K.R. MANGALAM UNIVERSITY
THE COMPLETE WORLD OF EDUCATION

Course: Data Structure

(Course Code : ENCS205)

UNIT-1: Foundations of Data Structures

School of Engineering & Technology
K.R. Mangalam University, Gurugram (Haryana)

Session 4:
Algorithm Efficiency Basics

Session 4: Learning Objectives

By the end of this session, will be able to:

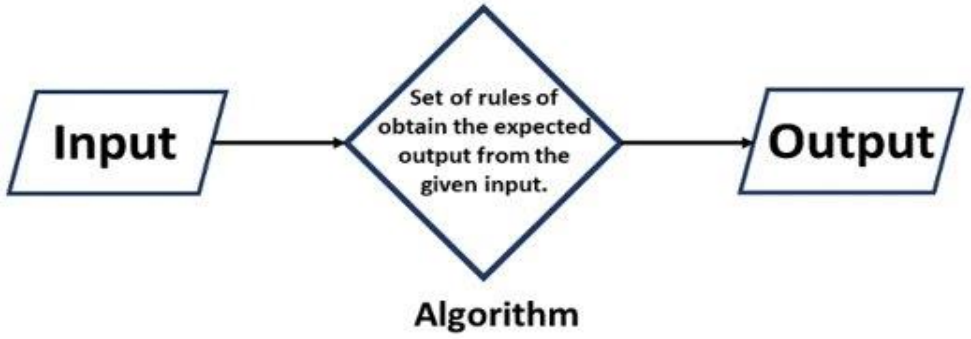
- Define and understand algorithms
- Identify key algorithm properties
- Describe steps in algorithm development
- Analyze time and space complexity
- Understand best, worst, and average cases

Introduction To Algorithm

- An algorithm is a step-by-step recipe for solving an instance of a problem.
- Every single procedure that a computer performs is an algorithm.
- An algorithm is a precise procedure for solving a problem in finite number of steps.
- An algorithm states the actions to be executed and the order in which these actions are to be executed.
- An algorithm is a well-ordered collection of clear and simple instructions of definite and effectively computable operations that when executed produces a result and stops executing at some point in a finite amount of time rather than just going on and on infinitely.

Algorithm Properties

1. **Input:** Input data, supplied externally (zero or more).
2. **Output:** Result of program.
3. **Finiteness:** In every case, algorithm terminates after a finite number of steps.
4. **Definiteness:** The steps should be clear and unambiguous.
5. **Effectiveness:** An algorithm should be written using basic instructions. It should be feasible to convert the algorithm in a computer program.



Various steps in developing Algorithms

- Devising the Algorithm:
  - It's a method for solving a problem.
- Validating the Algorithm:
  - The proof of correctness of the algorithm.
- Expressing the Algorithm:
  - To implement the algorithm in a programming language.

Example : Algorithm for addition of 10 ns

1. Assign 10 to n
2. Assign 0 to sum
3. Assign 1 to i
4. If (i>n) go to step 9
5. Read x
6. Assign (sum + x) to sum
7. Assign (i+1) to i
8. Go to step 4
9. Print sum
10. Stop

Efficiency of an Algorithm

- Writing efficient programs is what every programmer hopes to be able to do.
- An algorithm is an idea upon which a program is built.
- An algorithm should meet three things:
  - ✓It should be independent of the programming language in which the idea is realized
  - ✓Every programmer having enough knowledge and experience should understand it
  - ✓It should be applicable to inputs of all sizes

Analysis of Algorithm

- It is measured by the amount of resources it uses, the time and the space.
- The time refers to the number of steps the algorithm executes.
- Space refers to the number of unit memory storage it requires.
- An algorithm's complexity is measured by calculating the time taken and space required for performing the algorithm.
- The input size, denoted by n, is one parameter, used to characterize the instance of the problem.

Algorithm Complexity

- An algorithm's complexity is a measure of the amount of data that it must process in order to be efficient.
- **Time Complexity**
  - Time complexity is defined in terms of how many times it takes to run a given algorithm, based on the length of the input.
- **Space Complexity**
  - The amount of memory used by a program to execute it is represented by its space complexity

```
#include<stdio.h>
void main(){
    int i, n, sum, x;
    sum =0;
    printf("\n Enter no. of the data to be added");
    scanf("%d", &n);
    for (i=0;i<=n;i++)
    {
        scanf("%d",&n);
        sum =sum+x;

    }

    printf("\n sum = %d", sum);
}
```

Memory Space Requirement:
Space required to store the variable I, n, sum and x

=> 2+2+2+2 =8 (bytes)

	Frequency	Computation time
sum =0;	1	t1
printf("\n Enter no. of the data to be added");	1	t2
scanf("%d", &n);	1	t3
for (i=0;i<=n;i++)	n+1	(n+1)t4
{		
scanf("%d",&n);	n	n *t5
sum =sum+x;	n	n* t6
}		
printf("\n sum = %d", sum);	1	t7

$\Rightarrow$**Total computation time** (t) = t1+t2+t3+ (n+1)t4 +nt5+ nt6+ t7 = n(t4+t5+t6) = kn

