Data Structures

Searching Technique – Binary Search

CODE FORTHINGS

Introduction

Introduction

- It is fast and Efficient Searching Technique
- List should be in a sorted order
- It can be implemented using

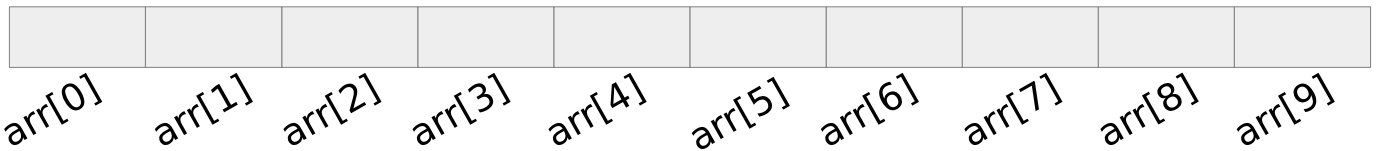
  Iterative

  Recursive

Concept

arr[SIZE]

SIZE = 10

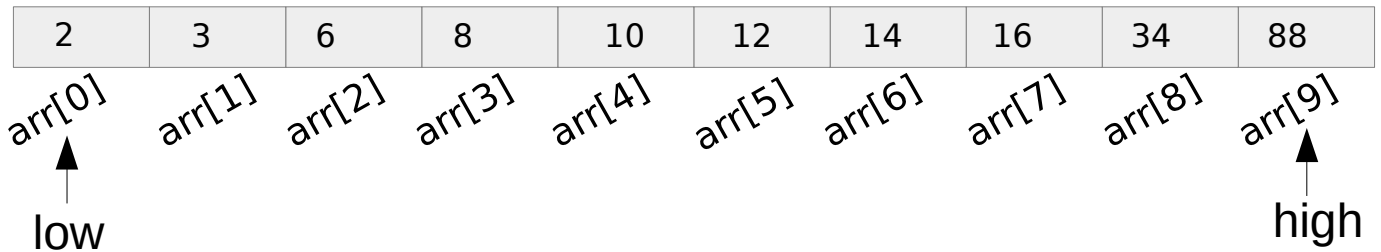


Concept

arr[SIZE]

key = 12

SIZE = 10

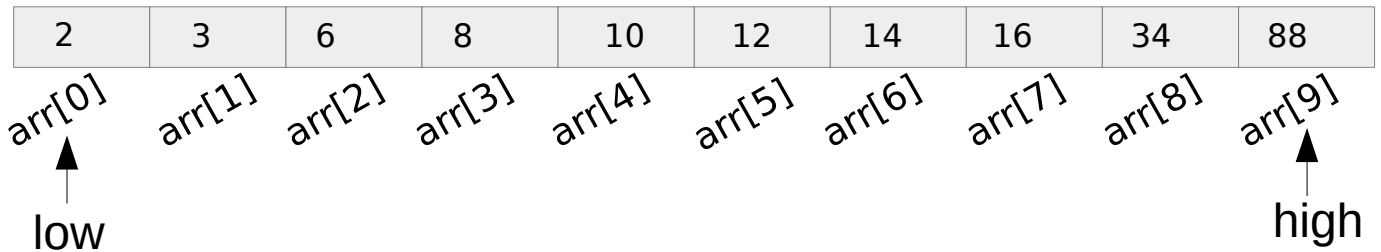


Concept

arr[SIZE]

key = 12

SIZE = 10



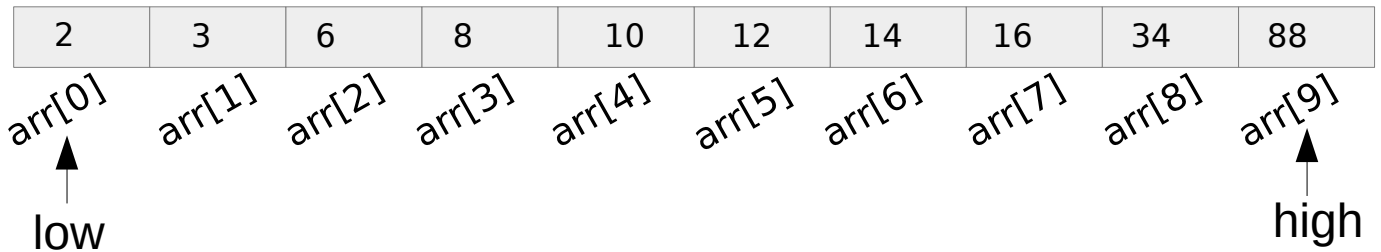
low high mid = (low+high)/2

Concept

arr[SIZE]

