Data Structure & Algorithms

Sunbeam Infotech

Agenda

- Introduction
- Data Structure
- Time complexity
- Space complexity
- Linear Search
- Binary Search
- Basic Sorting Algorithms

Trainer Introduction

- Name: Mr. Nilesh Ghule.
- Qualification: M.Sc. Electronics
- Experience: 16+ years training along with consulting & POC for Sunbeam.
- Designation: Technical Director of Sunbeam Infotech.

- Skills/Technologies/Platform:
  - C, C++, Java, JS, Scala, Python, C#.
  - Java EE (Spring, Hibernate, Spring Boot).
  - Operating Systems, Linux, Device Drivers, RTOS
  - MySQL, MongoDb, Redis, HBase, Neo4J.
  - Big Data, Hadoop & Eco Systems, Spark, Kafka
  - ARM7, ARM-CM3, AVR, 8051 & IoT
  - Win32 SDK, MFC, COM, .NET
  - Symbian, J2ME, Windows CE, Android.
- Contact:
  - nilesh@sunbeaminfo.com
  - gitlab: nilesh-g
  - linkedin.

Course Introduction

gitlab.com/nilesh-s/dsao-01

- Data Structure and Algorithms
  - Data Structures: Linked lists, Stack, Queue, Trees, Graphs, Hash Table.
  - Algorithms: Sorting, Searching, Graphs & Other DS algorithms
- Course Goals
  - Learn DS & Algo from scratch.
  - Implement each DS & Algo.
  - Understand complexity of algos.
- Course Schedules
  - Mon-Fri: 6:30 PM to 9:30 PM
  - Sat: 6:30 PM to 7:30 PM

10 min break

- Course Format
  - 6:30 PM to 9:30 PM – Online lecture + lab
  - Participants are encouraged to code alongside (at least copy the code from gitlab).
  - Post your queries in chat box.
  - Practice assignments will be shared. They are optional. If any doubts, share on WA group (possibly with screenshot). Faculty members or peers can help.
- Programming language
  - DS & Algos are language independent.
  - Classroom coding will be in C++ (use IDE of your choice).
  - Will share Java codes at the end of session.

Data Structure

- Data Structure
  - Organizing data in memory
  - Processing the data
- Common data structures
  - Array
  - Linked List
  - Stack
  - Queue
  - Hash Tables
- Advanced data structures
  - Tree
  - Heap
  - Graph

- Asymptotic analysis
  - It is not exact analysis
  - Big' O notation
- Space complexity
  - Unit space to store the data (Input space) and additional space to process the data (Auxiliary space).
  - O(1), O(n), O(n^2)
- Time complexity
  - Unit time required to complete any algorithm.
  - Approximate measure of time required to complete any algorithm.
  - Depends on loops in the algorithm.
  - O(n^3), O(n^2), O(n log n), O(n), O(log n), O(1)

Time complexity

- Write a program to calculate factorial of given number. → O(n)
- Find prime numbers between 1 to n. → O(n²)
- Print given number into binary. → O(log n)
- Print table of given number. → O(1)

```
for (i=1; i<=10; i++)
    pf ("%d", num *i);
```

T = k

↳ O(1)

O(n³)

O(n²)

O(n log n)

O(n)

O(log n)

O(1)

```
res = 1
for(i=1; i<=n; i++)
    res = res * i;
```

itr = n

$T \propto n$

$O(n)$

---

n = 100 → 98

```
for(num=1; num<=n; num++)
{
    for(i=2; i<num; i++){
        if(num % i == 0)
            break;
    }
    if(i == num)
        pf("%d", num);
}
```

itr = 0 + 0 + 1 + 2 + 3 + ... + (n-2)