Time Complexity

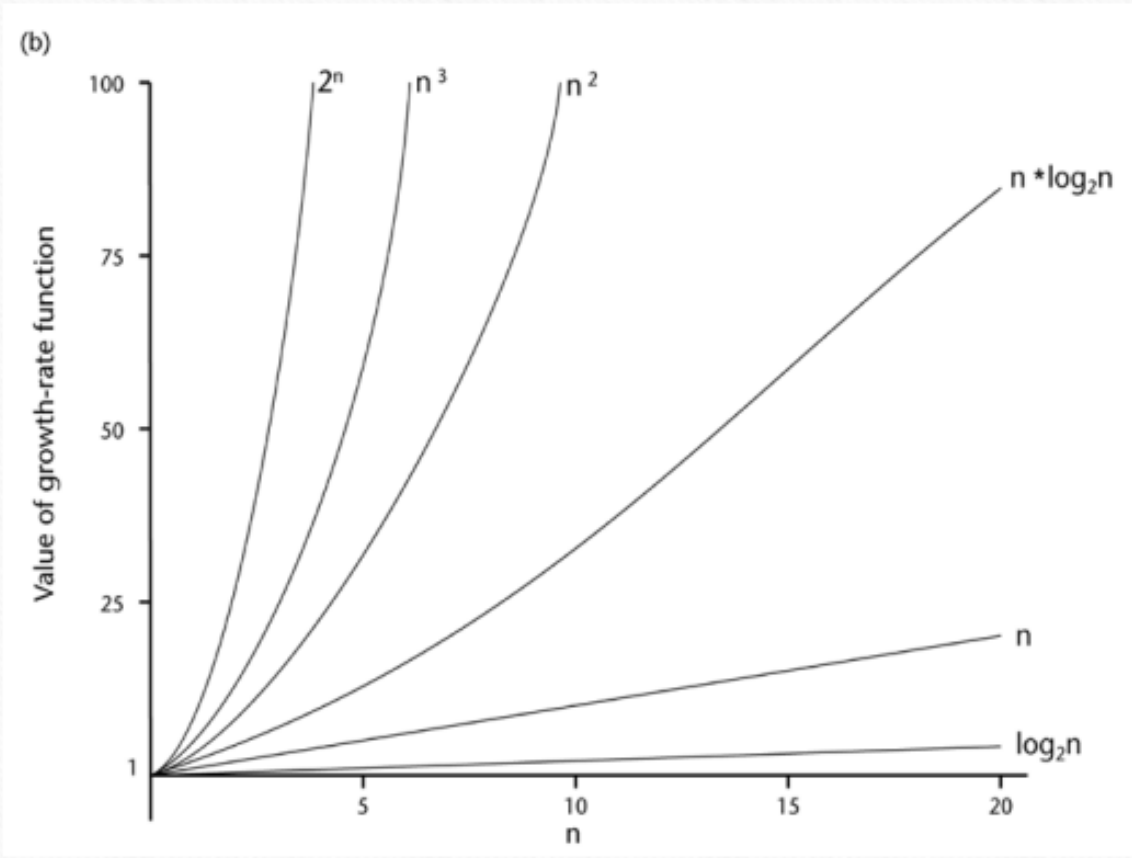
- **Worst Case:** It is the longest time that an algorithm will use over all instances of size n for a given problem to produce a desired result.
- **Average Case:** It is the average time(or average space) that the algorithm will use over all instances of size n for a given problem to produce a desired result.
- **Best Case:** It is the shortest time (or least space) that the algorithm will use over all instances of size n for a given problem to produce a desired result.

Space Complexity

- Space Complexity of a program is the amount of memory consumed by the algorithm (apart from input and output, if required by specification) until it completes its execution.
- The way in which the amount of storage space required by an algorithm varies with the size of the problem to be solved.

Growth of Functions (Asymptotic notations)

- The growth rate for an algorithm is the rate at which the cost of the algorithm grows as the size of its input grows.



Function	Growth Rate Name
c	Constant
$log\ N$	Logarithmic
$log^2 N$	Log-squared
N	Linear
$N\ log\ N$	
N^2	Quadratic
N^3	Cubic
2^N	Exponential

Summary of Session 4

- Learned what an **algorithm** is and its key properties
- Understood steps to **develop and express** algorithms
- Explored **time and space complexity** for efficiency analysis
- Differentiated between **best, worst, and average cases**
- Introduced **asymptotic notations** for performance evaluation

Test Your Knowledge

1. Which of the following is **not** a necessary property of a valid algorithm?
   a) Finiteness
   b) Effectiveness
   c) Scalability
   d) Definiteness

2. Consider the following pseudo code:

```
sum = 0
for i = 1 to n
    for j = 1 to i
        sum = sum + j
```

What is the **time complexity** of this algorithm?

a) O(n log n)
b) $O(n^2)$
c) $O(n^3)$
d) O(n)

3. What is the **primary reason** an algorithm must have the property of **effectiveness**?

a) To run on large data
b) To be able to convert it into a computer program
c) To save memory
d) To reduce logical errors

Answers

1. c) Scalability
2. b) $O(n^2)$
3. b) To be able to convert it into a computer program

Thank You

K.R. MANGALAM UNIVERSITY

THE COMPLETE WORLD OF EDUCATION

Recognised under the section 2 (f) of the UGC Act 1956

Empowering the Youth; Empowering the Nation