key = 12

SIZE = 10



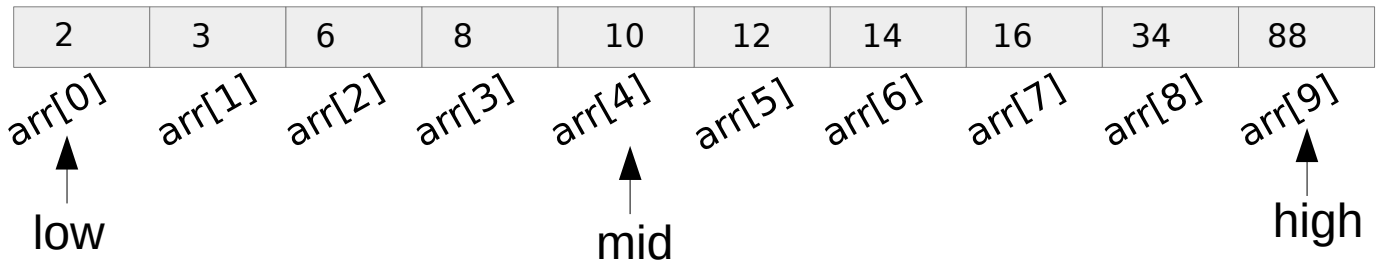
low	high	mid = (low+high)/2
0	9	mid = (0+9)/2 = 4

Concept

arr[SIZE]

key = 12

SIZE = 10



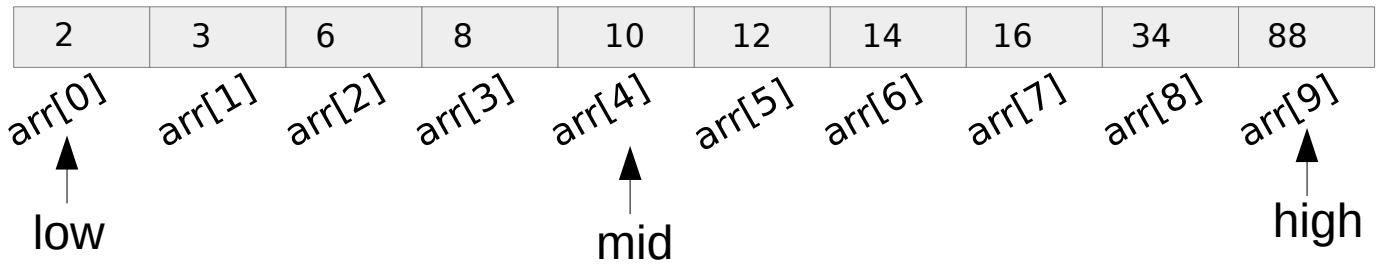
low	high	mid = (low+high)/2
0	9	mid = (0+9)/2 = 4

Concept

arr[SIZE]

key = 12

SIZE = 10



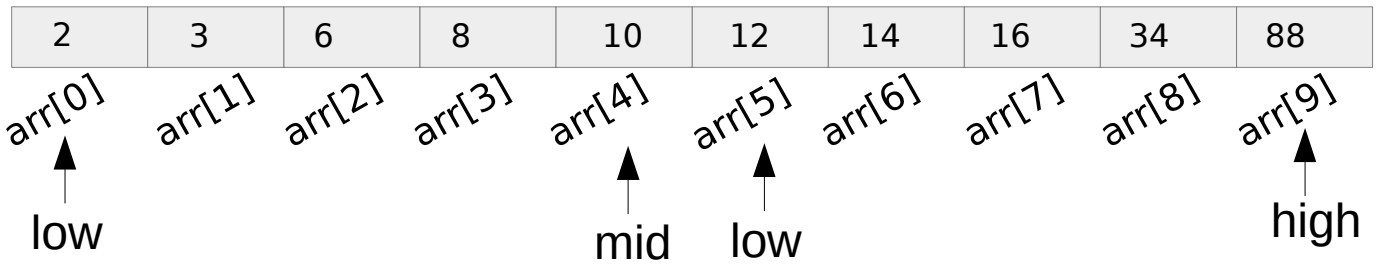
low	high	mid = (low+high)/2
0	9	mid = (0+9)/2 = 4
5	9	

