

课程尚未开始 请大家耐心等待

关注微信公共账号，获得最新面试题信息及解答



Facebook: <http://www.facebook.com/ninechapter>

Weibo: <http://www.weibo.com/ninechapter>

Renren: <http://page.renren.com/601712402>

Binary Search & Sorted Array

九章算法IT求职面试培训课程 第2章

www.ninechapter.com

Binary Search

Classical Binary Search

Given an sorted integer array - nums, and an integer - target. Find the first position of target in nums, return -1 if target doesn't exist.

```
public int binarySearch(int[] nums, int target)
```

How we do binary search?



The diagram illustrates the initial state of a binary search. A horizontal array of 10 elements is shown. Above the array, a box labeled 'start' has an arrow pointing to the first element (index 0). Another box labeled 'end' has an arrow pointing to the last element (index 8). The array itself is a table with two rows: 'index' and 'nums'.

index	0	1	2	3	4	5	6	7	8
nums	2	3	5	8	13	21	34	55	89

1. Find 5

How we do binary search?

	start			mid			end		
	↓			↓			↓		
index	0	1	2	3	4	5	6	7	8
nums	2	3	5	8	13	21	34	55	89

1. Find 5, mid=4

How we do binary search?

A diagram illustrating the initial state of a binary search. Three boxes labeled 'start', 'mid', and 'end' are positioned above a table. Arrows point from each box to the cell at index 4 of the 'index' row. The 'end' box is highlighted in yellow.

start	mid	end							
↓	↓	↓							
index	0	1	2	3	4	5	6	7	8
nums	2	3	5	8	13	21	34	55	89

1. Find 5, mid=4, 2. Find it!

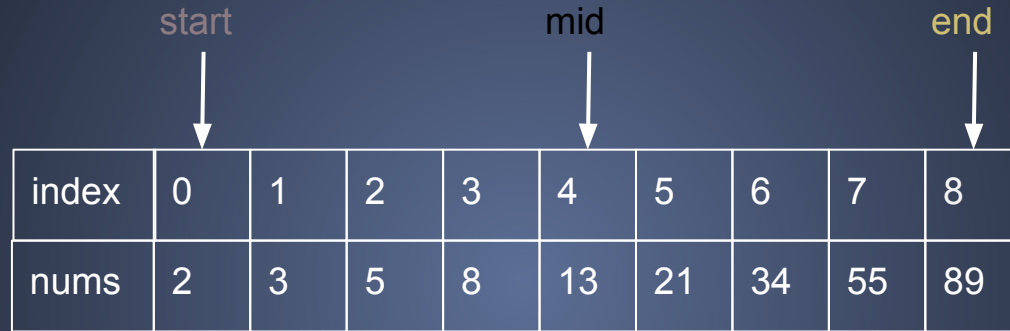
How we do binary search?

			start	mid	end				
			↓	↓	↓				
index	0	1	2	3	4	5	6	7	8
nums	2	3	5	8	13	21	34	55	89

1. Find 5, mid=4, 2. Find it!

2. Find 8, mid=4, 2, 3. Find it!

How we do binary search?



The diagram shows a horizontal array of 10 cells. The first row is labeled 'index' and contains values 0 through 8. The second row is labeled 'nums' and contains values 2, 3, 5, 8, 13, 21, 34, 55, and 89. Above the array, three labels with arrows point to specific indices: 'start' points to index 0, 'mid' points to index 4, and 'end' points to index 8. The 'end' label is colored yellow.

index	0	1	2	3	4	5	6	7	8
nums	2	3	5	8	13	21	34	55	89

1. Find 5, mid=4, 2. Find it!
2. Find 8, mid=4, 2, 3. Find it!
3. Find 14, mid=4

How we do binary search?

					start		mid		end
					↓		↓		↓
index	0	1	2	3	4	5	6	7	8
nums	2	3	5	8	13	21	34	55	89

1. Find 5, mid=4, 2. Find it!
2. Find 8, mid=4, 2, 3. Find it!
3. Find 14, mid=4, 6

How we do binary search?

