**Collections – Practical Questions**

**Non-Generic Collections**

**ArrayList**

Create an ArrayList to store five random numbers. Add, retrieve, and print each element.

Add three more numbers at the end, remove the second number, and display the updated list.

Check if a specific number exists in the ArrayList and print the index if it does.

**Hashtable**

Create a Hashtable to store student IDs (as keys) and names (as values). Add five entries and print all key-value pairs.

Retrieve a student's name using their ID.

Update a name for a specific ID and display the updated Hashtable.

**Queue**

Simulate a ticket counter using Queue. Enqueue five customer names and display the queue.

Dequeue a customer as they are served and show the remaining queue each time.

**Stack**

Create a Stack to store a sequence of actions (like "login," "click," "logout").

Push three actions onto the stack and pop them one by one, displaying each action as it is removed.

**Generic Collections**

**List<T>**

Create a List<string> to store names of five students. Add, retrieve, and print each element.

Insert a name in the second position, then remove the fourth element, and display the updated list.

Search for a specific name in the list and print whether it’s found.

**Dictionary<TKey, TValue>**

Use a Dictionary<int, string> to store employee IDs as keys and their names as values. Add five entries and display all key-value pairs.

Retrieve the name of an employee using their ID.

Remove an entry using a specific key and print the updated dictionary.

**Queue<T>**

Simulate a printer queue using Queue<string>, where each item is a document name. Enqueue three documents and display the queue.

Dequeue a document as it gets printed, displaying the remaining queue after each operation.

**Stack<T>**

Use a Stack<int> to represent an undo feature. Push five numbers, where each number represents an action.

Pop and display each action as it’s undone, showing the remaining stack after each pop.