

Introduction to Data Structure

Data structure is a representation of logical relationship existing between individual elements of data. In other words, a data structure defines a way of organizing all data items that considers not only the elements stored but also their relationship to each other. The term data structure is used to describe the way data is stored.

To develop a program of an algorithm we should select an appropriate data structure for that algorithm. Therefore, data structure is represented as:

$$\text{Algorithm} + \text{Data structure} = \text{Program}$$

A data structure is said to be **linear** if its elements form a sequence or a linear list. The linear data structures like an array, stacks, queues and linked lists organize data in linear order. A data structure is said to be **non linear** if its elements form a hierarchical classification where, data items appear at various levels.

Trees and **Graphs** are widely used **non-linear data structures**. Tree and graph structures represents hierarchial relationship between individual data elements. Graphs are nothing but trees with certain restrictions removed.

Importance of Data Structure

Due to the complexity of applications and the daily growth in data, there may be issues with processing **speed, data searching, handling numerous requests**, etc. Data structure offers a method for effectively **managing, organising, and storing data**. Data structures make it simple to navigate through the data elements. Data structures offer **productivity, reuse, and abstraction**. Because storing and retrieving user data as quickly as feasible is a program's primary job, it plays a significant role in improving performance.

Types of Data Structure

Data structures are divided into two types:

- **Primitive data structures.**
- **Non-primitive data structures.**

→ Primitive Data Structure :

Primitive Data Structures are the basic data structures that directly operate upon the machine instructions. They have different representations on different computers. Integers, floating point numbers, character constants, string constants and pointers come under this category.

→ Non-Primitive Data Structure :

Non-primitive data structures are more complicated data structures and are derived from primitive data structures. They emphasize on grouping same or different data items with relationship between each data item. Arrays, lists and files come under this category. Figure shows the classification of data structures.

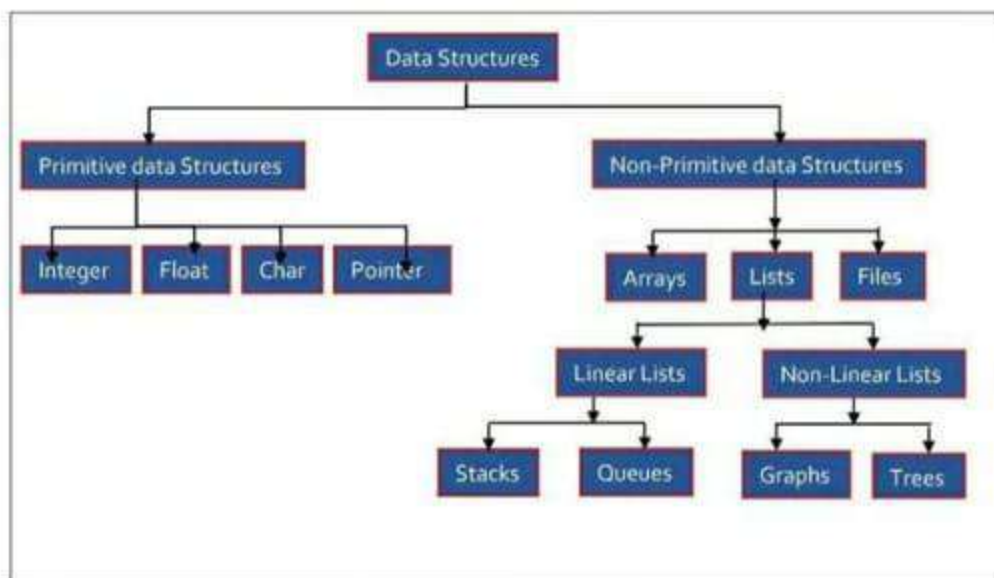


Figure : Classification of Data Structures.

The non-primitive data structure is divided into two types:

- **Linear data structure**
- **Non-linear data structure**

→ Linear Data Structure

In **linear data structures**, the elements are arranged in sequence one after the other.

Elements are arranged in particular order, they are easy to implement.

