Python Interview Topics

Python Fundamentals

1. What is Python?
   - Python was founded by Guido van Rossum in 1991.
   - Python is a high-level, interpreted, general-purpose programming language.
   - It can be used to build almost any type of application with the right tools/libraries.
   - Additionally, python supports objects, modules, exception-handling, and automatic memory management which help in modeling real-world problems and building applications to solve these problems.

2. What are the benefits of using Python?
   - Because, Python is a simple, easy-to-learn language because of its syntax that enhances readability.
   - Python is used in almost any type of domain like software development, web development, Game development, and especially in data science.
   - It has advanced libraries django for development, scikit-learn for ML, tensorflow for dl, etc.

3. Mention 5 fields in which Python is popularly used.
   - Robotics Programming
   - Web Development
   - Game Development
   - Software Development
   - Data-science

4. What is a dynamically typed language?
   - Before we understand a dynamically typed language, we should learn about what typing is. “Typing” refers to type-checking in programming languages. In a “strongly-typed” language, such as Python, "1"(str) + 2(int) will result in a type error since these languages don't allow for "type-coercion" (implicit conversion of data types). On the other hand, a “weakly-typed” language, such as Javascript, will simply output "12" as a result.
   - Type-checking can be done in two stages:
     “Static” - Data Types are checked before execution.
     “Dynamic” - Data Types are checked during execution.
   - Python is an interpreted language, that executes each statement line by line (means at runtime), and thus type-checking is done on the fly, during execution. Hence, Python is a Dynamically Typed Language.
   - Read more : TechTarget

5. What is interpreted language? How it is different from compiled language?
   - Firstly, we need to understand that the code is written in the English language, but computers understand the binary language. So language should be converted into binary language.
   - For this, to convert the English language into binary code there is the software called Interpreter and Compiler.
   - Some languages c++/java have a compiler, whereas Python has an interpreter.
   - The main difference between a compiler and an interpreter is that the compiler reads the code in one go and then converts the code into binary and gives the output. In Interpreted language, the interpreter reads the code line by line, executes it, and gives the output.
   - Another difference between compiled and interpreted language is that compiled type language is fast as it reads the code in one go. Thus, we can say that Python is slow as compared to C++ and Java.
   - Hence, Python is an interpreted language.

6. What is PEP 8 and why is it important?
   - PEP 8 is a document with a set of rules or guidelines for the programmers to guide how to write the code properly in a company or an open source contribution.
   - It guides how to improve the readability and consistency of the code.
   - Read more: PEP8

7. Difference between Functional programming, Procedural programming, Object-oriented programming, and Event-driven programming.
   - Functional programming - It decomposes a problem into a set of functions.
   - Procedural programming - It solves a problem by implementing one statement (procedure) at a time. Thus it contains explicit steps that are executed in a specific order. It also uses functions, but these are not mathematical functions like the ones used in functional programming. Functional programming focuses on expressions, whereas Procedural programming focuses on statements.
   - Object-oriented programming model - It mimics the real world by creating inside the computer a mini-world of objects.
   - Event-driven programming model - It generates events when we interact with different GUI elements like Windows, check boxes, buttons, combo boxes, scroll bars, menus, etc. Each event is tackled by calling an event handler function.

8. What is “print” in python?
   - It Prints the specified message to the screen. It Returns a printed representation of the objects.
   - print (*objects, sep=’ ‘, end=’n’)

9. What are the common built-in data types in Python?
    - None Type: Represents the NULL values in Python.
    - Integer (int): Stores integer literals including hex, octal, and binary numbers as integers.
    - Float: Stores literals containing decimal values.
    - Complex: Stores complex numbers in the form (A + Bj) and has attributes: real and imag
    - Bool: Stores boolean value (True or False).
    - List: Mutable sequence used to store collection of items.
    - Tuple: Immutable sequence used to store collection of items.
    - Range: Represents an immutable sequence of numbers generated during execution.
    - Str: Immutable sequence of Unicode code points to store textual data.
    - Dict: Stores comma-separated list of key: value pairs.
    - Set: Mutable unordered collection of distinct hashable objects.
    - Frozenset: Immutable collection of distinct hashable objects.

