2marks answers::

Question 1: Define time complexity.

Answer: Time complexity refers to the amount of time it takes an algorithm to execute, typically expressed in terms of the size of the input data. It's a measure of how the execution time grows as the input size increases.

Question 2: Define space complexity.

Answer: Space complexity refers to the amount of memory space an algorithm requires to run, also expressed in terms of the size of the input data. It considers the extra space used by the algorithm beyond the input data itself.

Question 3: Define Data Structures.

Answer: Data structures are specialized formats for organizing, processing, retrieving, and storing data efficiently. They provide efficient ways to access and manipulate data.

Question 4: Write different types of data structures.

Answer: There are various data structures, each with its strengths and weaknesses. Some common types include:

Arrays: Ordered collections of items of the same data type, accessed using an index.

Linked Lists: Linear collections of elements where each element (node) contains data and a reference (link) to the next element.

Stacks: LIFO (Last-In-First-Out) data structures where elements are added and removed from the top.

Queues: FIFO (First-In-First-Out) data structures where elements are added at the back and removed from the front.

Trees: Hierarchical data structures with a root element and child nodes.

Hash Tables: Data structures that use a hash function to map keys to values for efficient retrieval.

Question 5: List Different Types of liner data structures.

Answer: The prompt likely refers to "linear data structures," which are data structures where elements are arranged in a linear sequence. Common examples include:

Arrays (as mentioned earlier)

Linked Lists (as mentioned earlier)

Stacks (as mentioned earlier)

Queues (as mentioned earlier)

Question 6: Define Abstract Data type (ADT).

Answer: An Abstract Data Type (ADT) is a theoretical concept that specifies the operations (behaviors) that can be performed on a data structure without going into the details of its implementation. It focuses on what the data structure can do rather than how it's implemented.

Question 7: What is the importance of linear data structures?

Answer: Linear data structures are fundamental because:

They are simple to understand and implement.

They are efficient for basic operations like insertion, deletion, and searching in certain orders.

They serve as building blocks for more complex data structures.

Question 8: List out Abstract Data types.

Answer: While ADTs can be applied to various data structures, the concept itself isn't a list of specific data structures. However, common examples of data structures that can be viewed as ADTs include:

Lists (abstract concept of a linear sequence)

Stacks (abstract concept of LIFO behavior)

Queues (abstract concept of FIFO behavior)

Sets (abstract concept of a collection of unique elements)

Maps (abstract concept of a key-value paired collection)

Question 9: List Different Types of non-linear data structures.

Answer: Non-linear data structures don't have a strict linear order among their elements. Examples include:

Trees (as mentioned earlier)

Graphs: Collections of nodes (vertices) connected by edges.

Hash Tables (as mentioned earlier)

Question 10: What is an array? What is the syntax to declare an array?

Answer: An array is a collection of items of the same data type stored at contiguous memory locations. Elements are accessed using an index that starts from 0.

Syntax (C language example):

C

```
data_type array_name[size];
```

Use code with caution.

Example:

C

```
int numbers[10];    // An array to store 10 integers
```

Use code with caution.

Question 11: List any four operations performed on arrays.

Answer: Common array operations include:

Traversal: Accessing each element in the array, often using a loop.

Insertion: Adding a new element at a specific position (may involve shifting elements).

Deletion: Removing an element from a specific position (may involve shifting elements).

Searching: Finding the index of a specific element within the array.

Question 12: List any four types of asymptotic notations.

Answer: Asymptotic notations describe the growth rate of a function as the input size tends to infinity. Common notations include:

Big O notation (O): Represents the upper bound of an algorithm

13, Advantages of ADT

Encapsulation: ADT allows for the hiding of the internal state and requires all interaction to be performed through an interface, reducing complexity.

Abstraction: ADT provides a means to work with data at a conceptual level without needing to know the details of implementation.

14, Linear vs Nonlinear Data Structures

Linear: Data elements are arranged in a sequential manner, e.g., arrays, lists.

Nonlinear: Data elements are not arranged sequentially but rather hierarchically, e.g., trees, graphs.

15, Disadvantages of ADT

Complexity: The level of abstraction can sometimes make the system more complex.

Performance Overhead: The additional layer of abstraction can lead to performance issues in some cases.

16, Applications of Linear Data Structures

Array Implementation: For storing and accessing elements in a fixed order.

Stack Operations: Used in algorithm implementations like depth-first search.

Queue Management: In scenarios like process scheduling in operating systems.

Database Management: For storing data in a structured format that can be easily accessed and manipulated

NOTE: These are only for reference you can change these according to your lecturer notes.