計算量問題について

アルゴリズム講座第一回

発表者：皆藤 広樹

なぜアルゴリズムとデータ構造を勉強するのか？

そもそもアルゴリズムとは何なのか？

そもそもアルゴリズムとは何なのか？
→日本語訳すると「算法」
→情報学的に言うと処理の手順

そもそもアルゴリズムとは何なのか？
→日本語訳すると「算法」
→情報学的に言うと処理の手順

よりよいアルゴリズムを学ぶことは処理の効率化、すなわち
計算量の削減またメモリ使用量の削減につながる

計算量やメモリ使用量を減らすとどうなるのか？

計算量やメモリ使用量を減らすとどうなるのか？

・PCの処理が少なくなる

→処理時間が短くなる

計算量やメモリ使用量を減らすとどうなるのか？

・PCの処理が少なくなる

→処理時間が短くなる

現実的な処理時間、

またユーザーにとってストレスのないプログラムになる

すなわち、アルゴリズムを学ぶことは

すなわち、アルゴリズムを学ぶことは

PCに負担の少ないプログラム、または
処理時間の短いプログラム作成につながる

実際に計算量を比較してみよう

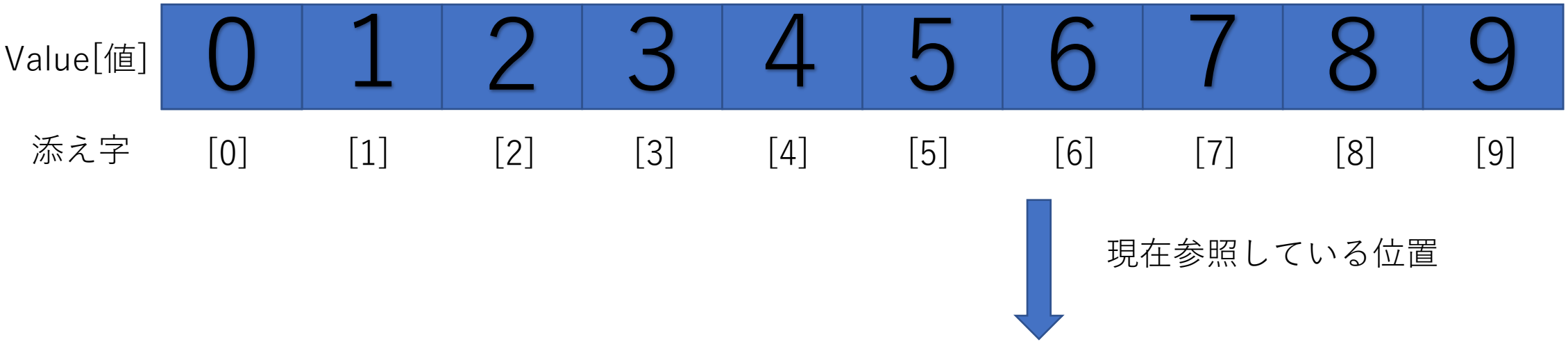