    - Note: set is mutable and thus cannot be used as a key for a dictionary. On the other hand, frozenset is immutable and thus, hashable, and can be used as a dictionary key or as an element of another set.

10. Difference b/w type() and isinstance().
    - type() checks the exact type of an object.
    - isinstance() checks if an object is an instance of a specified class or its subclass.

11. Explain the difference between mutable and immutable data types with examples.
    - Mutable: Objects whose values can be changed after they are created. Ex - Lists, dictionaries, sets.
    - Immutable: Objects whose values cannot be changed after they are created. If we need a different value, we must create a new object. Ex - Integers, floats, strings, tuples.

12. Explain the concept of sequences in Python and give examples of sequence data types.
    - A sequence is a data type that represents an ordered collection of items.
    - Sequences are iterable and support various operations like indexing, slicing, and concatenation.
    - Some common sequence data types in Python include str (string), list, tuple, and range.

13. Explain the concept of type casting in Python.
    - Type casting in Python refers to the process of converting a variable from one data type to another. This is useful when you want to perform operations or comparisons involving variables of different data types.

14. How can you check the length of a sequence in Python?
    - In Python, the built-in function len() is used to determine the length of a sequence, such as a string, list, tuple, or any other iterable.

15. What is the purpose of the None data type in Python?
    - In Python, None is a special constant representing the absence of a value or a null value. It is often used to signify that a variable or a function does not have a meaningful value or does not return anything.

16. What are variables?
    - Variables in programming are used to store data that can be used and manipulated throughout a program. They act as containers for values, allowing you to name and refer to data in your code.

17. What are Static binding and Dynamic Binding?

18. What are Identifiers and Keywords?
    - Keywords are the words that are already present in the python such as is, for, if, def, etc.
    - Identifiers are nothing but the variables. But, identifiers can not be start with digit. We can use special characters. And identifiers can be the keywords.

19. What is a loop in Python?
    - In Python, a loop is a programming construct that allows the execution of a sequence of statements or a block of code repeatedly. There are two main types of loops in Python: for loop and while loop.

20. Explain the difference between for and while loops in Python.
    - In Python, both for and while loops are used for repetitive execution of a block of code.
    - ‘for’ loops are used when the number of iterations is known, such as when iterating over elements in a list or any iterable object. The loop variable takes on each value in the sequence during each iteration and perform the specified operation on the value.
    - ‘while’ loops are used when the number of iterations is not known, and the loop continues as long as a certain condition is true. The loop variable is typically initialized before the loop, and the condition is checked before each iteration.

21. What is the purpose of the range() function in a for loop?
    - The range() function in Python is often used in for loops to generate a sequence of numbers. It helps in iterating a specific number of times or creating a sequence of numbers without explicitly specifying all the values.

22. How can you use the break statement to exit a loop prematurely in Python?
    - The break statement is used to exit a loop prematurely, regardless of the loop's normal exit condition. It is often used when a certain condition is met, and you want to stop the loop immediately.

23. Discuss the use of the continue statement in Python loops.
    - The continue statement is used in Python to skip the rest of the code inside a loop for the current iteration and move to the next iteration. It is often used when a specific condition is met, and you want to skip the remaining code in the loop for that particular iteration.

24. How do you iterate over both the index and element in a sequence using the enumerate() function?
    - In Python, the enumerate() function is used to iterate over both the index and the element in a sequence (e.g., a list, tuple, or string). It returns pairs of the form (index, element), allowing you to access both the position and value during iteration.