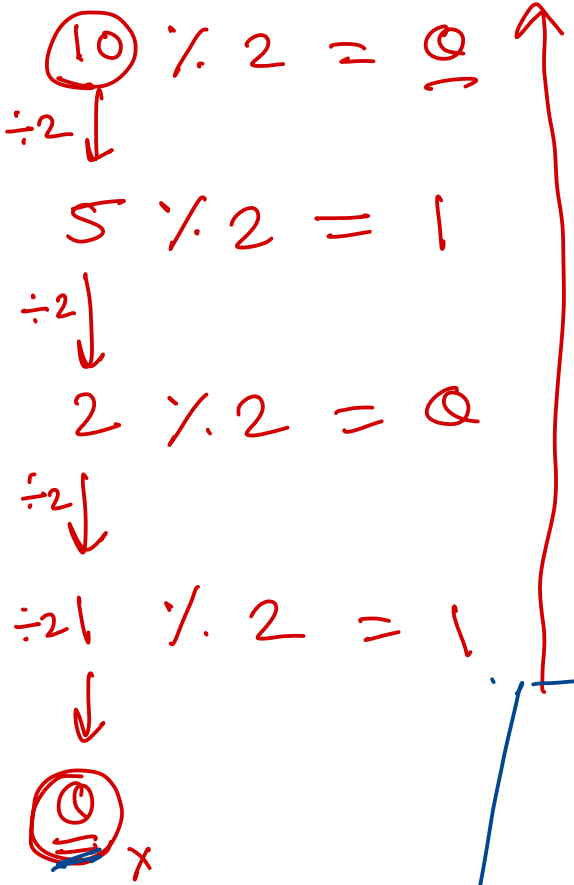
$T \propto \frac{(n-2)(n-1)}{2}$

$T \propto n^2 - 3n + 2$

$n \gg 1$

$n^2 \ggg n$ | theory of approximation,

$T \propto n^2$ → $O(n^2)$



10 10 10

```
while(n > 0) {
    pf("%d", n%2);
    n = n / 2;
}
```

$2^{itr} = n$

$\log 2^{itr} = \log n$

$itr \log 2 = \log n$

$itr = \frac{\log n}{\log 2}$

$T \propto \frac{\log n}{\log 2}$

$O(\log n)$

$1024 \rightarrow 10$ itr

$2^{itr} = n$

$\log 2^{itr} = \log n$

$itr \log 2 = \log n$

$itr = \frac{\log n}{\log 2}$

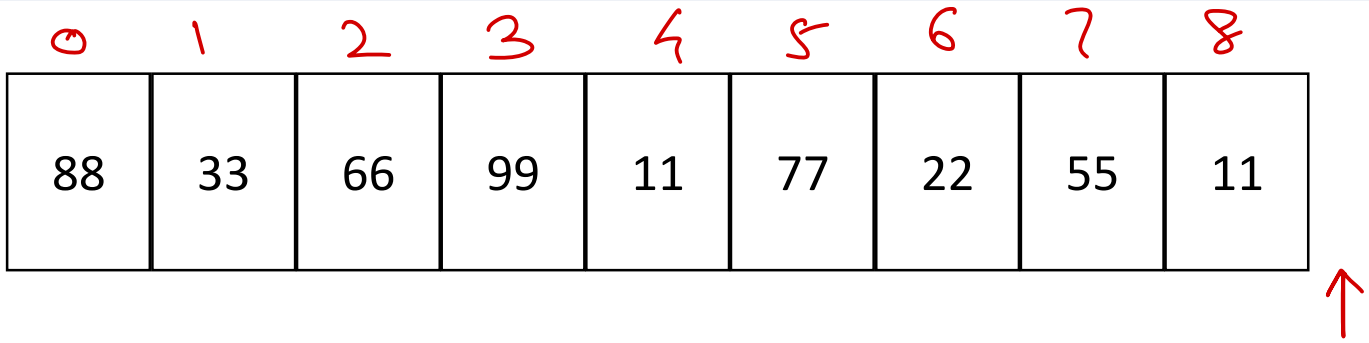
$T \propto \frac{\log n}{\log 2}$

$T \propto \log n$

$O(\log n)$ →

Linear Search

- Find a number in a list of given numbers (random order).



- Time complexity
  - Worst case
  - Best case
  - Average case

```
int linear_search(int a[], int n, int key)
{
    int i;
    for(i=0; i<n; i++){
        if(a[i] == key)
            return i;
    }
    return -1;
}
```

Thank you!

Nilesh Ghule <nilesh@sunbeaminfo.com>