前知識

とある整列された配列から特定の要素を探すプログラム
を考える

Nは配列の要素数
線形探索　最悪計算量　N
二分探索　最悪計算量　$\log N$

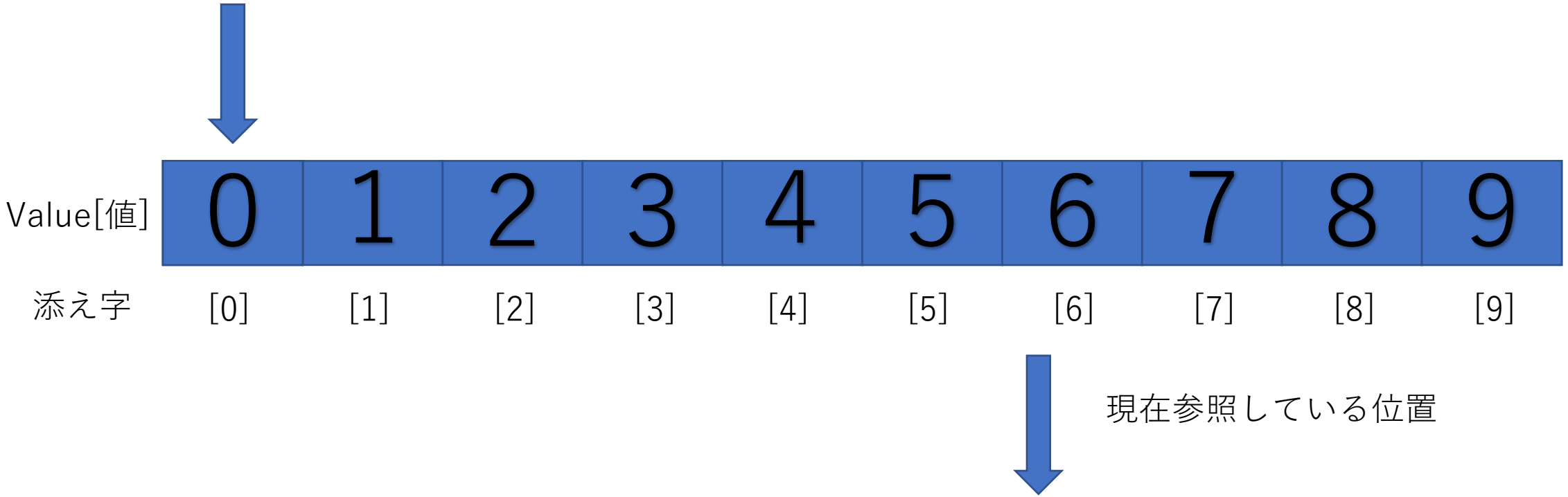
数字の7を探すプログラムを考える。

線形探索



