Binary Search

UF Programming Team

UFPT Tryouts

Tryouts for teams competing at SER (11/14) will be held this Sunday, September 20th, in the dungeon from 12:00 to 4:00pm

Please arrive around 11:30am so we can make sure everyone has the correct setup for the contest

Format

- 8-12 questions
- Ranked by number of solves, followed by penalty points
  - Penalty points = time (in minutes) problem solved + (20 x number of wrong submissions)
- Questions will cover a wide range of topics

UFPT Tryouts (cont.)

Example of programming contest leaderboard

RK	TEAM	SLV.	TIME	A	B	C	D	E	F	G	H	I	J	K	L	M
1	St. Petersburg National Research University of IT, Mechanics and Optics	13	1801	1 8	1 266	2 60	1 36	1 192	1 66	**1** **249**	1 134	1 78	1 71	1 285	1 152	2 164
2	Moscow State University	11	1293	1 21	**1** **175**	1 39	1 57	1 208	1 71	1 --	**1** **118**	2 78	1 107		1 136	1 263
3	The University of Tokyo	11	1369	1 18	4 288	1 58	1 33	2 177	1 70		1 156	1 110	**1** **47**	**1** **248**	1 84	4 --
4	Tsinghua University	10	1234	1 6	2 --	**2** **29**	1 93	1 277	2 78	2 --	1 168	1 58	2 134		1 107	1 224
5	Peking University	10	1250	**1** **5**		1 41	1 51	1 237	2 81		1 168	1 130	1 92		1 114	3 271
6	University of California at Berkeley	10	1347	1 10		2 70	1 68	1 171	2 78		1 191	1 106	1 140		1 153	3 280
7	University of Zagreb	10	1501	1 8	2 299	1 55	1 84	**1** **115**	2 33			1 233	1 164		1 137	4 273
8	Charles University in Prague	10	1567	1 13		1 168	1 73	1 160	1 106		1 292	1 209	1 52		1 250	2 224
9	Shanghai Jiao Tong University	10	1616	1 9	3 272	1 85	1 107	1 --	1 49		1 244	1 153	1 167		1 116	5 294
10	Massachusetts Institute of Technology	10	1629	1 16		1 55	1 117	1 187	2 111		2 223	**2** **53**	2 252		1 159	5 296

Motivating Problem

Imagine you ask your friend to think of a number between 1 and 100.

You try to guess what number they're thinking of, and they will tell you if the number you guessed is bigger or smaller than their number.

You repeat this process until you find their number.

This shouldn't take long if there are 100 numbers...

... but what if you told your friend to think of a number between 1 and 1,000,000?

How can we quickly find what number they are thinking of?

Binary Search

A *binary search algorithm* finds the position of a target value within a sorted array.

The algorithm begins by comparing the target value to the value of the middle element of the sorted array.

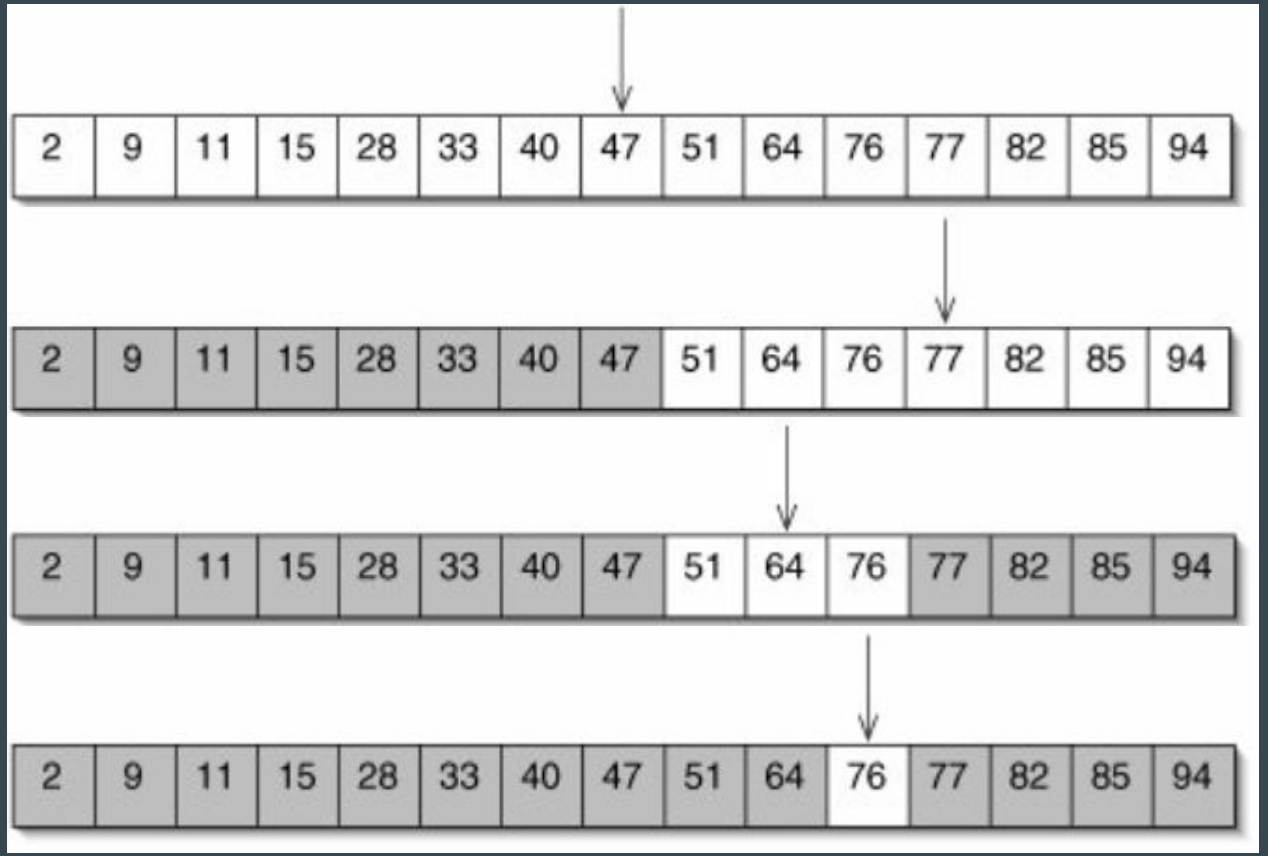
If the target value is equal to the middle element's value, the index is returned.