25. Explain the concept of a nested loop in Python.
    - In Python, a nested loop is a loop inside another loop. This allows for more complex iteration patterns, where the inner loop repeats its entire cycle for each iteration of the outer loop. Nested loops are commonly used to traverse elements in a 2D array, matrix, or for performing operations with multiple levels of repetition.

26. Discuss the use of the pass statement in a loop block in Python.
    - In Python, the pass statement is a null operation, and it acts as a placeholder where syntactically some code is required but no action needs to be taken. It is often used in situations where the syntax demands a statement, but you want to skip doing anything.
    - When used in a loop block, the pass statement allows you to create an empty loop body without causing any errors. This can be useful when you are planning to implement the loop logic later or if the loop should intentionally do nothing.

27. How can you use the zip() function in a for loop to iterate over multiple sequences?
    - In Python, the zip() function is used to combine multiple sequences into an iterator of tuples. It takes iterables (like lists, tuples, or strings) as arguments and returns an iterator that generates tuples containing elements from the input sequences.
    - When used in a for loop, zip() can be used to iterate over multiple sequences simultaneously. Each iteration of the loop produces a tuple containing elements from the corresponding positions of the input sequences.

28. What is the difference between range() and xrange()?
    - range() creates a static list that can be iterated through while checking some conditions. This is a function that returns a list with integer sequences.

- xrange() is same in functionality as range() but it does not return a list, instead it returns an object of xrange(). xrange() is used in generators for yielding.

29. What is docstring in Python?
    - Documentation string or docstring is a multiline string used to document a specific code segment. The docstring should describe what the function or method does.

30. How strings are stored in Python?
    - Stackoverflow
    - Quora
    - Medium

31. What are Lists?
    - List is a data type where you can store multiple items under 1 name. More technically, lists act like dynamic arrays which means you can add more items on the fly.

32. List Comprehension and its advantages.
    - List Comprehension provides a concise way of creating lists.
        - newlist = [expression for item in iterable if condition == True]
    - Advantages of List Comprehension
        - More time-efficient and space-efficient than loops.
        - Require fewer lines of code.
        - Transforms iterative statement into a formula.

33. Disadvantages of Python Lists.
    - Slow
    - Risky usage
    - eats up more memory

34. What is slicing in Python?
    - As the name suggests, ‘slicing’ is taking parts of.
    - Syntax for slicing is [start : stop : step]
    - start is the starting index from where to slice a list or tuple
    - stop is the ending index or where to stop.
    - step is the number of steps to jump.
    - Default value for start is 0, stop is number of items, step is 1.
    - Slicing can be done on strings, arrays, lists, and tuples

35. What is the difference between Python Arrays and lists?
    - Arrays in python can only contain elements of same data types i.e., data type of array should be homogeneous. It is a thin wrapper around C language arrays and consumes far less memory than lists.
    - Lists in python can contain elements of different data types i.e., data type of lists can be heterogeneous. It has the disadvantage of consuming large memory.

36. How List and nested lists are stored in memory?
    - From Notebook

37. How strings are stored in memory?
    - From Notebook

38. What is a tuple in Python?
    - In Python, a tuple is an ordered and immutable collection of elements. Tuples are similar to lists, but unlike lists, once a tuple is created, its elements cannot be changed or modified. Tuples are defined using parentheses ( ).

39. How do you create an empty tuple in Python?
    - In Python, an empty tuple can be created by using an empty set of parentheses ().

40. Explain the difference between a tuple and a list in Python.
    - In Python, both lists and tuples are used to store collections of elements, but they have key differences in terms of mutability and syntax.

41. Discuss the immutability property of tuples in Python.
    - the immutability property of tuples means that once a tuple is created, its elements cannot be modified, added, or removed. Tuples provide a fixed collection of elements that remain constant throughout their existence.