Concept

arr[SIZE]

key = 12

SIZE = 10



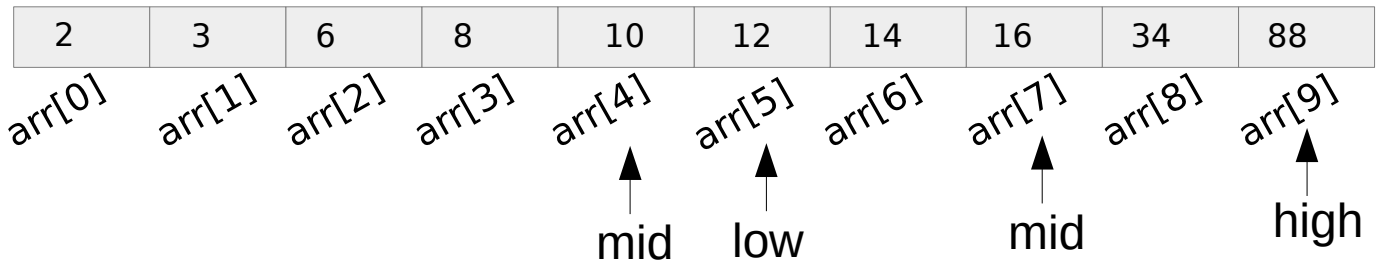
low	high	mid = (low+high)/2
0	9	mid = (0+9)/2 = 4
5	9	

Concept

arr[SIZE]

key = 12

SIZE = 10



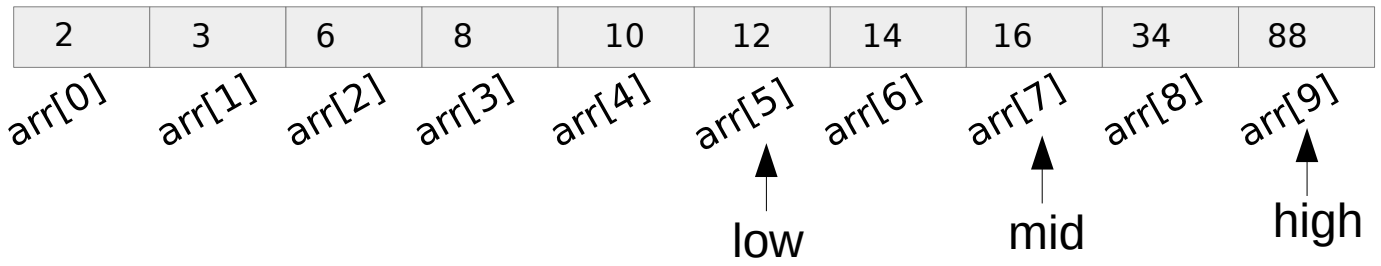
low	high	mid = (low+high)/2
0	9	mid = (0+9)/2 = 4
5	9	mid = (5+9)/2 = 7

Concept

arr[SIZE]

key = 12

SIZE = 10



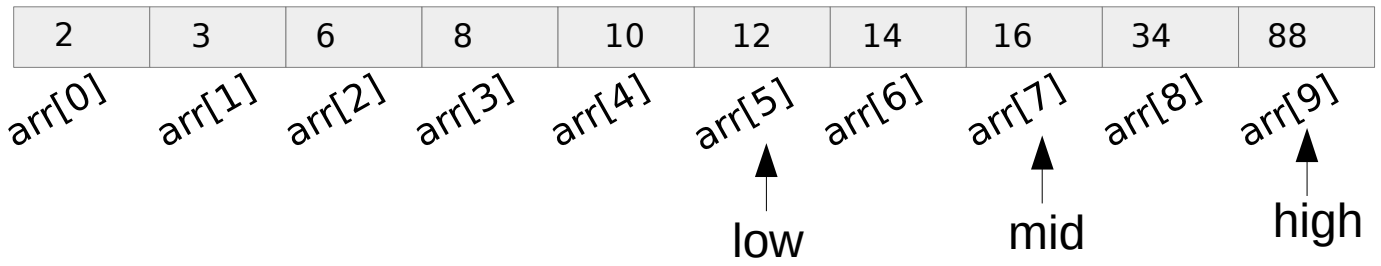
low	high	mid = (low+high)/2
0	9	mid = (0+9)/2 = 4
5	9	mid = (5+9)/2 = 7