					start	mid	end		
					↓	↓	↓		
index	0	1	2	3	4	5	6	7	8
nums	2	3	5	8	13	21	34	55	89

1. Find 5, mid=4, 2. Find it!
2. Find 8, mid=4, 2, 3. Find it!
3. Find 14, mid=4, 6, 5

How we do binary search?

start, mid
end



index	0	1	2	3	4	5	6	7	8
nums	2	3	5	8	13	21	34	55	89

1. Find 5, mid=4, 2. Find it!
2. Find 8, mid=4, 2, 3. Find it!
3. Find 14, mid=4, 6, 5, 4. Return -1

Recursion or While-Loop?

Binary Search

A generic binary search template

<http://www.lintcode.com/en/problem/binary-search/>

<http://www.ninechapter.com/solutions/binary-search/>

Keypoints:

1. $\text{start} + 1 < \text{end}$
2. $\text{start} + (\text{end} - \text{start}) / 2$
3. $A[\text{mid}] ==, <, >$
4. $A[\text{start}] A[\text{end}] ? \text{target}$

Search for a range

<http://lintcode.com/en/problem/search-for-a-range/>

<http://www.ninechapter.com/solutions/search-for-a-range/>

Search Insert Position

<http://lintcode.com/en/problem/search-insert-position/>

<http://www.ninechapter.com/solutions/search-insert-position/>

Search in Rotated Sorted Array

<http://lintcode.com/en/problem/search-in-rotated-sorted-array/>

<http://www.ninechapter.com/solutions/search-in-rotated-sorted-array/>

Search in Rotated Sorted Array II

<http://lintcode.com/en/problem/search-in-rotated-sorted-array-ii/>

Linear algorithm, go for-loop!

Search a 2D Matrix

<http://www.lintcode.com/en/problem/search-a-2d-matrix/>

<http://www.ninechapter.com/solutions/search-a-2d-matrix/>

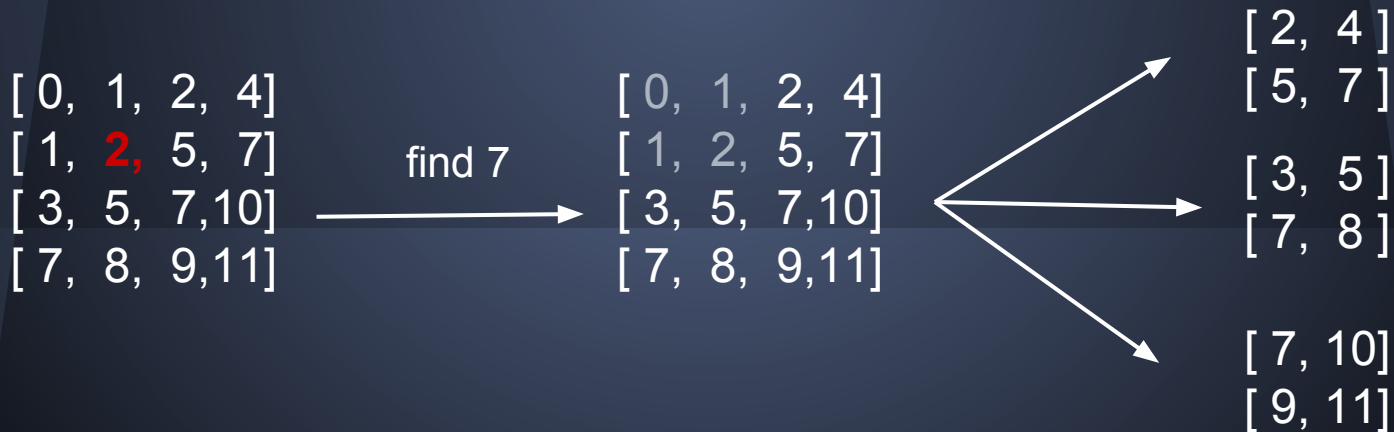
Search a 2D Matrix II

<http://lintcode.com/en/problem/search-a-2d-matrix-ii/>

```
[ 0, 1, 2, 4]
[ 1, 2, 6, 9]
[ 3, 5, 7,10]
[ 7, 8, 9,11]
```

