Python Questions :

1Q) . What is the difference between list, tuple, and set in Python?

Lists :

- A list is a collection of items that can be of any data type, including strings, integers, floats, and other lists.
- Lists are denoted by square brackets `[]` and elements are separated by commas.
- Lists are **mutable**, meaning they can be modified after creation.

Tuples :A tuple is a collection of items that can be of any data type, including strings, integers, floats, and other tuples.

- Tuples are denoted by parentheses `()` and elements are separated by commas.
- Tuples are **immutable**, meaning they cannot be modified after creation.

Sets :

- A set is an unordered collection of unique items.
- Sets are denoted by curly braces `{}` and elements are separated by commas.
- Sets are **mutable**, meaning they can be modified after creation.
- Sets do not maintain the order of elements.

2Q) How do you use list comprehensions in Python?

Ans: List comprehensions in Python are a concise way to create lists from other iterables . Here's a brief overview:

Basic syntax : [expression for variable in iterable].

Syntax

Conditional list comprehension :

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

even_numbers = [n for n in numbers if n % 2 == 0]

print(even_numbers)  # [2, 4]
```

3Q) What is the purpose of the _init_ method in Python classes?

Ans : The "**_init_** "method in Python classes is a special method that is automatically called when an object of the class is created. It is used to:

- Initialize the attributes of the class
- Set default values for attributes
- Perform any necessary setup or initialization.

Java Questions :

1Q) What is the difference between JDK, JRE, and JVM?

Ans : **JDK (Java Development Kit)**

- A software development kit that provides a set of tools and libraries for developing Java applications
- Includes the JRE, compiler (javac), and other development tools
- Used by developers to write, compile, and run Java programs

JRE (Java Runtime Environment)

- A package that provides the environment for running Java programs
- Includes the JVM, libraries, and utilities for running Java applications
- Used by end-users to run Java programs, but not for development

JVM (Java Virtual Machine)

- A virtual machine that runs Java bytecode on a computer

- Provides a platform-independent environment for executing Java programs
- Responsible for loading, verifying, and executing Java classes, as well as managing memory and resources

2Q) How does the for-each loop work in Java?

Ans : he for-each loop in Java is used to iterate over a collection of objects, such as an array or a **`Collection`** (e.g., **`List`**, **`Set`**). Here's how it works:

- The loop iterates over the elements of the collection, one at a time.
- For each iteration, the loop variable takes on the value of the next element in the collection.
- The loop continues until all elements in the collection have been processed.

Example :

```
String[] fruits = {"apple", "banana", "cherry"};

for (String fruit : fruits) {
   System.out.println(fruit);
}
```

3Q) What is the difference between abstract classes and interfaces in Java?

Ans :

ABSTARCT CLASS	INTERFACES
• Creating a base class for a group of related classes • Providing a partial implementation of a class • Both abstract and concrete methods • State (instance variables)	• Defining a contract or a protocol • Achieving multiple inheritance (by implementing multiple interfaces) • Only abstract methods (Java 8 and later: can have default and static methods too) • No state (instance variables)

4Q) How does exception handling work in Java? Explain try, catch, and finally?

Ans : In Java, exception handling is a mechanism to handle runtime errors. Here's how it works:

- **Try block**: Code that might throw an exception is placed in a **`try`** block.
- **Catch block**: If an exception is thrown, the **`catch`** block catches it and handles it. You can have multiple "**`catch`**" blocks to handle different types of exceptions.
- **Finally block**: The "**`finally`**" block is optional and is executed regardless of whether an exception was thrown or not. It's used to release resources, close files, etc.

5Q). What is the difference between == and equals() in case of strings in Java?

Ans :

== (IDENTITY OPERATOR)	EQUALS ( ) METHOD
• Whether both objects are the same instance • Whether both references point to the same memory location • **==** checks for reference equality, not content equality • It's not reliable for comparing string contents	• Whether both strings have the same characters in the same order • Whether the contents of both strings are equal • **`equals()`** checks for content equality, not reference equality • It's the recommended way to compare string contents in Java