Question 1. Assignment-2-Elizaveta-240200031

In Python list can store items in order, grow dynamically, allow insertion and deletion, and support index-based access.

In C++ version List class mimics these properties manually using arrays. Dynamic resizing; when the array is full, append() doubles its capacity. Ordered storage; elements are stored and display in the order they were added. Deletion; remove_at() shifts elements left. Index-based access; we use array based index.

Question 2

In Python dictionaries store key-value pairs, they add, update & search and view the data.

In C++ version, we mimic those behaviours using two arrays: one for keys and one for values. the add() adds a new entry or updates existing one. Search() replicates get() and display() prints all the pairs. Encapsulation keeps the internal array private, preventing accidental modification from outside the class.

Question 3

Rectangle inherits from shape, so automatically has width & height and show_dimensions() method. This avoids rewriting common properties in every shape. Area() & perimeter we added to rectangle for its specific properties. Inheritance helps organize code logically & codes can be reused for different classes.

Question 4

Manager inherits from employee which inherits from person this allows manager to access name, age, emp_id & salary without redefining them. Access specifics (protected) ensures derived classes can use these members while they remain hidden from outside. This hierchey shows multi-level inheritance.

Question 5.

SportPerson inherits from both person & athlete, combining their properties (name, age, sport, medals). This allows a single object to hold all relevant information without duplicating the code. There could be clashes with same variable names but it could be resolved with :: for desired class method.

Question 6

Show_total_score can access student mark & sport_score without making them public. this is useful when operation logically involves multiple classes but shouldn't belong to either class as a method.

Question 7

The sorter class mimics python's inbuilt method sort(), it sorts array's elements using Bubble sort() to reorder them. Input() collects elements, display() prints the array, bubble_sort() implements a simple sort. OOP's design groups of the array & sorting method in one class, improving modularity and code reuse. This makes it easy to extend later with other sorting algo. without changing the main program logic.

Question 8

Function ~~Overloading~~ overloading (compile-time polymorphism) is seen in display() with different parameters (int vs string). Function overriding (run-time polymorphism) occurs when derived redefines greet() from base and we call it via derived object. Virtual function in C++ ensures the correct method is called at run-time. This shows both type of polymorphism showing how the same function name can behave differently depending on input parameters or object type.

Question 9

BankAccount uses private members to enforce encapsulation, preventing direct access from outside. The constructor initializes the account safely when creating object. Methods in the class provide controlled access to private data. Encapsulation ensures invalid operations and prevented which is crucial when it comes to banking systems.

Question 10

Shape is a abstract class with pure virtual function area so no object of shape can be created. Circle & square override area() to provide specific calculations. This allows writing generic code for shapes, while customizing each derived class. Abstract classes improve scalability, new shapes can be ~~added~~ added without modifying existing code.