If the target value is less than the middle element's value, then the search continues on the lower half of the sorted array

If the target value is greater than the middle element's value, then the search continues on the upper half of the sorted array

Visualization

Imagine we are searching for the number 76 in the following sorted array...



Time Complexity

Suppose we have a sorted array containing ***16*** elements

If we search through the entire array to find an element, what is the maximum number of elements we are checking?

$\Rightarrow$ ***16***

If we use binary search, what is the maximum number of elements we are checking?

$\Rightarrow$ ***4***

Time Complexity (cont.)

Suppose we have a sorted array containing N elements

If we search through the entire array to find an element, what is the maximum number of elements we are checking?

$\Rightarrow N$

If we use binary search, what is the maximum number of elements we are checking?

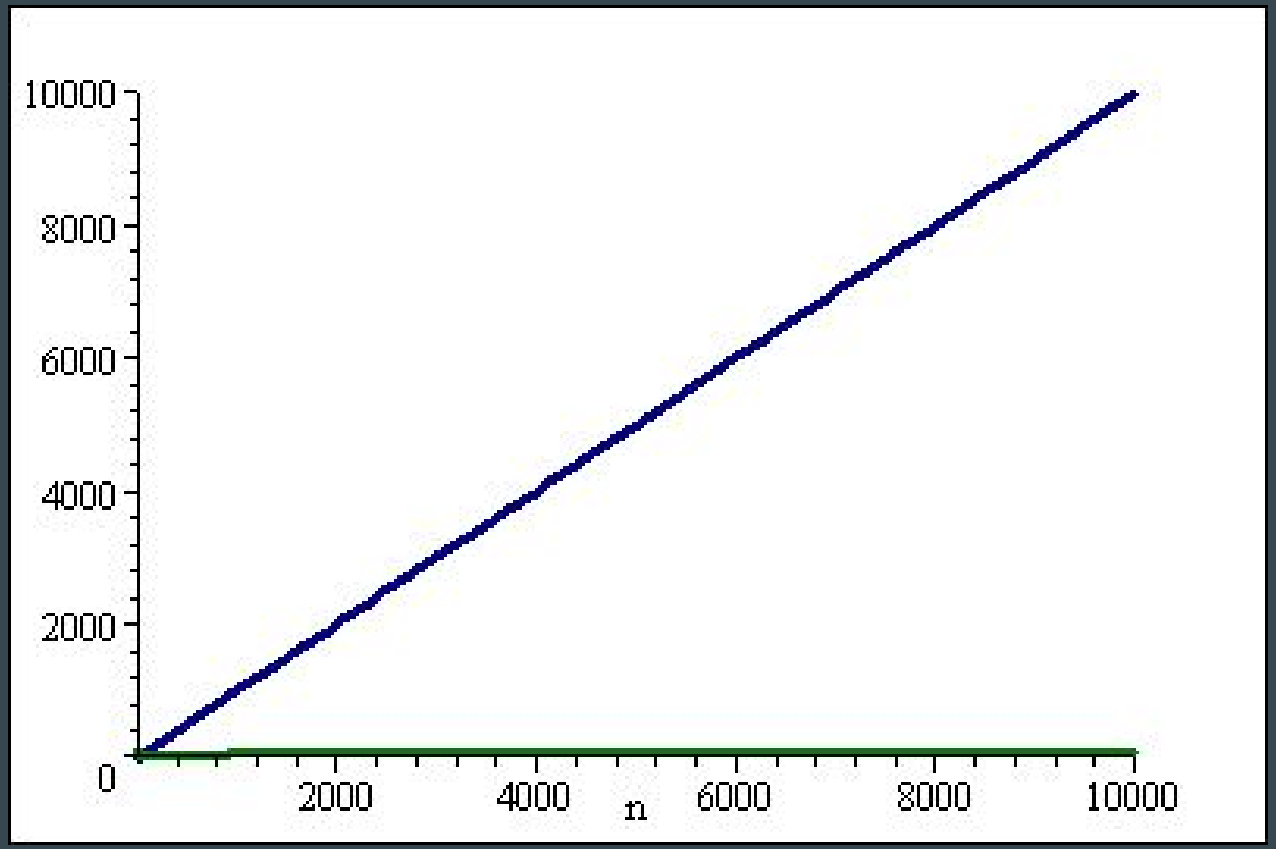
$\Rightarrow log(N)$

Time Complexity (cont.)

N	*log(N)*
10	~3
100	~7
10,000	~13
1,000,000	~20

Looking back at our motivating problem, it would take around 20 guesses to determine a number between 1 and 1,000,000!

Time Complexity (cont.)



Pseudocode

```
lo = 0
hi = N
while ( lo <= hi )
    mid = ( lo + hi ) / 2
    if ( target == a[mid] )
        done
    else if ( target > a[mid] )
        lo = mid
    else if ( target < a[mid] )
        hi = mid
```

Problem Discussion: Part I

ACM Amman Collegiate Programming Contest - Runtime Error

http://codeforces.com/gym/100712

Problem statement: Given an array of size ***N***, and an integer ***K***, find two numbers in the array, ***X*** and ***Y***, such that ***X * Y = K***

Print out the ***X-Y*** pair that satisfies the above condition (if a pair exists); if there are multiple pairs, print the one with the smallest ***X***

Problem Discussion: Part I (cont.)

ACM Amman Collegiate Programming Contest - Runtime Error

We need to first sort the array so that the values are in ascending order

For each number, ***X***, in the array:

