the dataset.
* This approach preserves the distribution of categorical variables.

9.**What is correlation and how it ranges?**

* **Correlation** measures the strength and direction of the linear relationship between two variables. It ranges from -1 to +1:
  + **Positive Correlation:** Both variables move in the same direction.
  + **Negative Correlation:** Variables move in opposite directions.
  + **Zero Correlation:** No linear relationship between variables.
* **Strength:** Absolute value closer to 1 indicates a stronger linear relationship.
* **Direction:** Positive (+1) or negative (-1) indicates the direction of the relationship.

10.**What is the difference between Analytics & Analysis?**

* **Analytics:** Involves the use of data analysis and statistical methods to derive insights, make decisions, and predict outcomes in business or scientific contexts. It often includes techniques like machine learning, data mining, and predictive modeling.
* **Analysis:** Refers to the process of examining data or information to understand its components, structure, and relationships to derive conclusions or interpretations. It focuses on interpreting data patterns and making informed decisions based on findings.

11.**What are the Types of outlier detection methods?**

* **Statistical Methods:** Based on measures like z-scores or interquartile ranges.
* **Distance-based Methods:** Identify outliers based on distance from other points in the dataset (e.g., k-nearest neighbors).
* **Clustering-based Methods:** Identify outliers as points that do not belong to any cluster.
* **Supervised Methods:** Use machine learning models to classify outliers based on labeled data.

**SQL and Database Concepts:**

1.**What is logistic regression?**

**Logistic regression** is a statistical method used for binary classification tasks.

It predicts the probability of occurrence of an event by fitting data to a logistic curve (sigmoid function).

It's called "regression" because it models the relationship between a dependent variable and one or more independent variables.

2. **What is Linear regression?**

**Linear regression** is a statistical method used to model the relationship between a dependent variable and one or more independent variables.

It assumes a linear relationship between variables and aims to find the best-fitting line (or hyperplane in higher dimensions) through the data points.

3.**What is supervised and unsupervised learning?**

* **Supervised learning** involves training a model on labeled data (input-output pairs). The model learns to predict output values from input data. Examples include regression and classification tasks.
* **Unsupervised learning** involves training a model on unlabeled data. The model finds patterns and structures in the data without specific output labels. Examples include clustering and dimensionality reduction.

4.**What is descriptive, inferential, predictive analytics?**

* **Descriptive analytics** involves analyzing historical data to understand patterns and relationships. It answers "what happened?" based on data observations.
* **Inferential analytics** involves making inferences or predictions about a population based on a sample of data. It uses statistical techniques to generalize insights from sample data to the broader population.
* **Predictive analytics** involves using statistical models and machine learning algorithms to predict future outcomes based on historical data. It answers "what will happen?" based on patterns and trends in the data.

5.**What is Binomial theorem?**

* The **Binomial theorem** states that for any positive integer nnn, the expansion of (a+b)n(a + b)^n(a+b)n can be given as ∑k=0n(nk)an−kbk\sum\_{k=0}^{n} \binom{n}{k} a^{n-k} b^k∑k=0n​(kn​)an−kbk, where (nk)\binom{n}{k}(kn​) represents the binomial coefficient.

6.**Advantages and Disadvantages of Decision tree?**

* **Advantages:** Easy to interpret and visualize, can handle both numerical and categorical data, requires little data preparation.
* **Disadvantages:** Prone to overfitting, sensitive to small variations in data, may create biased trees if some classes dominate.

7.**Advantages and Disadvantages of KNN?**

* **Advantages:** Simple to implement, non-parametric (does not make assumptions about data distribution), effective for small datasets.
* **Disadvantages:** Computationally expensive for large datasets, sensitive to irrelevant features and noisy data, requires careful selection of distance metric.

8.**Difference between SQL and NoSQL?**

* **SQL (Structured Query Language)** databases are relational databases that use structured tables and schema to store data. They are suitable for complex queries and transactions.
* **NoSQL (Not Only SQL)** databases are non-relational databases that store data in a flexible, schema-less manner. They are suitable for handling large volumes of unstructured or semi-structured data and offer better scalability and performance for certain types of applications.

9.**Difference between modules and packages?**

* **Module:** A single file containing Python code that defines functions, classes, and variables. It can be imported and used in other Python scripts.
* **Package:** A collection of related modules bundled together. It includes an \_\_init\_\_.py file to mark the directory as a package and may contain sub-packages.

