Data Structure

&

Algorithms

Algorithms

sequence of finite steps

Multiply two numbers

Multiply

finite amount of

Sequence **of**

Steps

a & b

2) Take

c = a \* b

1 Take two numbers

time

3) Return c

Properties of Algorithms

Terminate after finite amount of time

2) Produce

atte alt one output

3) Independent of any programming language

unambiguous (Deterministic)

")

Is enie a

valid

algorithm

or

not??

code

2x3

273

6

10

While (True) {

Infinite times

Not a

System.out.println(

Valid algorithm

"1

му

name is

Priya")

4 Array

4 Linked List

4 Stack

4 Queue

J

4 Tree &

many

Data Structure

to store the data in

more

a

structured

manner

Problem

Statement

Approach 1

n=10

1 to 10

Loop

int sum=0;

sum=0;

Sum of natural number

for (int i = 1; i<=m; i++) {

given

en the value of

3

by

the wer?

Approach (Better approach)

NO Loop

int sum=0.

1 time

sum

=

(n+1).

sum +=

i

or

return sum'

Time complexity = O(n)

Lineer time

complexity

rehim

sum.

Time complexity=00

constant time

Sumtime complexity

natural

numbers

Time & space complexity

Time complexity

# times any Statement is repeating

CPU Time → Execute the code

+

Asymtotic Notations objective →→ Lower time

time complexity (Big-

4 worst cale

average

cale

→

T

y call

-o)

time complexity

complexity (0) complexity (12)

beet case time complexity

complexity

Theta

omega

Example 1

0 = 10

for (int i=0; i<m; i++) α

tem. out.

out. pointin ("Hi")

System.

Time

complexity = O(n)

3=10

10 timel

Example 2

1=0

J=0

J = 1

-

Hi

Hi

Y

=2

100

र

for (int i=0; i2m; i++) α

for (int j=0; j < n; j++) {

out-printin ("Hi");

System.

tem.

نو

(100) til

timel

2

2 2

J=0

J=1

4 times

Hi

Hi

Time comp

complexity = 0 (n=)

Note:

1) Loops in

а

code

3

2) Bigger Loop

# timal statement is repeating

(maxima)

CPU

operations

O(n) < O(n)ential

0(n!) ol

Exponential time

O(n3)

complexity

O(n2)

O(n)

O(logn)

70(1) (minima)

Input size

Constant

time

complexity

Example 3

1 Array

Data

Structure

binary Search (

array size

binary Search (int) aor; int x) α

Low = 0;

high =n-1;

while (Low <= high) a

mid = (low + high) / 2;

if (adr (mid) = = x) <

y

return mid;

else if (aro (mid) <x) {

لا

Low = mid +1;

Array

d

elee &

}

high = mid-1;

Return -1;

→ sorted array

प

J

2

**3**

Ч

6

10

12

3)=10

Array (3)

X = 10

Output = 3

x=13

output = |

LOW = 0

high=4

mid =2

arr (2)=6= = 10 *X*

تو

n/

3/2

6510

eight

Low = 3

Dide of 6

mid = 3+ Y

3

0th iteration

1st iteration

2nd iteration

ok

k iterations

r searching

element

found

log

3=2

K=

11

K

K

= 1092

log

#iterations

Time complexity = 0 (logn)