TASK - 7

DOCUMENTATION

1. **How does dataset work, Alternative for Dataset.**

A dataset is a collection of organized data points, typically stored in a structured format like tables or spreadsheets. It represents information from a specific domain and is used for various purposes like training machine learning models, conducting statistical analysis, or supporting research.

Datasets often hold:

* Features: Attributes or characteristics of the data points.
* Targets: Desired outcomes or predictions associated with the data points.

They are accessed and manipulated using specialized software or programming tools depending on the format and size.

**Alternatives to Dataset:**

While datasets are essential for various tasks, there are alternatives depending on the specific needs:

* APIs: Application Programming Interfaces offer programmatic access to real-time data streams from various sources, eliminating the need for pre-existing datasets and enabling continuous learning.
* Data Lakes: Centralized repositories of raw, unstructured data from various sources allow for flexible exploration and analysis without requiring a predefined schema like datasets.
* Data Simulations: In situations where real data is unavailable or ethically restricted, simulations can be created to generate artificial data with specific characteristics, allowing for testing and development purposes.

1. **How datasets work.**

A dataset is a collection of organized data points, typically stored in a structured format like tables or spreadsheets. It represents information from a specific domain and is used for various purposes like training machine learning models, conducting statistical analysis, or supporting research.

Datasets often hold:

* Features: Attributes or characteristics of the data points.
* Targets: Desired outcomes or predictions associated with the data points.

They are accessed and manipulated using specialized software or programming tools depending on the format and size.

1. **How does Array work with Memory**  
    Array is the data structure which stores the values of the same type. And we

need to give the size of the array at the time of initialization. So whenever we create an array of a data type, let us consider integer then, the compiler will see that we need to create array of integers of size 4 and for this it needs to ask for memory, but how much? If we check integer size **sizeof(arr[0])** we will get 4 bytes, so to store our array we need 16 bytes **4 \* 4 = 16.** Computer memory is organized into memory cells, each storing 8 bits and has an index number.

1. **How does ArrayList work with Memory** ArrayList is a data structure which is similar to array, but the size can be

dynamic and we can make modifications like add and delete elements easily. When we create an ArrayList the JVM reserves a contiguous part of memory for it. When we are adding new elements into our list, when the number of elements reaches the size of the ArrayList it reserves a new, contiguous part of memory and copies all of its elements. At the time of initialization the array list size will be 10 or we can give the size. So if the number of elements exceeds the length of size of array list, then the size gets incremented by half of its size.

1. **Array VS ArrayList** When we create an array we need to define the size of the array. Whereas

when we create an ArrayList we need not to define the size. We cannot remove the element from the array once we added into it, But in ArrayList we can add and delete elements easily.  
 Example for Array: int[] arr = new int[10];  
 Example for ArrayList: ArrayList<Integer>arrLis = new ArrayList<>();

In array, we can access elements using index(for ex: arr[0]) and we can iterate

Using for loops(e.g., for or foreach). Whereas in ArrayList we can access methods(for ex: arrList.get(index)). We can use normal for loop or enhanced for-loop for iteration of arrayList.

1. **Understanding about memory working with Array and collection objects** Arrays:

* Arrays are contiguous blocks of memory where elements are stored in consecutive locations.
* Accessing elements in an array is fast because it involves simple arithmetic calculations to determine the memory location of each element.
* Arrays in most programming languages are fixed-size, meaning their size cannot be changed once they are created.
* When an array is re-sized, a new, larger memory block may be allocated, and the elements from the old array are copied to the new one. This operation can be expensive, especially if done frequently.

Collection Objects:

* Collection objects, such as lists, sets, maps, etc., are more abstract data structures that typically manage dynamic collections of elements.
* Collection objects are often implemented using arrays or other data structures internally.
* Dynamic collections, like ArrayList in Java or List in C#, may internally use arrays to store elements. When the collection needs to grow beyond the capacity of its underlying array, a new, larger array is allocated, and the elements are copied from the old array to the new one. This process is called resizing.
* Memory management for collection objects often involves allocating and deallocating memory for nodes, arrays, or other internal structures. It also includes handling resizing, rehashing (in the case of hash tables), and dealing with memory fragmentation.