1. See if ***K*** is divisible by ***X*** (if not, go to the next number in the array)
2. Set ***Y = K / X***
3. Binary search the array for ***Y***
4. If ***Y*** is in the array, you've found your ***X***
5. If ***Y*** is not in the array, go to the next number in the array

Problem Discussion: Part II

ACM-ICPC World Finals 2015 - Cutting Cheese

https://icpc.kattis.com/problems/cheese

Problem statement: You are given a block of cheese that is 100mm x 100mm x 100mm which has ***M*** spherical holes located throughout; you need to make cuts into the cheese so that there are ***N*** slices of cheese, and each resulting slice has equal volumes (height and width of each slice are 100mm)

Print out the lengths of each slice as you are cutting the block of cheese from left to right (***x = 0*** to ***x = 100***)

Problem Discussion: Part II (cont.)

ACM-ICPC World Finals 2015 - Cutting Cheese

What if there were ***no*** holes in the cheese?

We find the average volume of each slice:

totalVolume = (100 * 100 * 100)

avgVolume = totalVolume / N

We know each slice has a width and height of 100, so we can easily get the lengths;

length = avgVolume / (100 * 100)

Problem Discussion: Part II (cont.)

ACM-ICPC World Finals 2015 - Cutting Cheese

What if there ***are*** holes in the cheese?

We can still find the average volume of each slice

$$totalVolume = (100 * 100 * 100) - \sum volume\ of\ hole_i$$

$$avgVolume = totalVolume / N$$

We know what the average volume of each slice should be, so now what do we do?

Problem Discussion: Part II (cont.)

ACM-ICPC World Finals 2015 - Cutting Cheese

We use ***binary search*** to determine the length of each slice

Let ***P*** be where we made our last cut in the cheese (***P*** is initially 0)

For each slice:

1. Initially set ***lo = P, hi = 100***
2. Set ***mid = (lo + hi) / 2***
3. Calculate the volume from ***P*** to ***mid***
4. If the volume is greater than ***avgVolume***, set ***hi = mid***
5. If the volume is less than ***avgVolume***, set ***lo = mid***
6. If the volume is (roughly) equal to the ***avgVolume***, make the cut

Problem

SPOJ - Aggressive Cows

spoj.com/problems/AGGRCOW