Concept

arr[SIZE]

key = 12

SIZE = 10



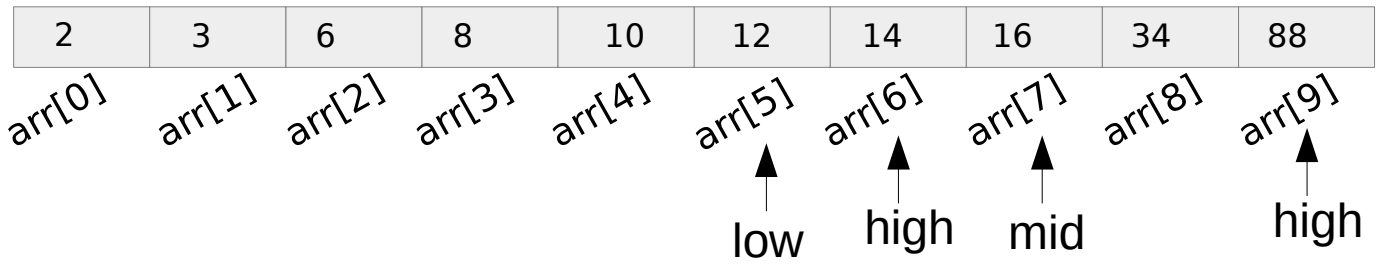
low	high	mid = (low+high)/2
0	9	mid = (0+9)/2 = 4
5	9	mid = (5+9)/2 = 7
5	6	

Concept

arr[SIZE]

key = 12

SIZE = 10



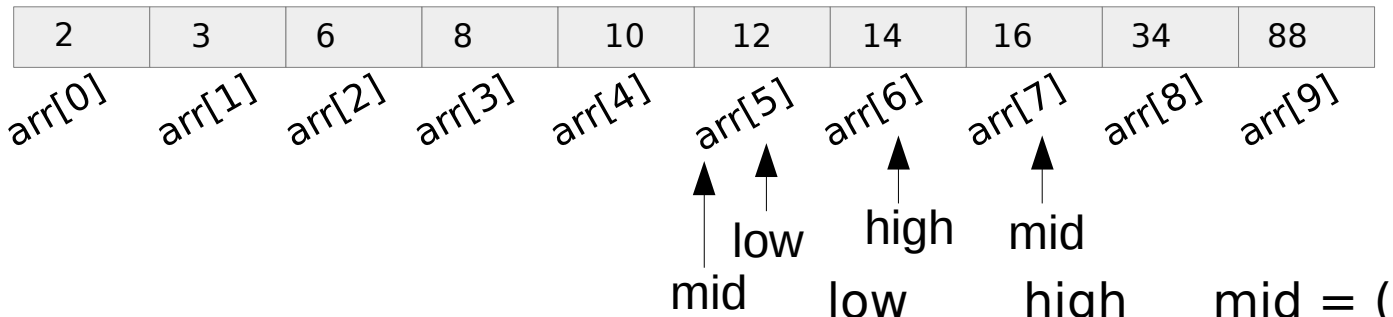
low	high	mid = (low+high)/2
0	9	mid = (0+9)/2 = 4
5	9	mid = (5+9)/2 = 7
5	6	

Concept

arr[SIZE]

