ApexaiQ: 04th Feb_Day2

Python OOPs Concept:

1) What is class?
Class is a Blueprint of Object.

2) What is Object?
Object is an instance of class.

3) Python Inheritance
Inheritance allows a class (child class) to acquire properties and methods of another class (parent class). It supports hierarchical classification and promotes code reuse.

Types of Inheritance:
1. **Single Inheritance:** A child class inherits from a single parent class.
2. **Multiple Inheritance:** A child class inherits from more than one parent class.
3. **Multilevel Inheritance:** A child class inherits from a parent class, which in turn inherits from another class.
4. **Hierarchical Inheritance:** Multiple child classes inherit from a single parent class.
5. **Hybrid Inheritance:** A combination of two or more types of inheritance.

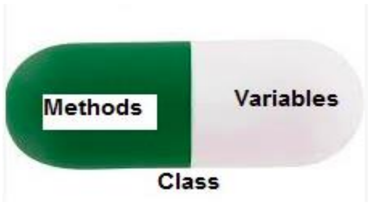
4) Python Polymorphism
Polymorphism allows methods to have the same name but behave differently based on the object's context. It can be achieved through method overriding or overloading.

Types of Polymorphism
1. **Compile-Time Polymorphism**: This type of polymorphism is determined during the compilation of the program. It allows methods or operators with the same name to behave differently based on their input parameters or usage. It is commonly referred to as method or operator overloading.
2. **Run-Time Polymorphism**: This type of polymorphism is determined during the execution of the program. It occurs when a subclass provides a specific

implementation for a method already defined in its parent class, commonly known as method overriding.

5) Python Encapsulation

Encapsulation is the bundling of data (attributes) and methods (functions) within a class, restricting access to some components to control interactions. A class is an example of encapsulation as it encapsulates all the data that is member functions, variables, etc.



Types of Encapsulation:

1. **Public Members**: Accessible from anywhere.
2. **Protected Members**: Accessible within the class and its subclasses.
3. **Private Members**: Accessible only within the class.

6) Data Abstraction

Abstraction hides the internal implementation details while exposing only the necessary functionality. It helps focus on “what to do” rather than “how to do it.”

Types of Abstraction:

- **Partial Abstraction:** Abstract class contains both abstract and concrete methods.
- **Full Abstraction:** Abstract class contains only abstract methods (like interfaces).

7) What is Exceptional Handling?

Exception handling in Python is a mechanism to handle runtime errors, ensuring that the program does not crash unexpectedly. Python provides a structured way to catch and manage errors using **try, except, else, and finally** blocks.

Some common exceptions in Python include:

- ZeroDivisionError – Division by zero.

- ValueError – Incorrect value type.
- TypeError – Operation on incompatible types.
- IndexError – Index out of range in lists or tuples.
- KeyError – Accessing a non-existent dictionary key.
- FileNotFoundError – File not found.

Keywords:

- **try** – The block where you write the code that may cause an exception.
- **except** – This block catches the exception and handles it.
- **else** – Executes if no exceptions occur.
- **finally** – Executes regardless of whether an exception occurs or not.

8) File Handling?
File handling in Python allows reading, writing, and managing files efficiently. Python provides built-in functions to work with files using the open() function.

The open() function is used to open a file and returns a file object.
# file = open("filename", "mode")

File Modes in Python**:**

Mode	Description
r	Read mode (default). Fails if the file does not exist.
w	Write mode. Creates a new file or overwrites an existing file.
a	Append mode. Adds content at the end of an existing file.
x	Exclusive creation mode. Fails if the file already exists.
t	Text mode (default).
b	Binary mode (for non-text files like images).

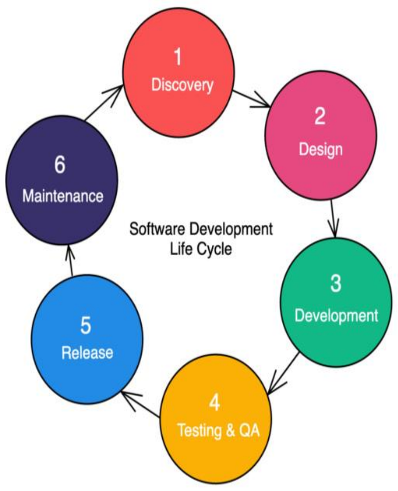
9) What is Multi-Threading?
Multithreading in Python allows multiple threads to execute concurrently, improving performance in I/O-bound tasks. The threading module provides functionality for creating and managing threads.

Threads: A thread is the smallest unit of a process that runs independently. A **multi-threaded** program runs multiple threads in parallel, sharing the same memory space.