42. Explain the concept of tuple packing and unpacking in Python.
    - tuple packing and unpacking are concepts related to working with tuples. Tuple packing refers to creating a tuple by combining multiple values, and tuple unpacking involves extracting individual values from a tuple.
        - # tuple unpacking
        - a,b,c = (1,2,3)
        - print(a,b,c)

43. Explain the concept of tuple comprehension in Python (or why it doesn't exist?).
    - Unlike list comprehension, tuple comprehension does not exist in Python. List comprehension is a concise way to create lists, but this feature is not extended to tuples. However, you can use the tuple() constructor with a generator expression to achieve a similar result.

44. Why tuples take less memory than lists?
    - From Notebook

45. What is a set in Python?
    - a set is an unordered collection of unique elements. Sets are commonly used for membership testing, removing duplicates from other collections, and performing mathematical set operations.

46. How do you create an empty set in Python?
    - An empty set can be created using the set() constructor.

47. Explain the key characteristics of sets in Python.
    - Sets do not allow duplicate elements.
    - Sets are unordered, so the elements have no specific order.
    - Elements can be added to a set using the add() method.
    - Elements can be removed from a set using the remove() method.

48. Discuss the difference between a set and other data types, such as lists or tuples.
    - Uniqueness: Sets only allow unique elements. Duplicate values are automatically removed.
    - Mutability: Sets are mutable, meaning you can add or remove elements after the set is created.
    - Ordering: Sets are unordered, meaning they do not maintain the order of elements.
    - Indexing: Sets do not support indexing or slicing since they are unordered.

49. What is the purpose of the add() method in Python sets?
    - The add() method in Python sets is used to add a single element to a set. It ensures that the element is unique, and if the element is already present, the set remains unchanged.

50. How do you remove an element from a set in Python?
    - To remove an element from a set in Python, you can use the remove() or discard() method. Both methods take an element as an argument and remove it from the set. If the element is not present, remove() will raise a KeyError, while discard() will not.

51. Explain the concept of set operations (union, intersection, difference).
    - Set operations in Python, such as union, intersection, and difference, allow you to perform mathematical operations on sets.

52. What is the purpose of the clear() method in Python sets?
    - The clear() method in Python sets is used to remove all elements from the set, resulting in an empty set.

53. Explain the concept of a frozen set in Python.
    - A frozen set in Python is an immutable, hashable version of a set. Once created, a frozen set cannot be modified by adding or removing elements. It is similar to a set,

but it provides an immutable version suitable for situations where immutability is required.

54. Discuss the use of the update() method in Python sets.
    - The update() method is used to update a set by adding elements from another iterable or set. This method modifies the original set in place, adding new elements while eliminating duplicates.

55. How do you check if one set is a subset of another in Python?
    - you can use the issubset() method or the <= operator to check if one set is a subset of another.

56. What is the purpose of the pop() method in Python sets?
    - The pop() method is used to remove and return an arbitrary element from the set. It raises a KeyError if the set is empty.

57. Explain the difference between union() and update() methods in sets.
    - The union() and update() methods in sets are used to combine elements from multiple sets.
    - The union() method returns a new set containing all unique elements from the sets involved, without modifying the original sets. On the other hand, the update() method modifies the set it is called on by adding elements from other sets.

58. How the elements can be accessed in sets ?
    - The elements in the sets can not be accessed using indexing (like list.)

59. Editing items:
    - The set is an immutable datatype, so the items can be added but it can not be edited.
    - The item can not be deleted also.

60. Discuss the role of the symmetric_difference() method in Python sets.
    - The symmetric_difference() method in Python sets is used to find the symmetric difference between two sets. The symmetric difference of sets A and B is the set of elements that are in either A or B, but not in both.

61. How indexing works in sets ?
    - From Notebook

62. What is a dictionary in Python?
    - a dictionary is a collection of key-value pairs. Each key must be unique within a dictionary, and it maps to a specific value. Dictionaries are defined using curly braces {}.

63. How do you create an empty dictionary in Python?
    - you can create an empty dictionary using curly braces {} or by using the built-in dict() constructor.