1. **How does Garbage collector work with Array/Array List/Dictionary and how is memory freed of an object and what is the process.** The garbage collector(GC) is responsible for managing memory by automatically reclaiming memory occupied by objects that are no longer needed.  
    It ensures efficient memory usage, prevents memory leaks, and allows developers to focus on writing code without explicit memory management.

Arrays:

* Arrays are contiguous blocks of memory that store elements of the same type.
* Setting an element in an array to null does not directly trigger garbage collection. The GC will collect the entire array when it determines that no references exist to any of its elements.
* In performance-critical scenarios, explicitly setting elements to null may marginally speed up garbage collection, but it’s usually unnecessary.

ArrayList (Dynamic Arrays):

* An ArrayList is a resizable array that automatically grows or shrinks as needed.
* When an element is removed from an ArrayList (e.g., using remove()), the GC will eventually collect the memory associated with that element.
* The GC identifies unused ArrayLists (those with no references) during its cycles and reclaims their memory.

Dictionaries (Maps):

* Dictionaries (or maps) associate keys with values.
* When a key-value pair is no longer reachable (e.g., the key is removed or the dictionary goes out of scope), the GC considers it for collection.
* Similar to arrays and ArrayLists, explicitly setting dictionary values to null is not necessary for garbage collection.

1. **Connection of Garbage Collector with Threading process.** Garbage Collector is responsible for reclaiming memory occupied by objects that are no longer needed. Threads are execution units within a process, and they can create and manipulate objects. Threads can also hold references to objects, preventing them from being garbage collected. When a thread is running, it is considered a garbage collection root, meaning it keeps object. The GC determines an object’s reachability based on these roots.

* Stop-the-world(STW): Some GC algorithms halt all the threads during collection. This ensures consistency but can impact application responsiveness.
* Concurrent GC: Other algorithms allow normal threads to run while GC proceeds concurrently. These collectors minimize STW(Stop the world) pauses.
* Parallel Collectors: These use multiple threads for GC tasks.
* Incremental Collectors: These break GC work into smaller chunks.

1. **How garbage collector works with multiple threads** In a multi-threaded environment, garbage collection (GC) can become more complex because multiple threads may be accessing and modifying memory concurrently. To handle this complexity, modern garbage collectors employ various strategies and techniques. They are  
    Thread Synchronization: Garbage collectors usually need to pause the execution of application threads to perform garbage collection. This pause, know as a “stop the world” event, ensures that garbage collector can safely examine and modify the memory without interference from other threads. During this pause, all application threads are stopped, and only the garbage collector threads run.

Concurrent Marketing: Some garbage collectors use concurrent marking techniques to identify which objects are reachable and which are garbage. Concurrent marking allows the garbage collector to run concurrently with the application threads, reducing the length and impact of the stop the world pauses. During concurrent marking, the garbage collector traverses the object graph, starting from the roots (e.g., global variables, thread stacks) and marking all reachable objects

1. **What do you think why Time and space complexity is important explain it  with an own example**   
    Time complexity means the time taken to a program to finish a task. Time complexity is important because it helps to complete a task in short amount of time. In early days of using computers with limited resources the large tasks take more time to execute, so due to lack of resources it takes more time to complete a task. So if we use a program which can complete a task in short time, then we can complete all the tasks in short time and save resources.  
    Let us take an example of searching a element in a array, so normally if we iterate over a array we can find the element. If the element is at the end of the array we need to iterate over complete array, so if the array size is so big then it would take more time to complete the searching tasks, which is not helpful. In this case the time complexity of linear search of element is so high.   
    We also have another approach for searching a element in a array, it is called as binary search. In this method we generally check only half of the array at a time. So if the desired element is in the first half the it only searches in the first half, if not it searches the other half. In this method it only takes half of the time of linear search for every half which takes less time than the linear search. So in this case the time complexity is low and better than the linear search

Space complexity means the memory used by a program. The more we use the data structures, the more the space complexity will be, and more resources will be taken by the system. Which may raise many problems, like we cannot write many programs and so on. So space complexity is also important with time complexity