10) What is Regex(Regular Expression)?
Regular Expressions (**regex**) in Python are patterns used to match strings. Python provides the re module to work with regex efficiently.
Basic Regex Functions:

Function	**Description**
re.match(pattern, string)	Matches pattern at the **beginning** of the string.
re.search(pattern, string)	Searches the **entire string** for a match.
re.findall(pattern, string)	Returns **all** occurrences of the pattern in a list.
re.split(pattern, string)	Splits the string by the pattern.
re.sub(pattern, replacement, string)	Replaces occurrences of the pattern.
re.finditer(pattern, string)	Returns an iterator of match objects.

11) SDLC:
The **Software Development Life Cycle (SDLC)** is a structured process for developing high-quality software efficiently. It consists of multiple phases, ensuring systematic development and maintenance.



Phases of SDLC

- **Planning**: Define objectives, identify constraints, and determine alternatives
- **Requirements Analysis**: Gather requirements from the client
- **Design**: Create a design framework, application architecture, and prototype

- **Coding**: Write the code for the software
- **Testing**: Test the software for quality and functionality
- **Deployment**: Make the software available for use
- **Maintenance**: Maintain the software over time

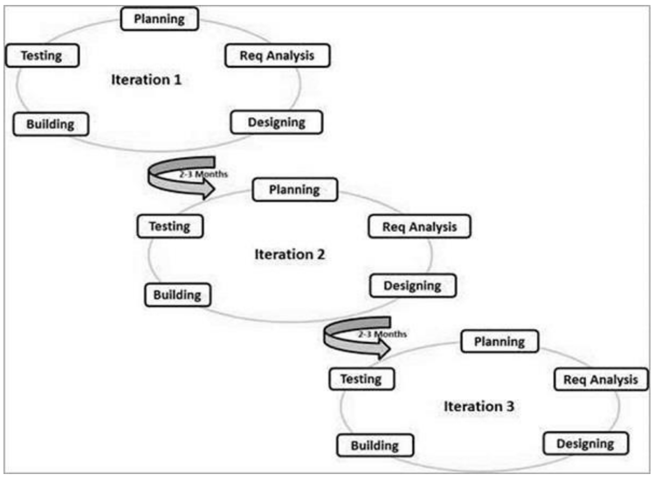
SDLC models

There are several different models for SDLC, including:

- **Waterfall**: A linear approach where each phase must be completed before the next one can begin
- **Agile**: An iterative approach that responds to changing requirements quickly
- **Spiral**: A risk-driven model that combines elements of the iterative and Waterfall models
- **Iterative**: An incremental approach that emphasizes continuous feedback
- **V-shaped**: A model that emphasizes testing and validation in a sequential process
- **Lean**: An iterative approach that focuses on increasing efficiency

12) What is Agile?

Agile model believes that every project needs to be handled differently and the existing methods need to be tailored to best suit the project requirements. In Agile, the tasks are divided to time boxes (small time frames) to deliver specific features for a release.
Iterative approach is taken and working software build is delivered after each iteration. Each build is incremental in terms of features; the final build holds all the features required by the customer.



13) What is a scrum?

Scrum **is an** Agile framework **used for** managing and developing complex projects, especially in software development**. It promotes iterative progress, teamwork, accountability, and continuous improvement.**

Scrum Framework Component:

1 Product Owner

Responsible for defining project vision and goals.

Manages and prioritizes the **Product Backlog**.

Represents customer and business needs.

2 Scrum Master

Facilitates the Scrum process and ensures team follows Agile principles.

Removes obstacles to team progress.

Acts as a coach for continuous improvement.

3 Development Team

A cross-functional team (developers, testers, designers, etc.).

Builds, tests, and delivers working product increments.

Collaborates and self-organizes tasks.

14) What is Project Management?

Project management is the process of planning, organizing, and overseeing the execution of a project to achieve specific goals within a set timeframe and budget. It involves applying knowledge, skills, tools, and techniques to project activities to meet project requirements. The primary aim of project management is to ensure that a project is completed successfully, adhering to constraints like scope, time, cost, and quality.

Key components of project management include:

1. **Project Planning**: Defining the scope, objectives, deliverables, and timelines.
2. **Resource Management**: Organizing the resources (human, financial, and material) needed for the project.
3. **Risk Management**: Identifying potential risks and developing strategies to mitigate them.

4. **Execution**: Carrying out the project plan, managing teams, and ensuring tasks are completed as per the plan.
5. **Monitoring and Controlling**: Tracking progress, adjusting plans as needed, and ensuring that the project stays within scope, time, and budget.
6. **Project Closure**: Finalizing the project, delivering the final product, and ensuring all objectives are met.