key = 12

SIZE = 10



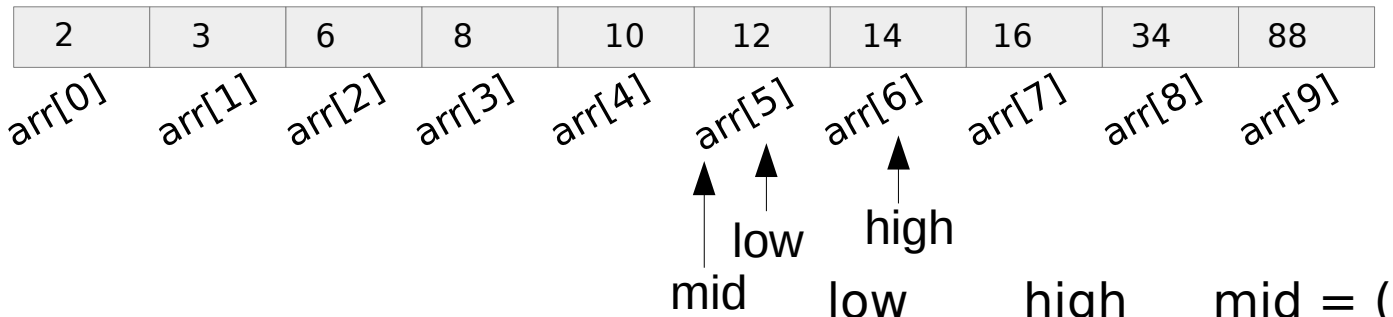
low	high	mid = (low+high)/2
0	9	mid = (0+9)/2 = 4
5	9	mid = (5+9)/2 = 7
5	6	mid = (5+6)/2 = 5

Concept

arr[SIZE]

key = 12

SIZE = 10



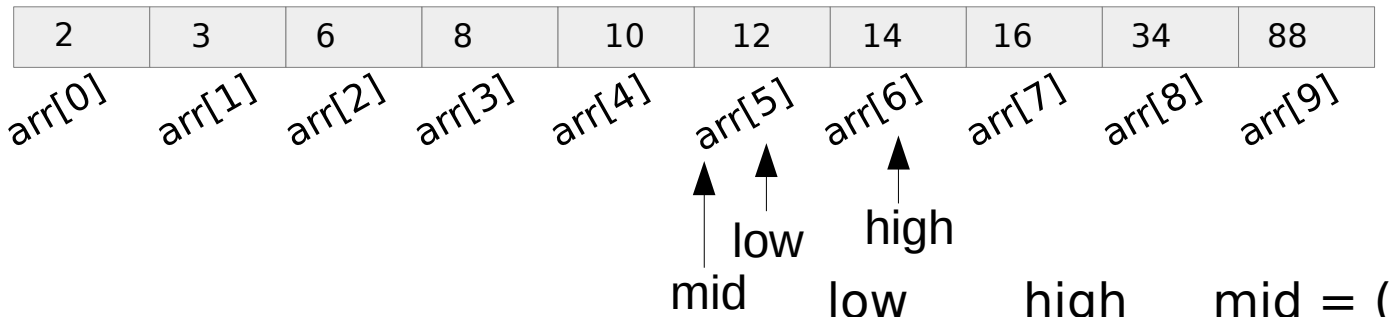
low	high	mid = (low+high)/2
0	9	mid = (0+9)/2 = 4
5	9	mid = (5+9)/2 = 7
5	6	mid = (5+6)/2 = 5

Concept

arr[SIZE]

key = 12

SIZE = 10



low	high	mid = (low+high)/2
0	9	mid = (0+9)/2 = 4
5	9	mid = (5+9)/2 = 7
5	6	mid = (5+6)/2 = 5

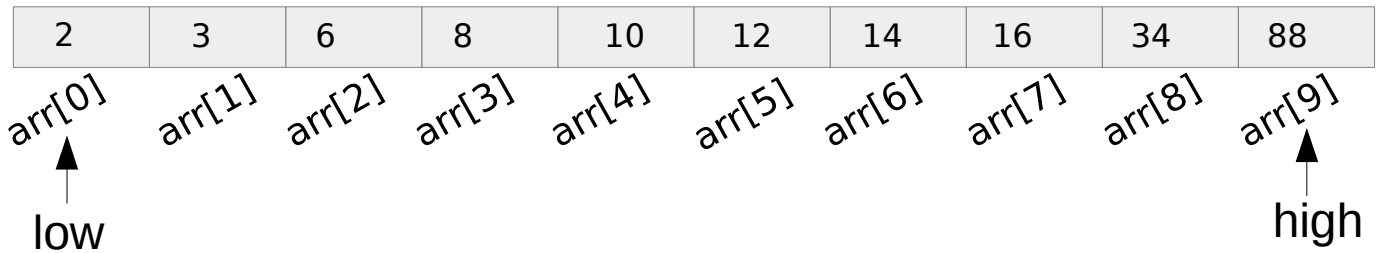
mid =5

Concept

arr[SIZE]