10.**Different types of Database?**

* **Relational Databases:** Store data in structured tables with predefined schemas (e.g., MySQL, PostgreSQL).
* **NoSQL Databases:** Store data in flexible formats without predefined schemas (e.g., MongoDB, Cassandra).
* **Graph Databases:** Store data in graph structures with nodes, edges, and properties (e.g., Neo4j, ArangoDB).
* **Columnar Databases:** Optimize data storage and retrieval for column-wise operations (e.g., Apache Parquet, Google BigQuery).

11.**Explain about ETL?**

* **ETL (Extract, Transform, Load)** is a process in data warehousing and data integration. It involves:
  + **Extract:** Extracting data from various sources (databases, files, APIs).
  + **Transform:** Cleaning, filtering, and transforming raw data into a suitable format for analysis.
  + **Load:** Loading transformed data into a data warehouse or target database for analysis and reporting.

12.**Explain Views and Keys in SQL?**

* **Views:** Virtual tables created by a query that fetches data from one or more tables. Views can simplify complex queries, provide data security, and hide complexity from users.
* **Keys:** In SQL, keys are used to uniquely identify records in a table.
  + **Primary Key:** Uniquely identifies each record in a table.
  + **Foreign Key:** Establishes a link between two tables by referencing the primary key of another table.

13.**What is drop out in computer vision?**

* **Dropout** is a regularization technique used in neural networks to prevent overfitting. During training, a random subset of neurons is ignored (dropped out) with a certain probability. This forces the network to learn redundant representations and improves generalization.

14.**What is the primary key?**

* In SQL databases, a **primary key** is a column (or set of columns) that uniquely identifies each row in a table. It must contain unique values and cannot have NULL values.

15.**What are the different types of SQL joins?**

* **Inner Join:** Returns records that have matching values in both tables.
* **Left Join (or Left Outer Join):** Returns all records from the left table and matching records from the right table.
* **Right Join (or Right Outer Join):** Returns all records from the right table and matching records from the left table.
* **Full Join (or Full Outer Join):** Returns all records when there is a match in either left or right table.

**Deep Learning:**

1.**Explain about RNN Model?**

* **RNN (Recurrent Neural Network)** is a type of neural network designed for sequential data processing. Unlike traditional feedforward networks, RNNs have loops that allow information to persist. This enables them to capture dependencies and patterns in sequential data such as time series, text, and speech.
* **Structure:** RNNs have a recurrent connection that allows information to be passed from one step of the network to the next. Each step (time step) in the sequence receives an input and produces an output while maintaining a hidden state that captures information from previous steps.
* **Applications:** RNNs are used in natural language processing (NLP) tasks such as language modeling, text generation, machine translation, and sentiment analysis. They are also applied in speech recognition, time series prediction, and other sequential data tasks.

2.**What is CNN?**

* **CNN (Convolutional Neural Network)** is a specialized type of neural network designed for processing grid-like data, such as images and videos.
* **Architecture:** CNNs consist of layers including convolutional layers, pooling layers, and fully connected layers. Convolutional layers apply filters (kernels) to input images to extract features. Pooling layers reduce spatial dimensions, and fully connected layers perform classification or regression tasks.
* **Feature Extraction:** CNNs are effective in feature extraction due to their ability to preserve spatial relationships and hierarchical feature learning. They automatically learn relevant features from raw pixel values, making them suitable for computer vision tasks.
* **Applications:** CNNs are widely used in image classification, object detection, facial recognition, medical image analysis, and video analysis.

3.**What is ANN and how does it work?**

* **ANN (Artificial Neural Network)** is a computational model inspired by the biological neural networks in the human brain. It consists of interconnected nodes (neurons) organized in layers: input layer, hidden layers, and output layer.
* **Working:** Input data is fed into the input layer, which passes it to the hidden layers through weighted connections. Each neuron in the hidden layers applies an activation function to the weighted sum of inputs. The output layer produces predictions or classifications based on the final hidden layer's activations.
* **Training:** ANN learns from data through a process called backpropagation, where it adjusts weights to minimize the difference between predicted and actual outputs (loss function). This optimization process uses algorithms like gradient descent to update weights iteratively.
* **Applications:** ANNs are used in a wide range of applications including regression, classification, pattern recognition, natural language processing, and recommendation systems. They are foundational in machine learning and deep learning research and applications.

