Placement Drive Coding Round Percentage Breakage

30%	20%	20%	20%	10%
Binary Search (Recursion) + Back Tracking + Dynamic Programming	Arrays Strings Search Sort Algos	Trees Graphs Heaps Tries	Bit Masking Greedy Special Algos	

* VTU Syllabus :→ (Search Space) (Array) → Sorted Array
* Square root (Integral Part) of a number using Binary Search.

$x \times x = 36$

int sqrt (int n) {

n = 36 → 6

0 — 36
s e

e = m-1

mid = (s+e)/2 = 36/2 = 18

18×18 = 324

0 — 17 → 17/2 = 8

0 — 7

4 — 7

6 — 7

8×8 = 64
e = m-1

7/2 = 3

3×3 = 9

13/2

6×6 = 36

ans = mid = 3,5

s = m+1

11/2 = 5

5×5 = 25

Integral Part

Square 18×18
m m

37 → 6

2	4	8	9	12
0	1	2	3	4

3×3 = 9 < 36
ans = 3
s = m+1

324 > 36
e = m-1

s = 0
e = n-1

0 — 36

s = 0
e = n

35
↳ 6

5×5 = 25 < 36
ans = 5
s = m+1

Dry Run :→

Square == n

n = 64 → 8

16/2 = 7

ans = -1

0 — 64
s e

square = m×m
= 32×32 = 1024

(sq > n) 1024 > 64

e = m-1

0 — 31

7×7 = 49
ans = 7
s = m+1
7+1
= 8

0 — 14
8 — 14 → 8 — 10

8×8 = 64

16/2 = 8

22/2 = 11

121

e = m-1

81

18/2 = 9

e = m-1

* In Binary Search, which are the values that keep changing during the execution of the algorithm?
* The start & the end values keep changing.
* Therefore, we can apply recursion to search for the key.

O(log N) Recursive Binary Search

* (i) First Occurrence | (ii) Last Occurrence | (iii) Total Number Of Occurrences of an element in an array using Binary Search :→ log n

First = 2
ans = -1
if (arr[mid] == key)
3 == 3
ans = mid
e = mid - 1
arr[mid] < key
s = m+1
arr[mid] > key
e = m-1

1	2	3	3	3	3	4	5
0	1	2	3	4	5	6	7

Total Occ
Lo - Fo + 1
5 - 2 + 1
3+1 = 4

key = 3 LO = 5
ans = -1
if arr[mid] == key
ans = mid
s = m+1
arr[mid] < key
s = m+1
arr[mid] > key
e = m-1

Important/Expected Coding Questions on Binary Search

(i) Square Root
(ii) First, Last, Total Occurrences
(iii) Missing Element in an Array
(iv) Peak In a Mountain Array
(v) Search in a 2D Matrix
(vi) Aggressive Cows
(vii) Book Allocation Problem
(viii) Painter's Partition Problem

Leet Code
Coding Ninjas

Have been repeated in companies like Microsoft, Oracle, Accenture, Wipro, Netflix, IBM, Infosys, TCS etc.

	0	1	2	3
0	1	3	5	7
1	10	11	16	20
2	23	30	34	60

r×c

array 1D

arr[mid]

[1, 3, 5, 7, 10, 11, 16, 20, 23, 30, 34, 60]

s = 0, e = r×c - 1
= 4×3 - 1 = 11

mid = (s+e)/2 = 11/2 = 5

Matrix

col = matrix[0].length = 4

→ r0
→ r1
→ r2

row = m/col
col = m % col
{row} {col}

int element = matrix[1][2]

1	3	5	7	10	11	16	20	23	30	34	60
0	1	2	3	4	5	6	7	8	9	10	11

m = (s+e)/2 = 11/2 = 5

5/4 = 1
5 % 4 = 1

k = 5

ele == target
return true;
ele < target → s = m+1; or
e = m-1;

s = 0, e = r×c - 1 ⇒ 3×4 - 1
= 12 - 1
= 11

row = mid / col
col = mid / col
element = matrix[row][col]
element == target
→ true
element < target
→ s = m+1
or → e = m-1
false

Bubble Selection Insertion

Time Complexity
$O(n^2)$

Nested For Loops
for (n steps) { O(n)
O(n)×O(n) for (n steps) { O(n)
= $O(n^2)$ } }

7, 6, 4, 3
4 elements

Rounds | Passes | Iterations

(1)
7	6	4	3
6	7	4	3
6	4	7	3
6	4	3	7
0 1 2 3
len - 2

(2)
6	4	3	7
4	6	3	7
4	3	6	7
OOB skip
len - 1

(3)
4 3 6 7
3 4 6 7
Sorted

n → ele
n-1 passes
1 → n-1 iterations

Object → Heap
×
static
psvm

Selection Sort :→ (swap)

0	2	3	6	1	5
0	1	2	3	4	5

minIndex = i
minIndex = j

i = 0
mI
j = i+1 to j < n
1+1 = 2
i++
⇒ if (minIndex != i)
swap (arr[i], arr[mI])

Insertion Sort :→ (UNO) Cards

Shift:
temp

8 5 2 9 6 1
5 8 2 9 6 1
2 5 8 9 6 1
2 5 6 8 9 1
1 2 5 6 8 9

temp
-1
?
j+1

j = -1

OOPs Association

git status

IS-A
extends
inheritance

HAS-A
variable

Aggregation
Car MP3 player
Loose coupling

Composition
Car Engine
Tight coupling

SDLC