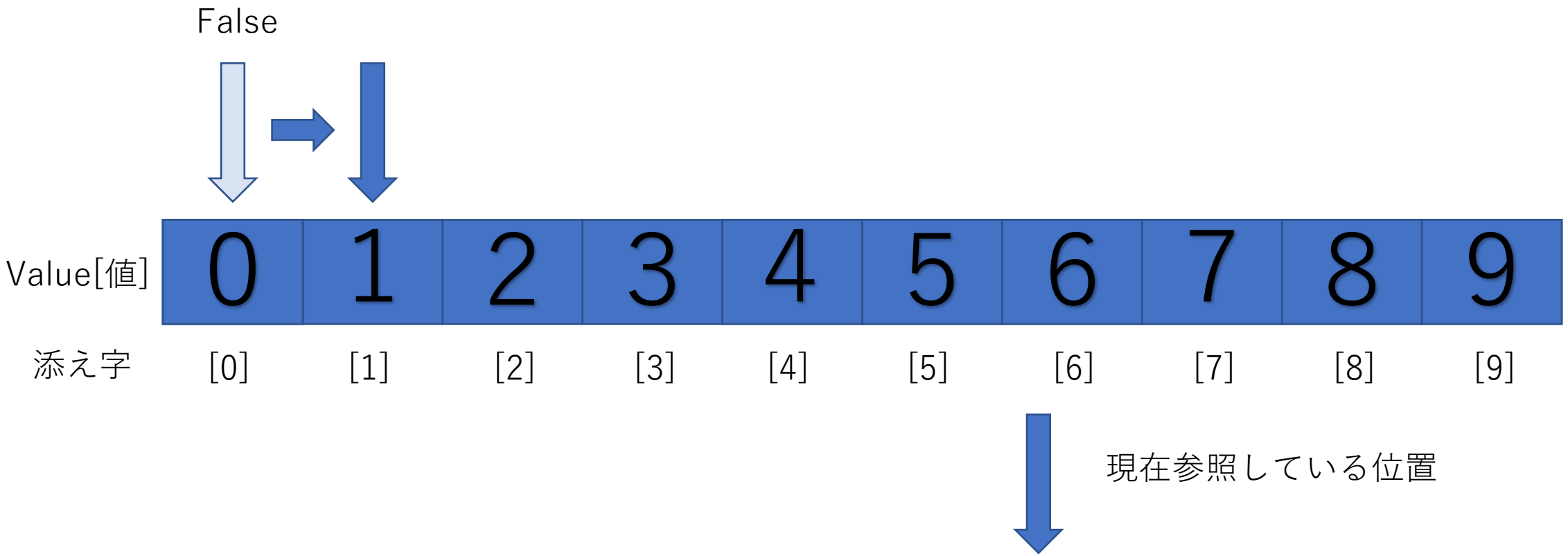
数字の7を探すプログラムを考える。

線形探索



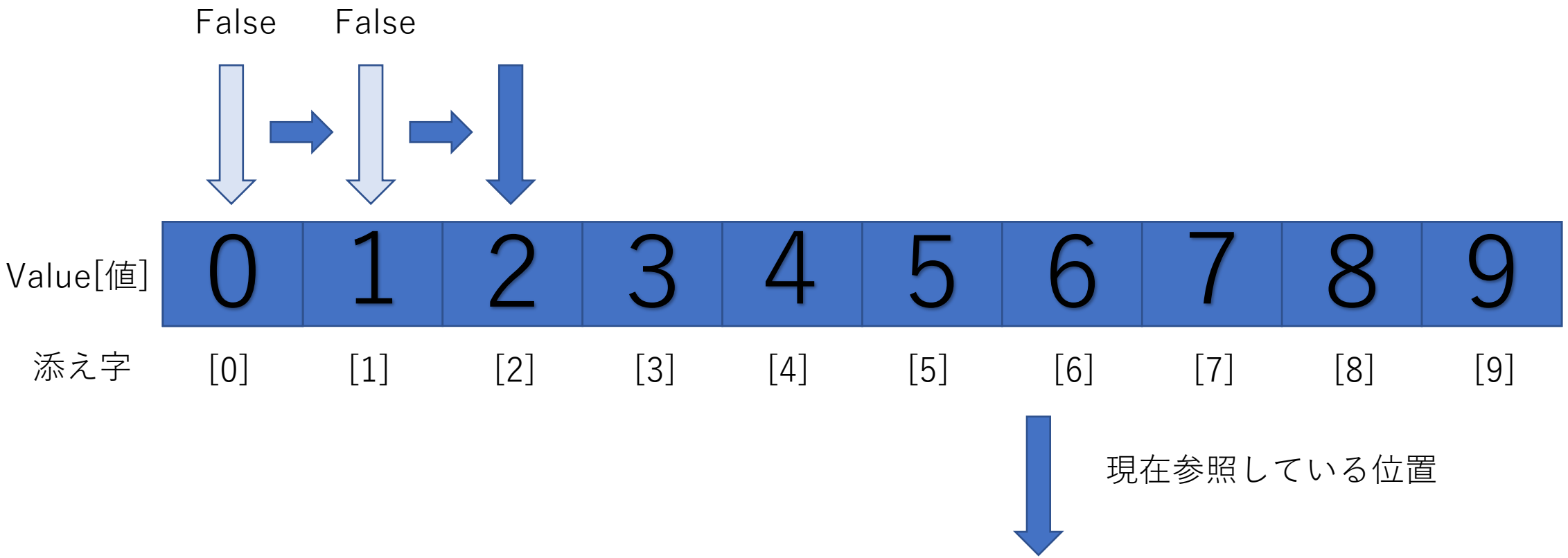
数字の7を探すプログラムを考える。

線形探索



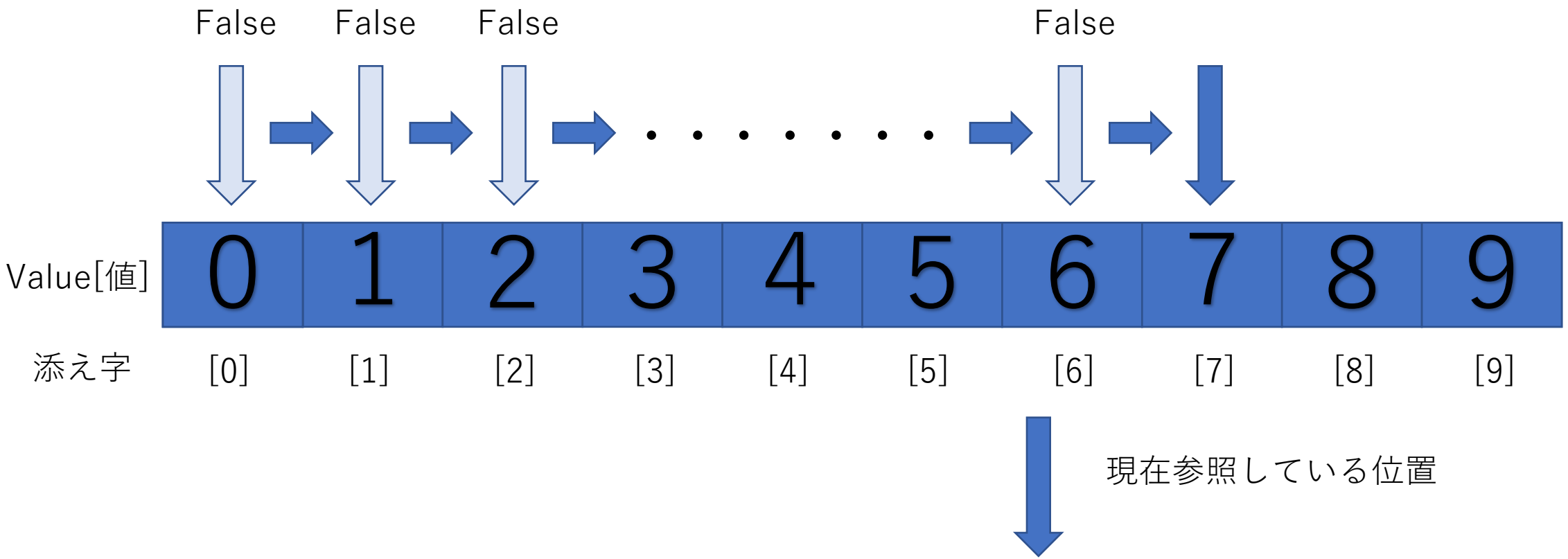