64. Explain the key characteristics of dictionaries in Python.
    - Unordered: Dictionaries are unordered, meaning that the order of elements is not guaranteed.
    - Mutable: You can modify dictionaries by adding, updating, or removing key-value pairs.
    - Dynamic: Dictionaries can grow or shrink in size as needed.
    - Keys: Keys in a dictionary must be unique and immutable (strings, numbers, or tuples).
    - Values: Values in a dictionary can be of any data type and can be duplicated.

65. What is the difference between a dictionary and a list or tuple?
    - In Python, dictionaries, lists, and tuples are three distinct data types, each with its own characteristics.

66. Discuss the role of the get() method in Python dictionaries.
    - In Python dictionaries, the get() method is used to retrieve the value associated with a specified key. It allows you to access a key's value without raising an error if the key is not found.

67. Explain the purpose of the update() method in Python dictionaries.
    - The update() method in Python dictionaries is used to merge the key-value pairs from one dictionary into another. If the keys already exist in the target dictionary, their values will be updated; otherwise, new key-value pairs will be added.

68. How can you remove a key-value pair from a dictionary in Python?
    - In Python, you can remove a key-value pair from a dictionary using the pop() method. This method takes a key as an argument and removes the corresponding key-value pair from the dictionary.

69. Discuss the concept of dictionary comprehension in Python.
    - Dictionary comprehension is a concise way to create dictionaries in Python using a single line of code. It is similar to list comprehension but produces dictionaries.

70. What happens if you try to access a key that doesn't exist in a dictionary?
    - When attempting to access a key that doesn't exist in a dictionary, a KeyError will be raised. It's important to handle such situations to prevent program crashes.

71. Explain the difference between the keys(), values(), and items() methods in Python dictionaries.
    - The keys(), values(), and items() methods are used to retrieve information from dictionaries in Python.
    - The keys() method returns a view of the dictionary's keys, values() returns a view of the dictionary's values, and items() returns a view of the dictionary's key-value pairs as tuples.

72. What is the purpose of the pop() method in Python dictionaries?
    - The pop() method in Python dictionaries is used to remove and return an item with a specified key. If the key is not found, it raises a KeyError or returns a default value if provided.

73. Discuss the concept of nested dictionaries in Python.
    - In Python, a nested dictionary is a dictionary that contains another dictionary or dictionaries as values. This allows you to represent a hierarchical or structured data organization.

74. Explain the difference between a dictionary and a set in Python.
    - A dictionary is a collection of key-value pairs, where each key must be unique within the dictionary. It allows you to store and retrieve values based on their associated keys.
    - A set, on the other hand, is an unordered collection of unique elements. It doesn't have key-value pairs like a dictionary, and its primary purpose is to check for membership and perform set operations (union, intersection, etc.).

75. What are lists and tuples? What is the key difference between the two?
    - Lists and Tuples are both sequence data types that can store a collection of objects in Python. The objects stored in both sequences can have different data types. Lists are represented with square brackets ['sara', 6, 0.19] , while tuples are represented with parantheses ('ansh', 5, 0.97).
    - The key difference between the two is that while lists are mutable, tuples on the other hand are immutable objects. This means that lists can be modified, appended or sliced on the go but tuples remain constant and cannot be modified in any manner.

76. Why dict key cant be mutable data types ?
    - From Notebook

77. What is a Python Function?
    - In Python, a function is a block of reusable code that performs a specific task. Functions help to organize and manage code by breaking it into smaller, more manageable parts. They also allow you to avoid repetition by writing code once and reusing it multiple times.

78. Explain the difference between return and print in a function.
    - The return statement is used to send a value back to the caller of the function. It exits the function and optionally passes back an expression to the caller. When a return statement is executed, the function terminates immediately, and control is passed back to the point where the function was called.
    - The print statement is used to display a value or message on the screen. It is primarily used for debugging or providing information to the user. The print statement does not affect the flow of the program. It simply sends output to the standard output (usually the console).