key=5

SIZE = 10

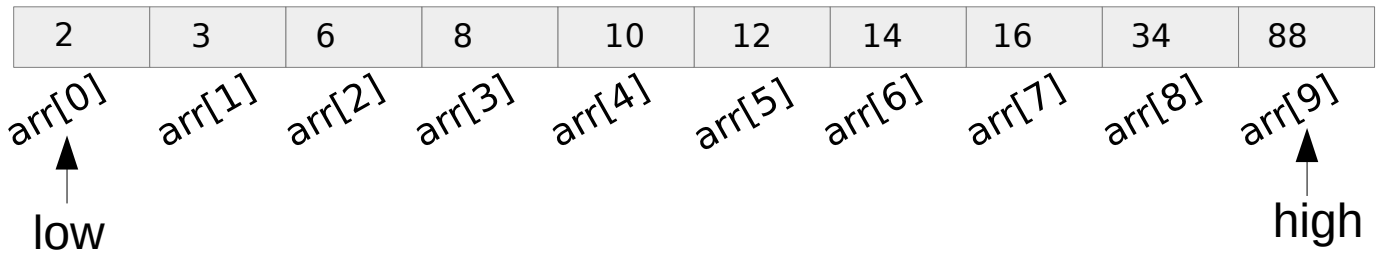


Concept

arr[SIZE]

key=5

SIZE = 10



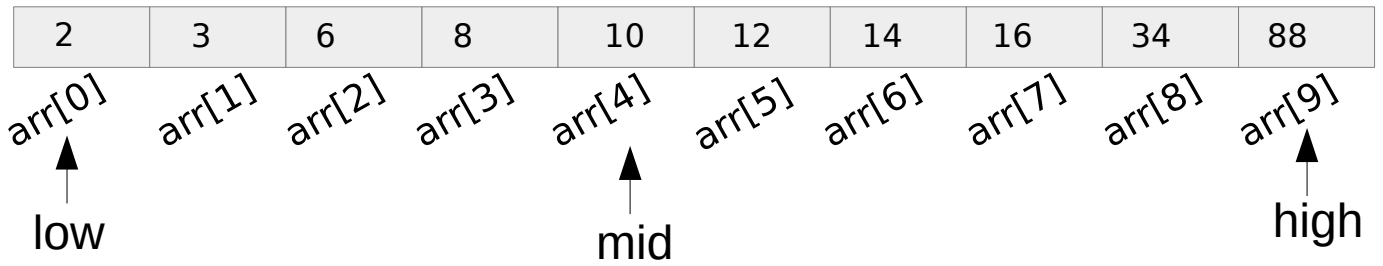
low	high	mid = (low+high)/2
0	9	mid = (0+9)/2 = 4

Concept

arr[SIZE]

key=5

SIZE = 10



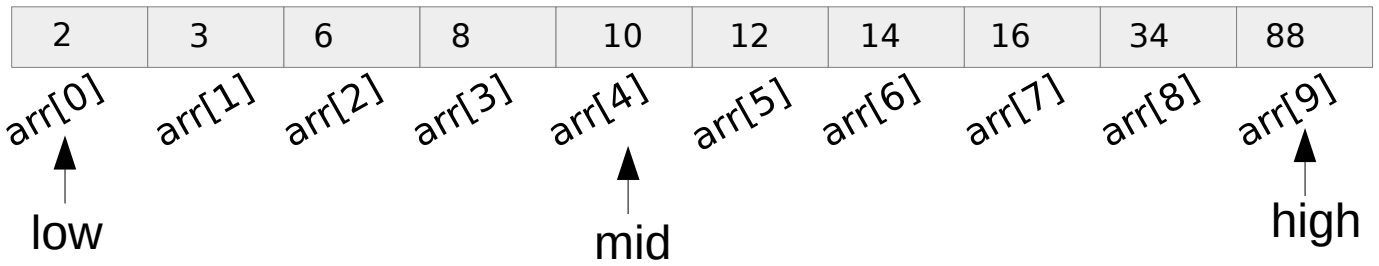
low	high	mid = (low+high)/2
0	9	mid = (0+9)/2 = 4