数字の7を探すプログラムを考える。

線形探索



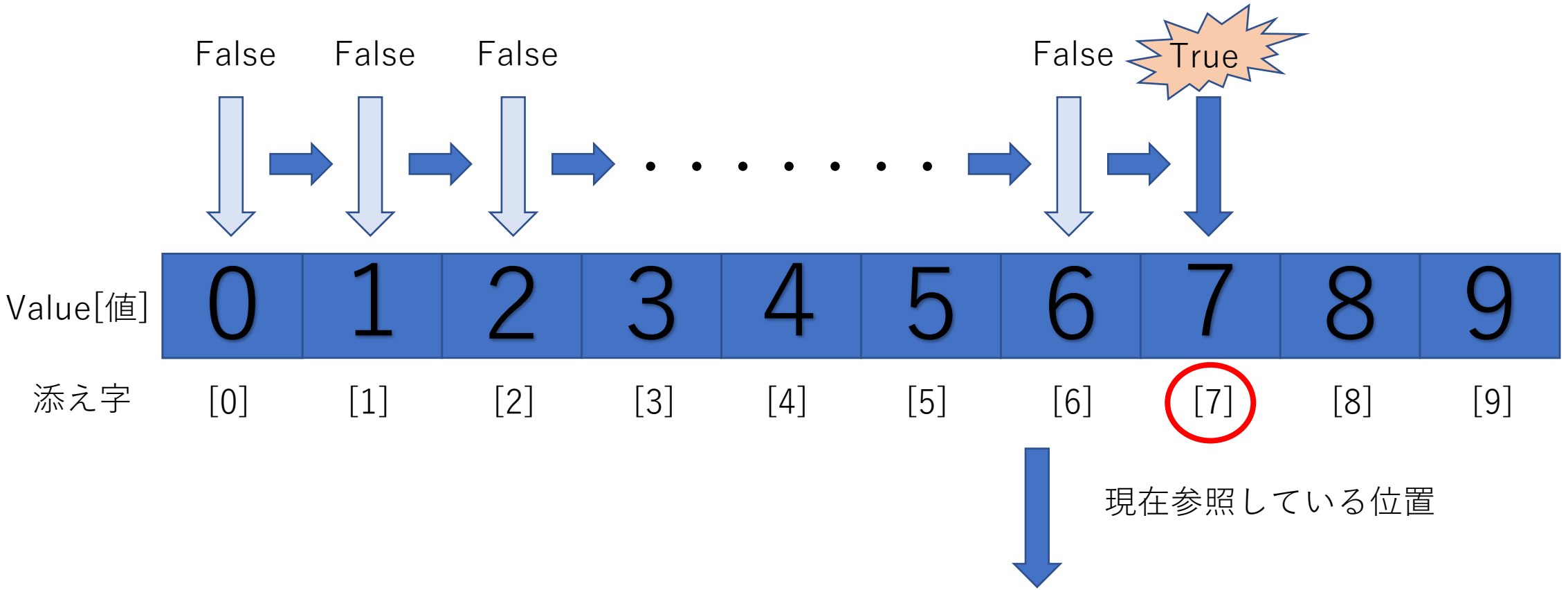
数字の7を探すプログラムを考える。

線形探索



数字の7を探すプログラムを考える。

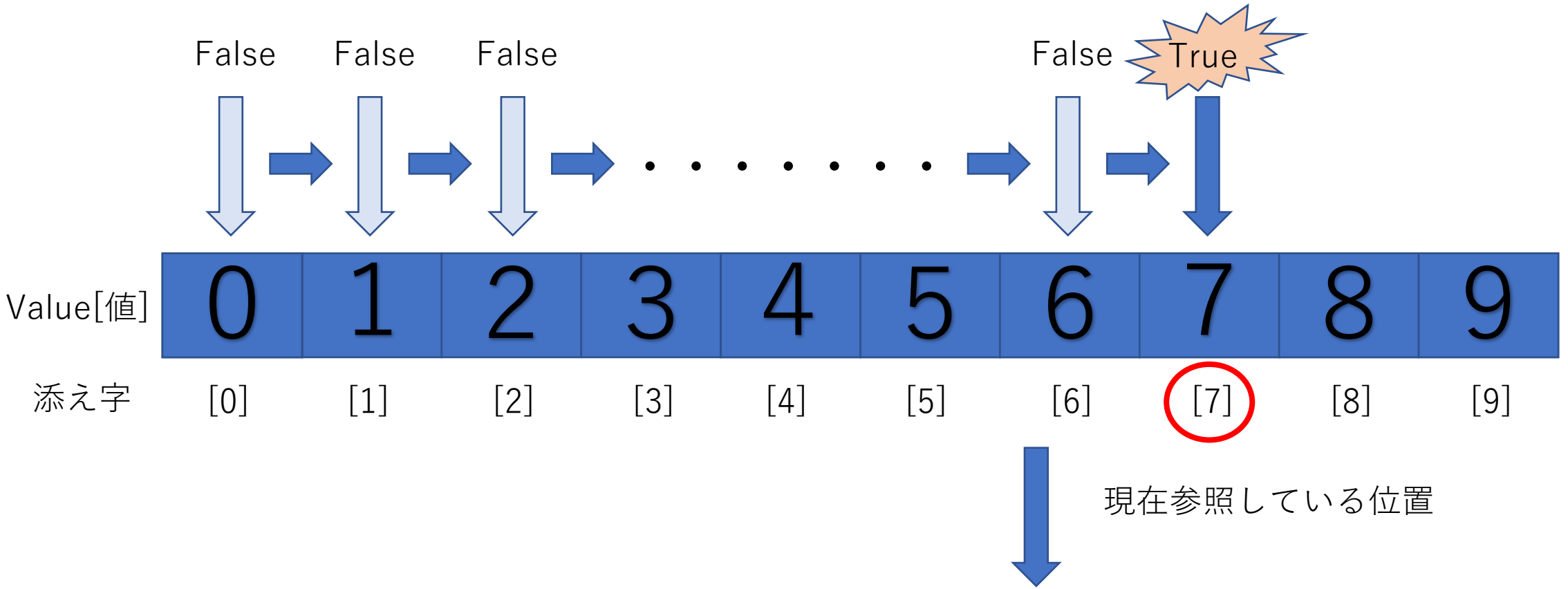
線形探索



数字の7を探すプログラムを考える。