79. What is the purpose of the *args and **kwargs parameters in Python functions?
    - *args is used to pass a variable number of non-keyword arguments to a function, while **kwargs is used to pass a variable number of keyword arguments.

80. What is lambda in Python? Why is it used?
    - Lambda is an anonymous function in Python, that can accept any number of arguments, but can only have a single expression. It is generally used in situations requiring an anonymous function for a short time period.

81. What is Object Oriented Programming (OOPs)?
    - Object-Oriented Programming (OOP) is a programming paradigm or approach that uses objects and classes to design and develop applications. The main goal of OOP is to create reusable, modular, and organized code.

82. Why OOPs?
    - Reusability: By using classes and inheritance, you can reuse existing code, making development faster and more efficient.
    - Modularity: Since code is organized into classes, it's easier to manage, debug, and understand.
    - Scalability: OOP makes it easier to manage and scale large projects due to its structured approach.
    - Maintainability: Changes in the code are easier to manage, as you can make updates to specific parts without affecting the whole system.

83. What is a Class?
    - A class is like a blueprint for objects. It defines the properties (attributes) and behaviors (methods) that the objects created from the class will have. For example, a class Car might have attributes like color, model, and speed, and methods like drive and stop.

84. What is an Object?
    - Objects are instances of classes. They represent real-world entities or concepts. For example, a car, a person, or a bank account can be objects.

85. What are the main features of OOPs?
    - Encapsulation: Encapsulation is the binding of data and methods that manipulate them into a single unit such that the sensitive data is hidden from the users.
    - Abstraction: Abstraction is similar to data encapsulation and is very important in OOP. It means showing only the necessary information and hiding the other irrelevant information from the user. Abstraction is implemented using classes and interfaces.
    - Polymorphism: Polymorphism allows methods to do different things based on the object it is acting upon, even though they share the same name. This can be achieved through method overriding and method overloading.
        - A) Compile-Time Polymorphism: Compile time polymorphism, also known as static polymorphism or early binding is the type of polymorphism where the binding of the call to its code is done at the compile time.
        - B) Runtime Polymorphism: Also known as dynamic polymorphism or late binding, runtime polymorphism is the type of polymorphism where the actual implementation of the function is determined during the runtime or execution.
    - Inheritance: Inheritance is a mechanism where a new class (derived class) inherits the attributes and methods of an existing class (base class). This helps in reusing code and creating a hierarchical relationship between classes.

86. What is Constructor?
    - Constructor is a builtin function (__init__) in which variable are declared and is called automatically whenever the object is created for this class, we don't need to call this function explicitly.

87. What are magic methods ?
    - Magic methods are the special kind of methods.
    - They are shown as method_name().
    - Eg. init(self) called as constructor.
    - By using magic methods, we can create our own data type.
    - This methods get automatically called whenever an object is called.
    - This methods are used to write configuration related code(connection of database, networking, etc.)
    - Or can say the functionality which can not be given to the user or the control of the program does not given to the user, it runs automatically when need occurs.

88. Method vs Function
    - In oop, When we make a function into a class we call it as a 'Method', or outside a class is simply called as a 'function'.

89. What is Self ?
    - The first golden rule of oop is that, only object of that class can access the data/attributes/methods.
    - We can not have an access to use a function inside a class with another function.

- To overcome this problem, we use the default parameter 'self'.
- We know that, everything in a python is an object. Therefore 'self' is also an object playing the role of an object created outside of a class.

90. What is Reference variable?
    - When we create an object of a class, then that object contains the memory address or the contents of the that class.
    - Basically we stored the created object in a variable and called that variable as an reference variable.
    - Hence, we can create an object without reference variable.

