**Problem: Hot or cold.** Your goal is the guess a secret integer between 1 and *N*. You repeatedly guess integers between 1 and *N*. After each guess you learn if it equals the secret integer (and the game stops); otherwise (starting with the second guess), you learn if the guess is *hotter* (closer to) or *colder* (farther from) the secret number than your previous guess. Design an algorithm that finds the secret number in *~ 2 lg N* guesses. Then, design an algorithm that finds the secret number in *~ 1 lg N* guesses.

*Hint*: use binary search for the first part. For the second part, first design an algorithm that solves the problem in *~1 lg N* guesses assuming you are permitted to guess integers in the range -*N* to 2*N*.

**Answers:**

**The First One:Binary Search**

此种解法类似于二分法，每次循环需要猜两次，分别为上限和下限，如果边界值猜中的话直接返回，否则：

如果右边界比较接近key，那么左边界向右边界靠，lo=mid；否则，右边界向左边界靠，high = mid;

由于每次循环猜两次，所以猜的次数为 ~2lgN;

**The Second One:**

Suppose you know that your secret integer is in [a,b], and that your last guess is c.You want to divide your interval by two, and to know whether your secret integer lies in between [a,m] or [m,b], with m=(a+b)/2.

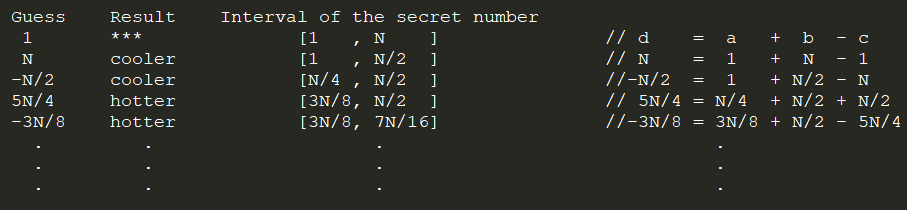
The trick is to guess d, such that (c+d)/2 = (a+b)/2.

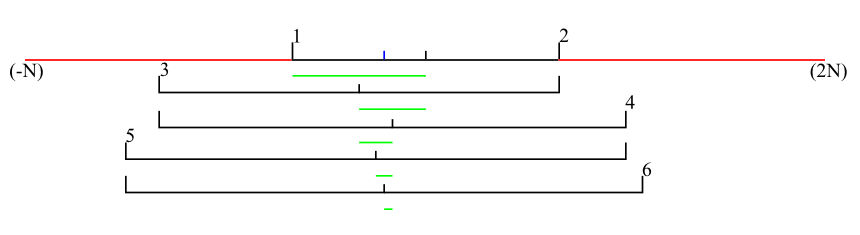
Without loss of generality, we can suppose that d is bigger than c. Then, if d is hotter than c, your secret integer will be bigger than (c+d)/2 = (a+b)/2 = m, and so your secret integer will lie in [m,b]. If d is cooler than c, your secret integer will belong to [a,m].

You need to be able to guess between -N and 2N because you can't guarantee that c and d as defined above will always be [a,b]. Your two first guess can be 1 and N.

So, your are dividing your interval be two at each guess, so the complexity is log(N)+O(1).

A short example to illustrate this (results chosen randomly):





**Q**: Why the guess range in [-N,2N]?

**A**: We still need to prove that our guess will always fall in bewteen [-N,2N];

By recurrence, suppose that c (our previous guess) is in [a-(a+b), b+(a+b)] = [-b,a+2b]

Then d = a+b-c <= a+b-(-b) <= a+2b and d = a+b-c >= a+b-(a+2b) >= -b

Initial case: a=1, b=N, c=1, c is indeed in [-b,a+2\*b]

QED(证明完毕)

理解：此方法的核心就是，假设范围为[a,b],lastguess = c, 要达到lgN的猜测次数，需要使用二分查找的思想，那么nowguess 怎么确定？利用 a+b/2 = (c+d)/2;那么 nowguess = d;例如：

第一次guessnum = 1;

第二次guessnum =N,Colder(假设);查找范围：[1,N] ---> [1,N/2];更新lastguess = N;

第三次:

分析：猜的数字应该使[1,N/2]再缩小为[1,N/4]或[N/4,N/2];那么由(lastguess + nowguess)/2 = (1+N/2)/2;得到nowguess = -N/2;所以

第三次guessnum = -N/2,Colder(假设);说明 -N/2距key比N更远；此时，查找范

围：[1,N/2] ---> [N/4,N/2];

依此类推。。。

该算法在最坏情况下的猜测次数为：log2(N)+2;该种算法与其说在猜数字，不如说在

猜范围；

URL: https://stackoverflow.com/questions/25558951/hot-and-cold-binary-search-game

Author:FlashXT;Date:2018.5.11,Friday;

CopyRight © 2018-2020,FlashXT & turboMan . All Right Reserved.