Popular linear data structures are:

1. Array Data Structure

In an array, elements in memory are arranged in continuous memory. All the elements of an array are of the same type. And, the type of elements that can be stored in the form of arrays is determined by the programming language.



An array with each element represented by an index

2. Stack Data Structure

In stack data structure, elements are stored in the LIFO principle. That is, the last element stored in a stack will be removed first.

It works just like a pile of plates where the last plate kept on the pile will be removed first.



In a stack, operations can be performed only from one end (top here).

3. Queue Data Structure

Unlike stack, the queue data structure works in the FIFO principle where the first element stored in the queue will be removed first.

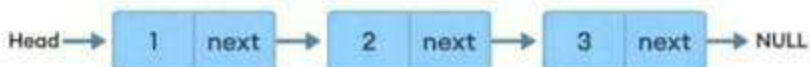
It works just like a queue of people in the ticket counter where the first person on the queue will get the ticket first.



In a queue, addition and removal are performed from separate ends.

4. Linked List Data Structure

In linked list data structure, data elements are connected through a series of nodes. And, each node contains the data items and address to the next node.



A linked list

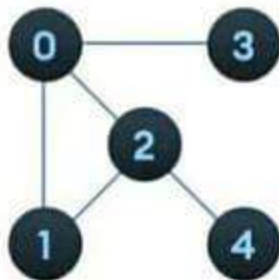
→ Non-Linear Data Structure

Unlike linear data structures, elements in non-linear data structures are not in any sequence. Instead they are arranged in a hierarchical manner where one element will be connected to one or more elements.

Non-linear data structures are further divided into graph and tree based data structures.

1. Graph Data Structure

In graph data structure, each node is called vertex and each vertex is connected to other vertices through edges.



Graph data structure example

<ul style="list-style-type: none"> The memory utilization is not efficient. 	Different structures utilize memory in different efficient ways depending on the need.
<ul style="list-style-type: none"> The time complexity increase with the data size. 	Time complexity remains the same.
<ul style="list-style-type: none"> Example: Arrays, Stack, Queue 	Example: Tree, Graph, Map

Data structures: Organization of data

The collection of data you work with in a program have some kind of structure or organization. No matter how complex your data structures are they can be broken down into two fundamental types:

- Contiguous
- Non-Contiguous

→Contiguous Structures

In **contiguous structures**, terms of data are kept together in memory (either RAM or in a file). An **array** is an example of a contiguous structure. Since each element in the array is located next to one or two other elements.

→Non-Contiguous Structures

Items in a **non-contiguous structure** are scattered in memory, but are linked to each other in some way. A **linked list** is an example of a non-contiguous data structure. Here, the nodes of the list are linked together using pointers stored in each node.

Figure below illustrates the difference between contiguous and non-contiguous structures.

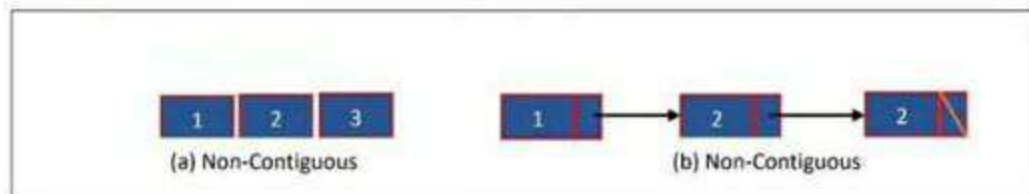


Figure : Contiguous and Non-contiguous structures compared

→ Hybrid structures

If two basic types of structures are mixed then it is a hybrid form. Then one part contiguous and another part non-contiguous.

For example, figure shows how to implement a double-linked list using three parallel arrays, possibly stored apart from each other in memory.