Search a 2D Matrix II

Quadrant Search



Search in a 2D Matrix

28	29	31	36	36	43	49	54	58	64	66	68	68	77	84	89	97	97	97	105
33	38	39	48	53	58	61	62	62	69	70	71	71	77	90	91	103	105	108	110
33	44	51	56	62	63	67	69	74	83	86	91	96	98	104	110	110	110	118	125
41	44	52	59	71	75	83	92	100	107	114	122	122	124	132	133	140	141	145	146
43	49	54	59	72	79	83	101	104	115	122	124	124	124	140	140	140	142	151	154
45	52	55	62	72	80	92	104	111	122	130	132	132	136	149	156	160	165	171	179
51	60	68	73	81	86	94	110	115	129	138	141	146	149	157	163	169	178	186	191
51	61	74	83	92	95	95	111	120	129	138	143	146	155	159	171	180	188	196	201
56	67	81	89	94	98	98	119	123	133	147	149	149	164	167	174	183	188	205	211
62	69	85	95	97	101	105	119	131	139	152	152	160	165	171	178	190	195	212	215
71	72	90	95	106	111	116	121	131	144	155	160	163	173	180	186	197	199	220	221
79	87	99	104	108	115	123	130	140	144	161	166	174	177	189	192	205	209	225	230
84	94	103	106	117	120	129	133	145	153	165	173	175	178	196	199	209	209	231	235
91	96	105	108	120	128	137	141	151	154	169	182	187	192	201	209	214	220	233	237
95	100	109	110	123	137	139	141	159	161	174	184	188	201	205	213	218	228	233	241
101	107	115	119	130	146	155	155	168	173	178	187	190	202	209	221	226	230	237	249
109	109	115	125	131	148	156	164	173	180	180	196	204	212	217	222	232	240	249	253
113	117	124	126	138	151	157	167	181	183	184	204	213	219	223	231	236	242	250	253
115	123	131	138	142	152	160	167	184	190	197	210	215	224	225	238	244	246	256	258
118	126	131	138	149	161	170	176	184	193	206	217	217	225	234	240	249	257	265	267

Search in a 2D Matrix

28	29	31	36	36	43	49	54	58	64	66	68	68	77	84	89	97	97	97	105
33	38	39	48	53	58	61	62	62	69	70	71	71	77	90	91	103	105	108	110
33	44	51	56	62	63	67	69	74	83	86	91	96	98	104	110	110	110	118	125
41	44	52	59	71	75	83	92	100	107	114	122	122	124	132	133	140	141	145	146
43	49	54	59	72	79	83	101	104	115	122	124	124	124	140	140	140	142	151	154
45	52	55	62	72	80	92	104	111	122	130	132	132	136	149	156	160	165	171	179
51	60	68	73	81	86	94	110	115	129	138	141	146	149	157	163	169	178	186	191
51	61	74	83	92	95	95	111	120	129	138	143	146	155	159	171	180	188	196	201
56	67	81	89	94	98	98	119	123	133	147	149	149	164	167	174	183	188	205	211
62	69	85	95	97	101	105	119	131	139	152	152	160	165	171	178	190	195	212	215
71	72	90	95	106	111	116	121	131	144	155	160	163	173	180	186	197	199	220	221
79	87	99	104	108	115	123	130	140	144	161	166	174	177	189	192	205	209	225	230
84	94	103	106	117	120	129	133	145	153	165	173	175	178	196	199	209	209	231	235
91	96	105	108	120	128	137	141	151	154	169	182	187	192	201	209	214	220	233	237
95	100	109	110	123	137	139	141	159	161	174	184	188	201	205	213	218	228	233	241
101	107	115	119	130	146	155	155	168	173	178	187	190	202	209	221	226	230	237	249
109	109	115	125	131	148	156	164	173	180	180	196	204	212	217	222	232	240	249	253
113	117	124	126	138	151	157	167	181	183	184	204	213	219	223	231	236	242	250	253
115	123	131	138	142	152	160	167	184	190	197	210	215	224	225	238	244	246	256	258
118	126	131	138	149	161	170	176	184	193	206	217	217	225	234	240	249	257	265	267