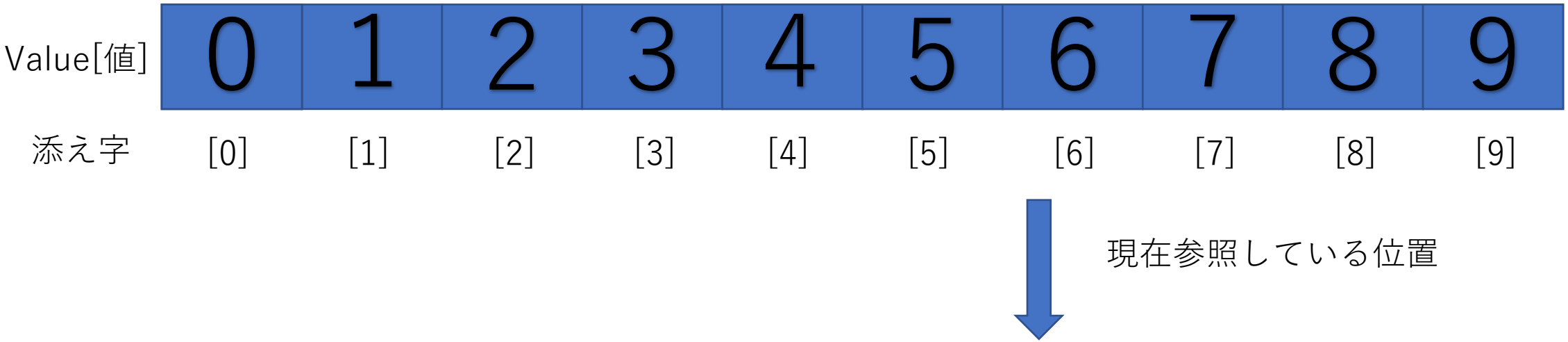
線形探索

比較回数(計算量)は8回



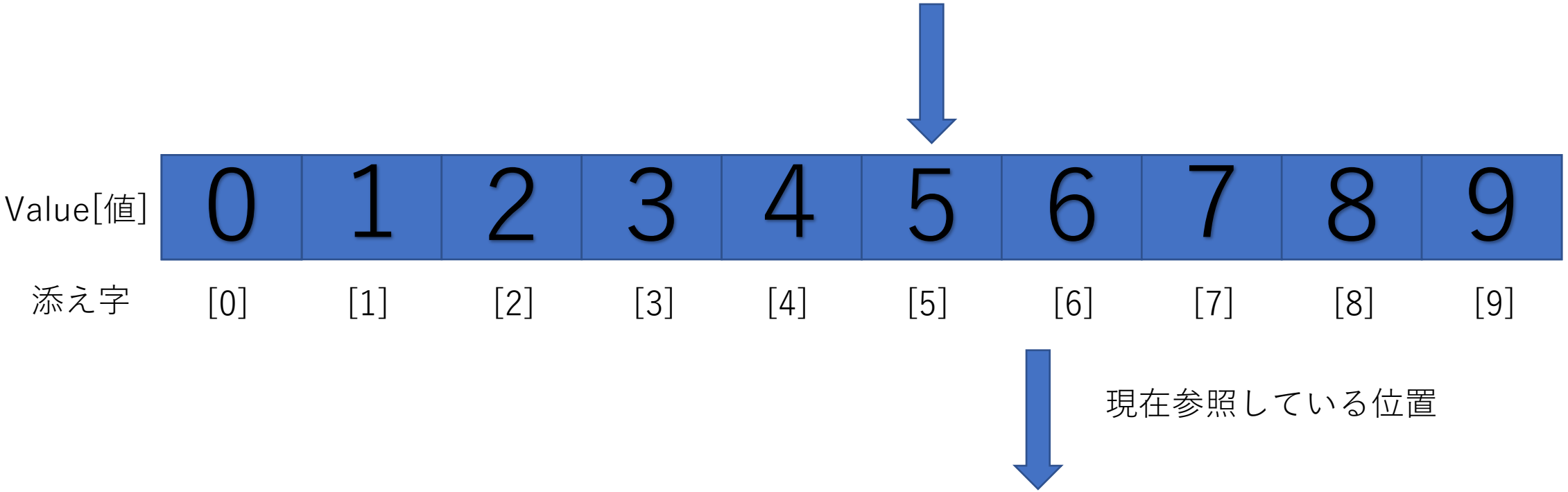
数字の7を探すプログラムを考える。

二分探索



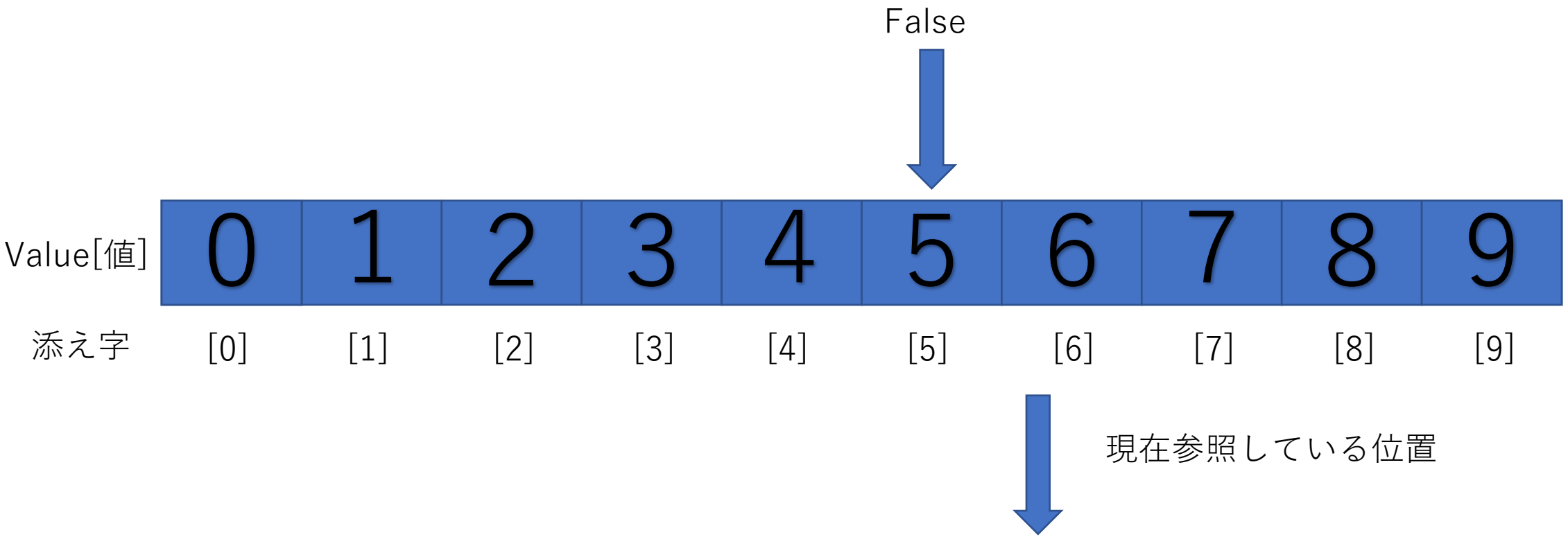