91. Are access specifiers used in python?
    - Python does not make use of access specifiers specifically like private, public, protected, etc. However, it does not derive this from any variables. It has the concept of imitating the behaviour of variables by making use of a single (protected) or double underscore (private) as prefixed to the variable names. By default, the variables without prefixed underscores are public.

92. How will you check if a class is a child of another class?
    - This is done by using a method called issubclass() provided by python. The method tells us if any class is a child of another class by returning true or false accordingly.

93. What is Static Variable?
    - Instance Variables are used for object whereas static variables are used for class.
    - Instance variables are those have different values and static variables has a common value.
    - obj_name.attribute -> instance ; class_name.attribute -> Static
    - static variable are declared inside the class outside all the methods.

94. What is Super keyword?
    - the super keyword is used to call methods from a parent class (also known as the superclass) in the child class (subclass). This is particularly useful in cases of inheritance, where you want to reuse code from the parent class in the child class.
    - 1) super cannot access variables/attributes 2) super cannot be used outside the class 3) super() is used in child class

95. What is Method Overloading and Overriding?
    - Method overloading occurs when multiple methods in the same class have the same name but different parameters (different type, number, or both). It allows a class to have more than one method with the same name, as long as their parameter lists are different.
    - Method overriding occurs when a subclass provides a specific implementation of a method that is already defined in its superclass. The method in the subclass has the same name, return type, and parameters as the method in the superclass.

96. What is destructor?
    - A destructor (__del__) is a special method that is called when an object is about to be destroyed or deleted. The primary purpose of a destructor is to perform cleanup activities, such as releasing resources or closing connections, before the object is removed from memory.

97. What's the meaning of single and double underscores in Python variable and method names?
    - Single Leading Underscore: `_var` are a Python naming convention that indicates a name is meant for internal use. It is generally not enforced by the Python interpreter and is only meant as a hint to the programmer.
    - Single Trailing Underscore: Sometimes the most fitting name for a variable is already taken by a keyword in the Python language. Therefore, names like class or def cannot be used as variable names in Python. In this case, you can append a single underscore to break the naming conflict.
    - Double Leading Underscore: A double underscore prefix causes the Python interpreter to rewrite the attribute name in order to avoid naming conflicts in subclasses. This is also called *name mangling*—the interpreter changes the name of the variable in a way that makes it harder to create collisions when the class is extended later.

98. What does Python's MRO (Method Resolution Order) mean?
    - Method Resolution Order is referred to as MRO. A class inherits from many classes under multiple inheritance.
    - If we attempt to access a method by building an object from the child class, the methods of the child class are first searched for the method.
    - If the method is not found in the child class, the inheritance classes are searched from left to right.
    - The `show` method is present in both the `Father` and `Mother` classes in the example presented below.
    - In MRO, methods and variables are searched from left to right because while conducting inheritance, `Father` class is written first and `Mother` class is written afterwards. So firstly `Father` class will be searched for `show` method if found then will get executed if not, `Mother` class will be searched.

99. What are the limitations of inheritance?
    - Increases the time and effort required to execute a program as it requires jumping back and forth between different classes.
    - The parent class and the child class get tightly coupled.
    - Any modifications to the program would require changes both in the parent as well as the child class.
    - Needs careful implementation else would lead to incorrect results.

100. What is Procedural Programming?

- Procedural programming is a programming paradigm that is based on the concept of procedure calls, where the logic of the program is structured into procedures, also known as routines or functions. These procedures contain a sequence of computational steps to be carried out. It is one of the most traditional and straightforward ways to write programs, emphasizing a linear top-down approach to solving problems.

101. How much memory does a class occupy?

- Classes do not consume any memory. They are just a blueprint based on which objects are created. Now when objects are created, they actually initialize the class members and methods and therefore consume memory.

102. What is a copy constructor?

- Copy Constructor is a type of constructor, whose purpose is to copy an object to another. What it means is that a copy constructor will clone an object and its values, into another object, is provided that both the objects are of the same class.