**Big Data and Data Engineering:**

1.**What are the 4 V's in Big Data?**

* The 4 V's in Big Data refer to:
  + **Volume:** The vast amount of data generated from various sources, including sensors, social media, and business transactions. Big Data technologies are designed to handle large volumes of data efficiently.
  + **Velocity:** The speed at which data is generated, collected, and processed. Real-time and near-real-time data processing is essential for applications like IoT, financial trading, and social media analytics.
  + **Variety:** The diversity of data types and sources, including structured, semi-structured, and unstructured data. Big Data systems must accommodate different data formats and integrate data from various sources.
  + **Veracity:** The accuracy and trustworthiness of data. Big Data often deals with data of varying quality, completeness, and consistency. Ensuring data veracity involves data cleaning, validation, and quality assurance processes.

2.**What is right join?**

* **Right Join (or Right Outer Join)** is an SQL operation that returns all records from the right table and the matched records from the left table. If there are no matches, NULL values are returned for the left table.
* **Usage:** Right join is used when you want to include all records from the right table and only matching records from the left table based on a specified condition.

3.**What data analysis techniques are present?**

* **Data analysis techniques** encompass a variety of methods used to extract meaningful insights and patterns from data. Some common techniques include:
  + **Descriptive Analytics:** Summarizing historical data to understand past trends and patterns.
  + **Exploratory Data Analysis (EDA):** Visualizing data to discover relationships, outliers, and patterns.
  + **Inferential Analytics:** Making inferences and predictions about a population based on a sample of data using statistical methods.
  + **Predictive Analytics:** Using statistical models and machine learning algorithms to forecast future outcomes based on historical data.
  + **Diagnostic Analytics:** Analyzing data to understand the root causes of events and behaviors.
  + **Prescriptive Analytics:** Recommending actions based on analysis and predictions to optimize outcomes.

**Programming Concepts:**

1. Write simple program for Decorators in Python.

# Example of a decorator function

def my\_decorator(func):

def wrapper():

print("Something is happening before the function is called.")

func()

print("Something is happening after the function is called.")

return wrapper

# Using the decorator

@my\_decorator

def say\_hello():

print("Hello!")

# Calling the decorated function

say\_hello()

1. Write program for multiple return in function.

# Example of a function with multiple return values

def calculate(x, y):

return x + y, x - y, x \* y

# Calling the function and unpacking the returned values

sum\_result, difference\_result, product\_result = calculate(5, 3)

print("Sum:", sum\_result)

print("Difference:", difference\_result)

print("Product:", product\_result)

1. What are decorators?

**Decorators** in Python are functions that modify the behavior of other functions or methods.

They allow you to add functionality to an existing function without modifying its structure.

Decorators are often used for logging, authorization, caching, and more.

1. What is class?

A **class** in Python is a blueprint for creating objects.

It defines properties (attributes) and behaviors (methods) that all objects of the class will have.

Objects are instances of classes, and classes provide a way to structure and organize code.

1. Explain the Scope of variables?

**Scope** in Python refers to the visibility and accessibility of variables within different parts of the code. There are generally two scopes:

* **Global scope:** Variables defined outside of any function or class. They can be accessed anywhere in the code.
* **Local scope:** Variables defined inside a function or a class method. They are accessible only within that function or method.

1. Explain the Difference between *multiple* Inheritance and *multi level* Inheritance.

 **Multiple Inheritance:** In multiple inheritance, a class inherits properties and methods from more than one parent class. Python supports multiple inheritance, allowing a child class to inherit from multiple parent classes using comma-separated class names in the class declaration.

 **Multilevel Inheritance:** In multilevel inheritance, a class inherits properties and methods from another class, which itself inherits from another class. This forms a chain of inheritance where each derived class inherits from its immediate superclass.

1. What is the purpose of ***init*** in class?

The \_\_init\_\_ method (constructor) in Python classes is used to initialize objects created from the class. It is called automatically when an object is instantiated and allows you to initialize attributes of the object.

1. Explain the Difference between methods and functions.

 **Function:** A function in Python is a block of reusable code that performs a specific task. It can be defined using the def keyword and can be called anywhere in the program.

 **Method:** A method in Python is a function that belongs to an object. It is defined inside a class and operates on objects of that class. Methods have access to the object's attributes and can modify them.

1. What is the difference between deep copy and shallow copy?

 **Shallow Copy:** A shallow copy of an object creates a new object but inserts references to the original elements into it. Changes made to the shallow copy will affect the original object if the elements are mutable.

 **Deep Copy:** A deep copy of an object creates a completely new object with its own copy of the data. Changes made to the deep copy do not affect the original object, even if the elements are mutable.

1. Explain classes?

**Classes** in Python are blueprints for creating objects.

They define the properties (attributes) and behaviors (methods) that all objects of the class will have.

Classes provide a way to structure and organize code into reusable components.