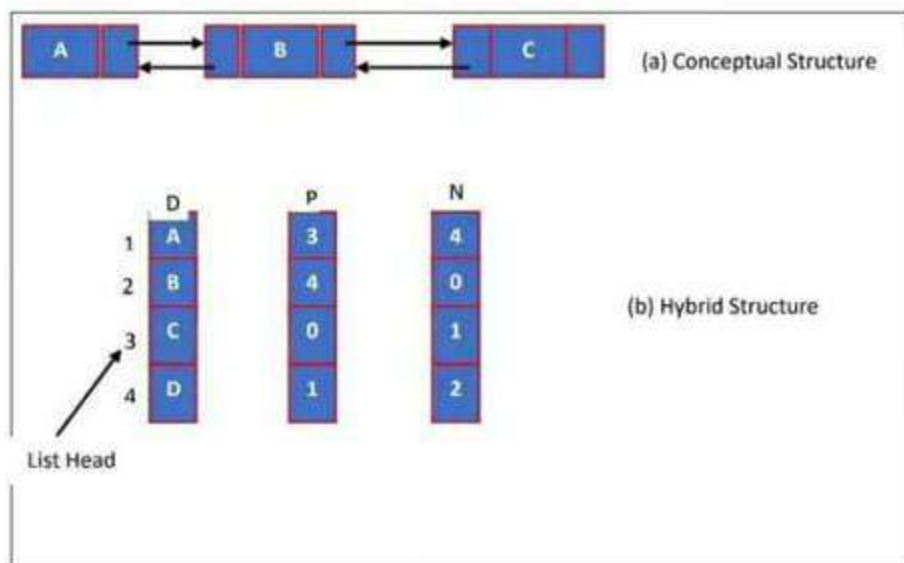


Figure : A double linked list via a hybrid data structure

The array D contains the data for the list, whereas the array P and N hold the previous and next "pointers". The pointers are actually nothing more than indexes into the D array. For instance, D[i] holds the data for node i and p[i] holds the index to the node previous to i, where may or may not reside at position i-1. Like wise, N[i] holds the index to the next node in the list.

Operation of Data Structure

Data structure operations are the methods used to manipulate the data in a data structure. The most common data structure operations are:

- **Traversal** : Traversal operations are used to visit each node in a data structure in a specific order. This technique is typically employed for **printing, searching, displaying, and reading** the data stored in a data structure.
- **Insertion** : Insertion operations add **new data elements** to a data structure. You can do this at the data structure's beginning, middle, or end.
- **Deletion** : Deletion operations remove data elements from a data structure. These operations are typically performed on nodes that are **no longer needed**.
- **Search** : Search operations are used to find a **specific data element** in a data structure. These operations typically employ a **compare** function to determine if two data elements are equal.
- **Sort** : Sort operations are used to **arrange** the data elements in a data structure in a specific order. This can be done using various sorting algorithms, such as insertion sort, bubble sort, merge sort, and quick sort.
- **Merge** : Merge operations are used to **combine** two data structures into one. This operation is typically used when two data structures need to be combined into a single structure.

Advantages of Data structures

- Data structure facilitates effective data storage in storage devices.
- The use of data structures makes it easier to retrieve data from a storage device.
- The data structure allows for the effective and efficient processing of both little and big amounts of data.
- Manipulation of vast amounts of data is simple when a proper data structure technique is used.
- The use of a good data structure may assist a programmer to save a lot of time or processing time while performing tasks such as data storage, retrieval, or processing.
- Most well-organized data structures, including stacks, arrays, graphs, queues, trees, and linked lists, have well-built and pre-planned approaches for operations such as storage, addition, retrieval, modification, and deletion. The programmer may totally rely on these facts while utilising them.
- Data structures such as arrays, trees, linked lists, stacks, graphs, and so on are thoroughly verified and proved, so anybody may use them directly without the need for study and development. If you opt to design your own data structure, you may need to do some study, but it will almost certainly be to answer a problem that is more sophisticated than what these can supply.
- In the long term, data structure utilization might merely encourage reusability.

Python Most Asked Interview Qns & Ans

1) What is Python? What are the benefits of using Python?

Python is a programming language with objects, modules, threads, exceptions, and automatic memory management. The benefits of python are that it is simple and easy, portable, extensible, build-in data structure, and it is open-source.