Concept

arr[SIZE]

key=5

SIZE = 10



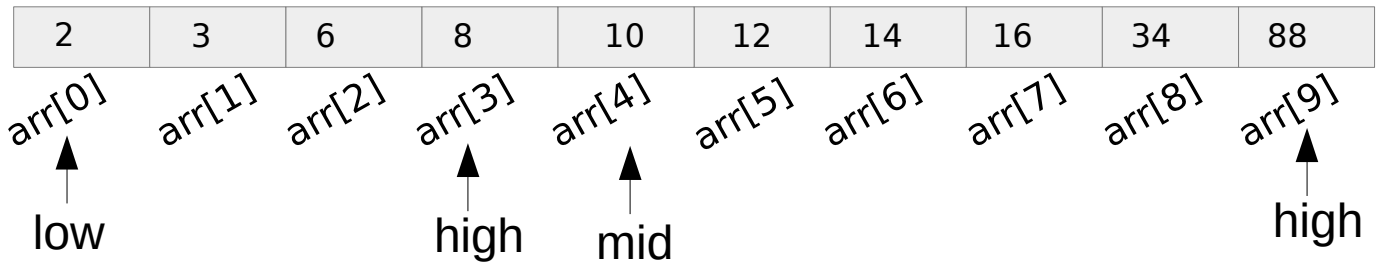
low	high	mid = (low+high)/2
0	9	mid = (0+9)/2 = 4
0	3	

Concept

arr[SIZE]

key=5

SIZE = 10



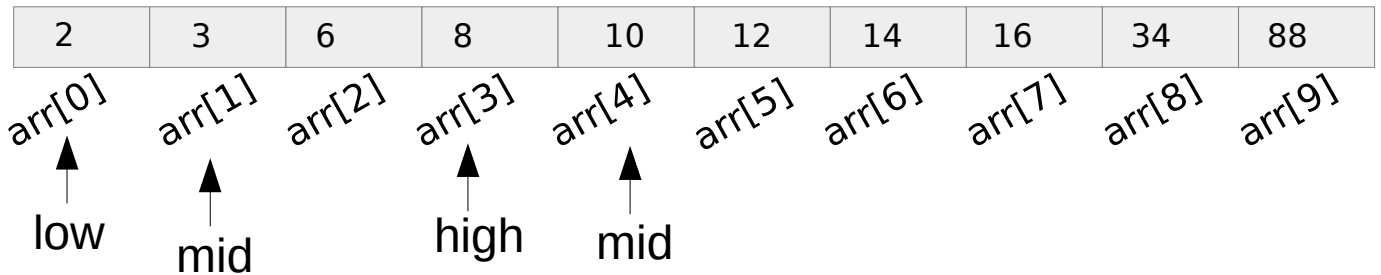
low	high	mid = (low+high)/2
0	9	mid = (0+9)/2 = 4
0	3	

Concept

arr[SIZE]

key=5

SIZE = 10



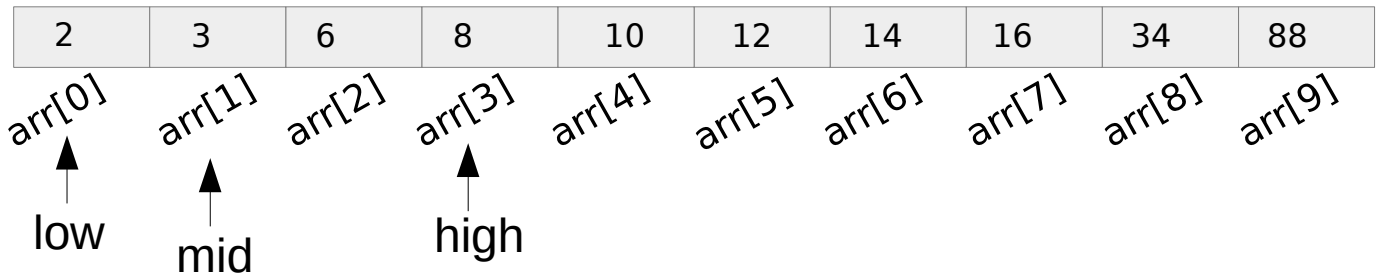
low	high	mid = (low+high)/2
0	9	mid = (0+9)/2 = 4
0	3	mid = (0+3)/2 = 1

