Бинарный поиск

**Целочисленный двоичный поиск (бинарный поиск)** (англ. *binary search*) — алгоритм поиска объекта по заданному признаку в множестве объектов, упорядоченных по тому же самому признаку, работающий за логарифмическое время.

В основном делится на 3 вида:  
1. В массиве  
2. Вещественный  
3. По ответу

1. В массиве:

Под упорядоченным массивом будем понимать массив, упорядоченный по неубыванию, те. a[1] < a[2] < … < a[N]

У нас имеется заданная своими границами область поиска. Мы выбираем ее середину, и, если искомый элемент меньше, чем средний, то поиск осуществляется в левой части, иначе — в правой. Действительно, сели искомый элемент меньше среднего, то и меньше всех элементов, которые находятся правее среднего, а значит, их сразу можно исключить из рассмотрения. Аналогично для случая, когда искомый элемент больше среднего

Код, осуществляющий бинарный поиск в упорядоченном массиве выглядит так:

while (left<right) {

midlle = (left + right) / 2;

if (a[middle]) <k) left = middle+1;

else right = middle;

}

Перед выполнением этого кода следует присвоить переменным l и r индексы начального и конечного элементов в массиве.

Сложность алгоритма бинарного поиска составляет О(logN), где N — количество элементов в массиве.

2) Вещественный:

*Вещественный бинарный поиск* - бинарный поиск, при котором осуществляется поиск вещественного значения на некотором вещественном отрезке [L, R]. При этом предполагается не точный поиск решения, а поиск решения с некоторой абсолютной погрешностью ε. Здесь ε - максимальная допустимая разность между найденным и точным значением искомого решения. В отличии от целочисленного бинарного поиска процесс продолжается до тех пор, пока отрезок поиска не станет меньше либо равен ε, т.е. пока R-L > ε.

while (fabs(right - left) > eps) {

middle = (left + right) / 2.0;

if (f(middle) < k) left = middle;

else right = middle;

}

3) По ответу:

Во многих задачах в качестве ответа необходимо вывести какое-либо число. При этом достаточно легко сказать, больше ли это число, чем нужно, или меньше, несмотря на то, что вычисление самого ответа может быть довольно трудоемкой операцией. В таком случае мы можем выбрать число заведомо меньшее ответа и число заведомо большее ответа, а правильное решение искать бинарным поиском.

(Код представляет собой обычный бинарный поиск)

Далее решаем задачи

На (1):

[Поиск наиболее близкого](https://informatics.msk.ru/mod/statements/view.php?id=192#1) , [Содержится ли число в массиве](https://informatics.msk.ru/mod/statements/view.php?id=192&chapterid=4#1)

На (2):

[Корень N'ой степени](https://informatics.msk.ru/mod/statements/view.php?id=3516&chapterid=3570#1)

На (3):

[Коровы - в стойла](https://informatics.msk.ru/mod/statements/view.php?id=1966&chapterid=1#1) , [Очень легкая задача](https://informatics.msk.ru/mod/statements/view.php?id=1966&chapterid=490#1)