

Batch Name : PreCAT OM19
Subject : Data Structures

DS DAY-01:

- to store marks of 100 students
int m1, m2, m3, m4,, m100;//400 bytes

i want to sort marks in a descending

int marks[100];//400 bytes
operations can be performed on data ele's efficiently.

Array: it is a collection/list of **logically related similar type of elements** in which data ele's gets stored into the memory at contiguous location.

We want to store rollno, name & marks of student/record

Structure: it is a collection/list of **logically related similar and dissimilar type of data elements** in which data ele's gets stored into the memory collectively as a single entity(record).

Class: it is a basic/linear data structure which is a collection of logically related similar and dissimilar type of data elements as well as functions.

```
typedef struct employee emp_t;

struct employee e1;//abstraction - abstract data type
emp_t e2;

class student
{
    //data members
private:
    int rollno;
    String name;
    float marks;
public:
    //member functions
    student();
    ~student();
    //getter functions
    //setter functions
    //facilitators
};
```

student s1;//abstract data type - abstraction

student s2;- reusability

struct employee e1;

to scan an array/traversal on an array: to visit each array element sequentially from first element till max last element.

- to learn data structure is not to learn any programming language, it is nothing but to learn an **algorithms**.

Q. What is a Program?

Program is a finite set of instructions written in any programming language (like C, C++, Java, Python, Assembly etc....), given to the machine to do specific task.

Program --> Machine

Q. What is an algorithm?

An Algorithm is a finite set of instructions written in human understandable language like english, if followed, accomplishes given task.

Algorithm --> Programmer User

- A Program is an implementation of an algorithm.
- An algorithm is a like blueprint of a program.

Q. What is a pseudocode?

An Algorithm is a finite set of instructions written in human understandable language like english **with some programming constraints**, if followed, accomplishes given task, such algorithm is referred as a pseudocode.

Algorithm to do sum of array elements: --> End User/Programmer User

Step-1: initially take value of sum variable as 0.

Step-2: start traversal of an array and keep adding each element into the sum variable one by one.

Step-3: return the final value of sum.

Pseudocode: its a special form of an algorithm --> Programmer User

```
Algorithm ArraySum( A, size ){
    sum = 0;
    for( index = 1; index <= size ; index++ ){
        sum += A[ index ];
    }

    return sum;
}
```

Program: --> Machine

```
int array_sum( int arr[], int size){
    int sum = 0;
    int index = 0;
    for( index = 0 ; index < size ; index++ ){
        sum += arr[ index ];
    }
    return sum;
}
```

Code <==> Program

Source Code - program written in any programming language.

- An algorithm is a solution of a given problem.

- **algorithm = solution**

- we can have many solutions for a the same problem, in this case one need to select an efficient solution/algo.

e.g.

searching: to search a given key element in a collection/list of data elements.

1. linear search
2. binary search

sorting: to arrange data elements in a collection/list of data elements either in an ascending order or in a descending order.

1. selection sort
2. bubble sort
3. insertion sort
4. quick sort
5. merge sort

Pune --> Mumbai

- multiple paths/routes may exist between two cities
 - when we know multiple paths between 2 cities --> an optimized/efficient one
- parameters/measures:** distance, cost, status, traffic situation, time

searching:

+ Analysis of an algorithm:

- to decide efficiency of an algo's, we need to do their analysis.
- analysis of an algo, is nothing but to calculate how much **time** i.e. computer time and **space** i.e. memory it needs to run to completion.

- there are 2 measures of analysis of algo:

1. time complexity of an algo is the amount of time i.e. computer time it needs to run to completion.

2. space complexity of an algo is the amount of space i.e. computer memory it needs to run to completion.

- space ==> memory required to store variables, constants & instructions in a program/algo.

- Linear search:

step-1: accept value of key element (which is to search) from user

step-2: start traversal an array from first element and compare value of key with each array ele sequentially till match is found or max till the last element. If match found then return true, otherwise return false.

Best case: If key is found at first position --> $O(1)$ - constant time complexity

if size of an array = 10 --> no. of comparisons = 1

if size of an array = 20 --> no. of comparisons = 1

if size of an array = 50 --> no. of comparisons = 1

if size of an array = n --> no. of comparisons = 1

Worst case: If either key is found at last position or key is not exists --> $O(n)$

if size of an array = 10 --> no. of comparisons = 10

if size of an array = 20 --> no. of comparisons = 20

if size of an array = 50 --> no. of comparisons = 50

if size of an array = n --> no. of comparisons = n

- We need to follow certain rules and we have to use some notations:

Asymptotic Notations:

1. Big Omega (Ω) - this notation is used
2. Big Oh (O)
3. Big Theta (θ)

- discrete mathematics

Rule:

1. if running time of an algo is having any additive/subtractive/multiplicative/divisive constant, then it can be neglected.

e.g.

$$O(n+3) \Rightarrow O(n)$$

$$O(n-4) \Rightarrow O(n)$$

$$O(n/3) \Rightarrow O(n)$$

$$O(n/2) \Rightarrow O(n)$$

$$O(n*2) \Rightarrow O(n)$$

- if it is not mentioned specifically then (by default) we have to consider an average case time complexity for all algorithms.

- usually magnitudes of time complexities of an algo's are same in average case and worst case.

Q. What is the time complexity of a linear search?

Average Case = $\theta(n)$

Q. What is the best case time complexity of a linear search?

Best Case = $\Omega(1)$

Q. What is the average case time complexity of a linear search?

Average Case = $\theta(n)$

space =

Prerequisite C Programming Language Topics: do revise following topics

1. storage classes
2. pointers
3. functions
4. structure