Search in a 2D Matrix

28	29	31	36	36	43	49	54	58	64	66	68	68	77	84	89	97	97	97	105
33	38	39	48	53	58	61	62	62	69	70	71	71	77	90	91	103	105	108	110
33	44	51	56	62	63	67	69	74	83	86	91	96	98	104	110	110	110	118	125
41	44	52	59	71	75	83	92	100	107	114	122	122	124	132	133	140	141	145	146
43	49	54	59	72	79	83	101	104	115	122	124	124	124	140	140	140	142	151	154
45	52	55	62	72	80	92	104	111	122	130	132	132	136	149	156	160	165	171	179
51	60	68	73	81	86	94	110	115	129	138	141	146	149	157	163	169	178	186	191
51	61	74	83	92	95	95	111	120	129	138	143	146	155	159	171	180	188	196	201
56	67	81	89	94	98	98	119	123	133	147	149	149	164	167	174	183	188	205	211
62	69	85	95	97	101	105	119	131	139	152	152	160	165	171	178	190	195	212	215
71	72	90	95	106	111	116	121	131	144	155	160	163	173	180	186	197	199	220	221
79	87	99	104	108	115	123	130	140	144	161	166	174	177	189	192	205	209	225	230
84	94	103	106	117	120	129	133	145	153	165	173	175	178	196	199	209	209	231	235
91	96	105	108	120	128	137	141	151	154	169	182	187	192	201	209	214	220	233	237
95	100	109	110	123	137	139	141	159	161	174	184	188	201	205	213	218	228	233	241
101	107	115	119	130	146	155	155	168	173	178	187	190	202	209	221	226	230	237	249
109	109	115	125	131	148	156	164	173	180	180	196	204	212	217	222	232	240	249	253
113	117	124	126	138	151	157	167	181	183	184	204	213	219	223	231	236	242	250	253
115	123	131	138	142	152	160	167	184	190	197	210	215	224	225	238	244	246	256	258
118	126	131	138	149	161	170	176	184	193	206	217	217	225	234	240	249	257	265	267

First Bad Version

<http://lintcode.com/en/problem/first-bad-version/>

Find a Peak

<http://lintcode.com/en/problem/find-a-peak/>

Sorted Array

Remove Duplicates from Sorted Array I, II

<http://lintcode.com/en/problem/remove-duplicates-from-sorted-array/>
<http://lintcode.com/en/problem/remove-duplicates-from-sorted-array-ii/>

Merge Sorted Array

<http://lintcode.com/en/problem/merge-sorted-array/>

Merge Sorted Array II

<http://lintcode.com/en/problem/merge-sorted-array-ii/>

<http://www.ninechapter.com/solutions/merge-sorted-array/>

Median of Two Sorted Arrays

<http://lintcode.com/en/problem/median-of-two-sorted-arrays/>

<http://www.ninechapter.com/solutions/median-of-two-sorted-arrays/>

Recover Sorted Array

<http://lintcode.com/problem/recover-rotated-sorted-array/>

Rotate String

<http://lintcode.com/en/problem/rotate-string/>

Reverse Words in a String

<http://lintcode.com/en/problem/reverse-words-in-a-string/>

Conclusion

Binary Search

- Exclude half every time

Sorted Array

- If array is sorted, try binary search
- If array is not sorted, try sort it first