Concept

arr[SIZE]

key=5

SIZE = 10



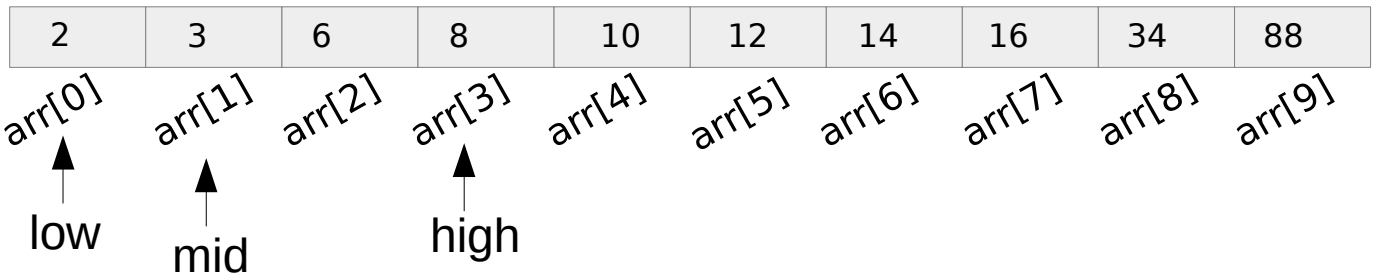
low	high	mid = (low+high)/2
0	9	mid = (0+9)/2 = 4
0	3	mid = (0+3)/2 = 1

Concept

arr[SIZE]

key=5

SIZE = 10



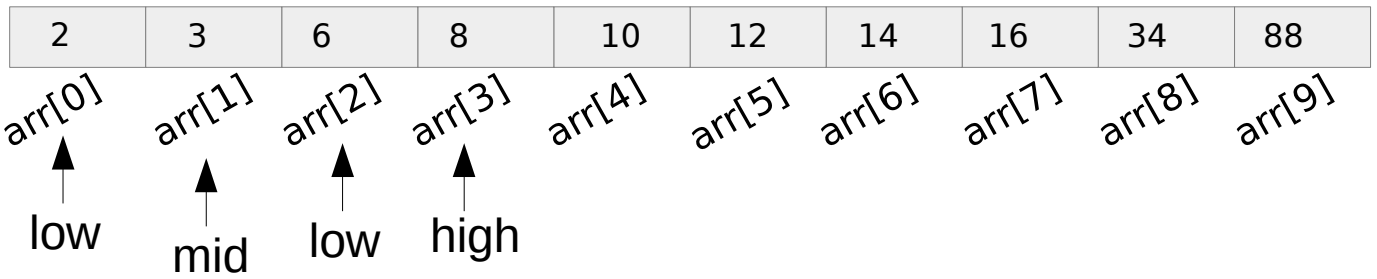
low	high	mid = (low+high)/2
0	9	mid = (0+9)/2 = 4
0	3	mid = (0+3)/2 = 1
2	3	

Concept

arr[SIZE]

key=5

SIZE = 10



low	high	mid = (low+high)/2
0	9	mid = (0+9)/2 = 4
0	3	mid = (0+3)/2 = 1
2	3	

Concept

arr[SIZE]

key=5

SIZE = 10

low	high	mid = (low+high)/2
0	9	mid = (0+9)/2 = 4
0	3	mid = (0+3)/2 = 1
2	3	mid = (2+3)/2 = 2

Concept

arr[SIZE]

key=5

SIZE = 10

low	high	mid = (low+high)/2
0	9	mid = (0+9)/2 = 4
0	3	mid = (0+3)/2 = 1
2	3	mid = (2+3)/2 = 2
2	1	

Concept

arr[SIZE]

key=5

SIZE = 10

low	high	mid = (low+high)/2
0	9	mid = (0+9)/2 = 4
0	3	mid = (0+3)/2 = 1
2	3	mid = (2+3)/2 = 2
2	1	

Concept

arr[SIZE]

key=5

SIZE = 10

low	high	mid = (low+high)/2
0	9	mid = (0+9)/2 = 4
0	3	mid = (0+3)/2 = 1
2	3	mid = (2+3)/2 = 2
2	1	

Time Complexity

Linear Search

- Time Complexity = O(n)

Binary Search

List is sorted

- Time Complexity = O(logn)

List is unsorted

- Time Complexity = TC(sort)+ O(logn)

  = O(nlogn)+O(logn)

  = O(nlogn)

CODE FOR THINGS

Algorithm -Binary Search Iterative