2) Explain namespace in Python

In Python, every name introduced has a place where it lives and can be hooked for. This is known as a namespace. It is like a box where a variable name is mapped to the object placed. Whenever the variable is searched out, this box will be searched to get the corresponding object.

3) What is lambda in Python?

It is a single expression anonymous function often used as inline function.

4) Why lambda forms in python do not have statements?

A lambda form in python does not have statements as it is used to make new function object and then return them at runtime.

5) Explain pass in Python

Pass means no-operation Python statement, or in other words, it is a place holder in a compound statement, where there should be a blank left, and nothing has to be written there.

6) In Python what are iterators?

In Python, iterators are used to iterate a group of elements, containers like a list.

7) What is the unittest in Python?

A unit testing framework in Python is known as unittest. It supports sharing of setups, automation testing, shutdown code for tests, aggregation of tests into collections, etc.

8) Explain slicing in Python?

A mechanism to select a range of items from sequence types like list, tuple, strings etc., is known as slicing.

9) What are generators in Python?

The way of implementing iterators are known as generators. It is a normal function except that it yields expression in the function.

10) What is docstring in Python?

A Python documentation string is known as docstring, it is a way of documenting Python functions, modules, and classes.

11) What are built-in type does python provides?

Python provides two built-in types: 1) Mutable and 2) Immutable.

Mutable built-in types are:

- List
- Sets
- Dictionaries
- Immutable built-in types
- Strings
- Tuples
- Numbers

Immutable built-in types are:

- Strings
- Tuples
- Numbers

12) What is PEP 8?

PEP 8 is a coding convention, a set of recommendation, about how to write your Python code more readable.

13) What is pickling and unpickling?

Pickle module accepts any Python object and converts it into a string representation and dumps it into a file by using dump function. This process is called pickling. While the process of retrieving original Python objects from the stored string representation is called unpickling.

14) How is Python interpreted?

Python language is an interpreted language. Python program runs directly from the source code. It converts the source code that is written by the programmer into an intermediate language, which is again translated into machine language that has to be executed.

15) How is memory managed in Python?

Python memory is managed by Python private heap space. All Python objects and data structures are located in a private heap. The programmer does not have an access to this private heap, and the interpreter takes care of this Python private heap.

The allocation of Python heap space for Python objects is done by the Python memory manager. The core API gives access to some tools for the programmer to code.

Python also has an inbuilt garbage collector, which recycles all the unused memory and frees the memory and makes it available to the heap space.

16) What are the tools that help to find bugs or perform the static analysis?

PyChecker is a static analysis tool that detects the bugs in Python source code and warns about the style and complexity of the bug. Pylint is another tool that verifies whether the module meets the coding standard.

17) What are Python decorators?

A Python decorator is a specific change that we make in Python syntax to alter functions easily.

18) What is the difference between list and tuple?

The difference between list and tuple is that list is mutable while tuple is not. Tuple can be hashed, for example., as a key for dictionaries.

19) How are arguments passed by value or by reference?

Everything in Python is an object, and all variables hold references to the objects. The reference values are according to the functions. Therefore, you cannot change the value of the references. However, you can change the objects if it is mutable.

20) What is Dict and List comprehensions are?

They are syntax constructions to ease the creation of a Dictionary or List based on existing iterable.

21) How can you copy an object in Python?

To copy an object in Python, you can try a `copy.copy()` or `copy.deepcopy()` for the general case. You cannot copy all objects but most of them.

22) What is negative index in Python?

Python sequences can be index in positive and negative numbers. For positive index, 0 is the first index, 1 is the second index, and so forth. For the negative index, (-1) is the last index, and (-2) is the second last index, and so forth.

23) How can you convert a number to a string?

In order to convert a number into a string, use the inbuilt function `str()`. If you want a octal or hexadecimal representation, use the inbuilt function `oct()` or `hex()`.

24) What is the difference between xrange and range?

Xrange returns the xrange object while range returns the list and uses the same memory and no matter what the range size is.

25) What is module and package in Python?

In Python, module is the way to structure a program. Each Python program file is a module, which imports other modules like objects and attributes.

The folder of Python program is a package of modules. A package can have modules or subfolders.