数字の7を探すプログラムを考える。

二分探索



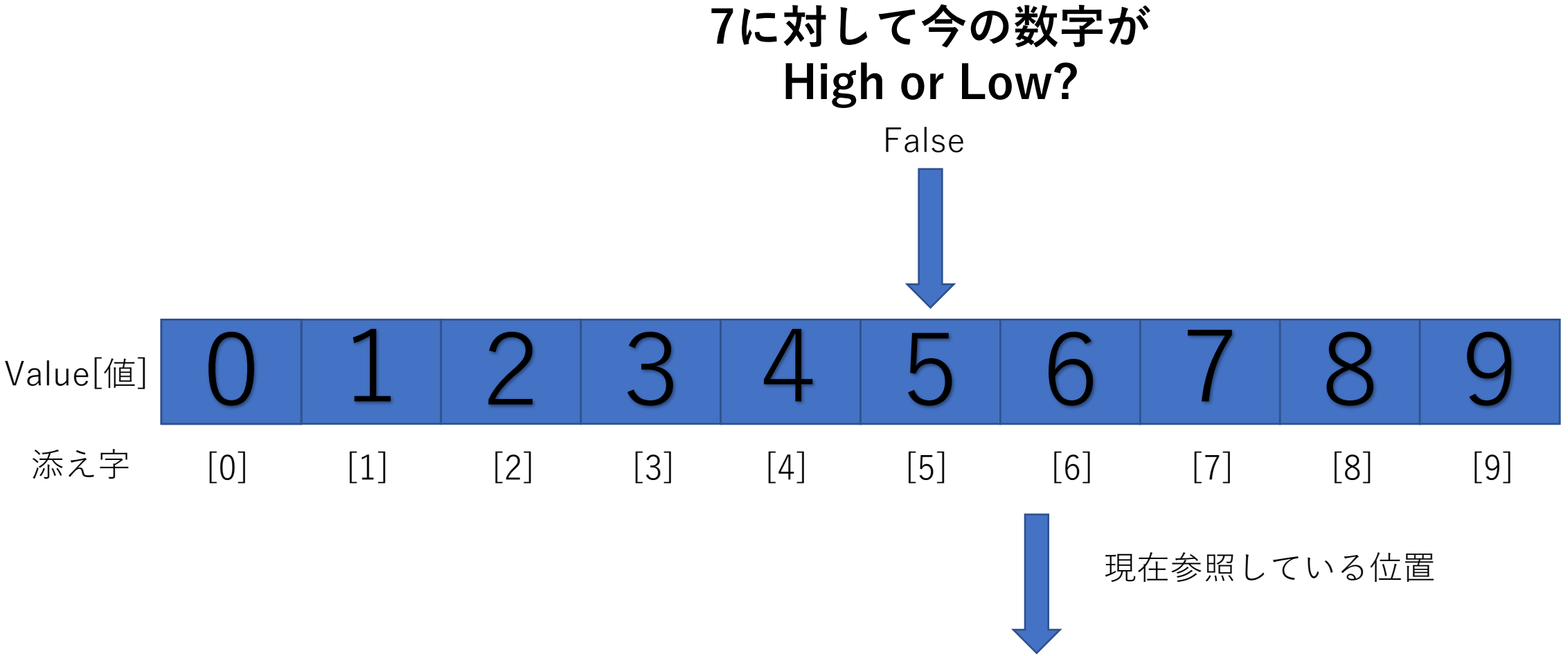
数字の7を探すプログラムを考える。

二分探索



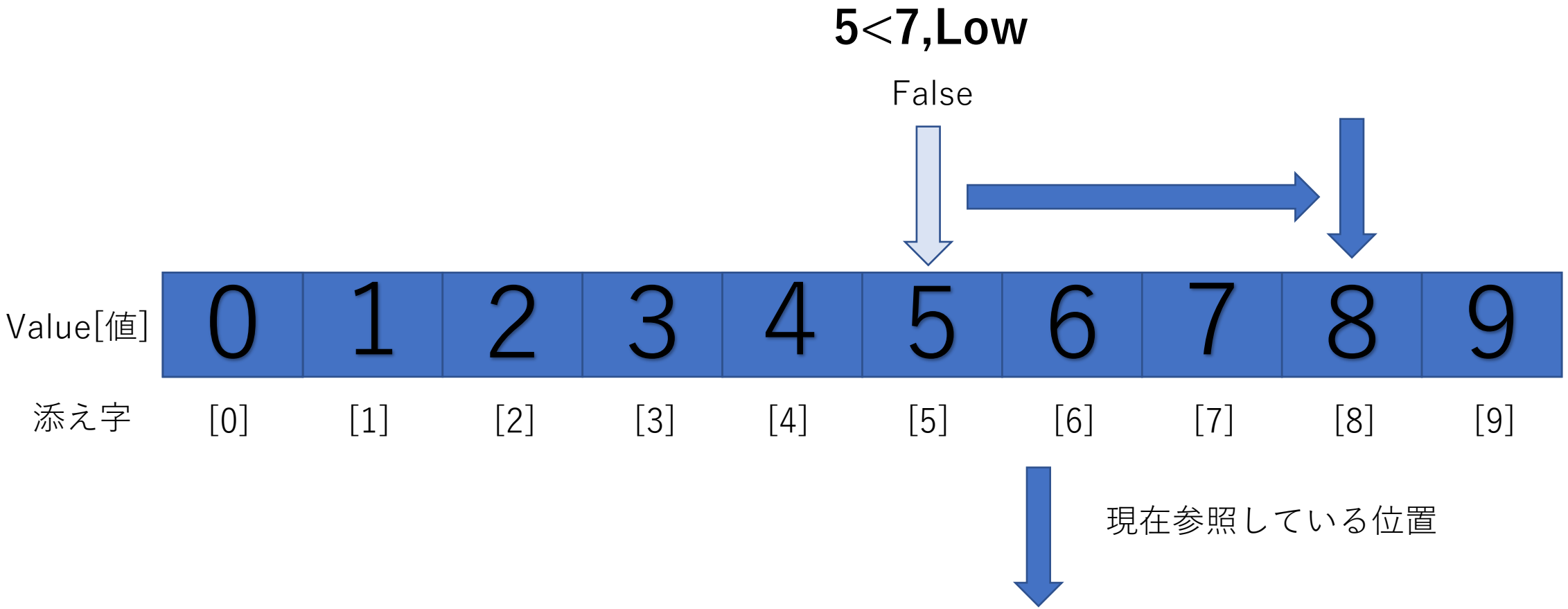
数字の7を探すプログラムを考える。

二分探索



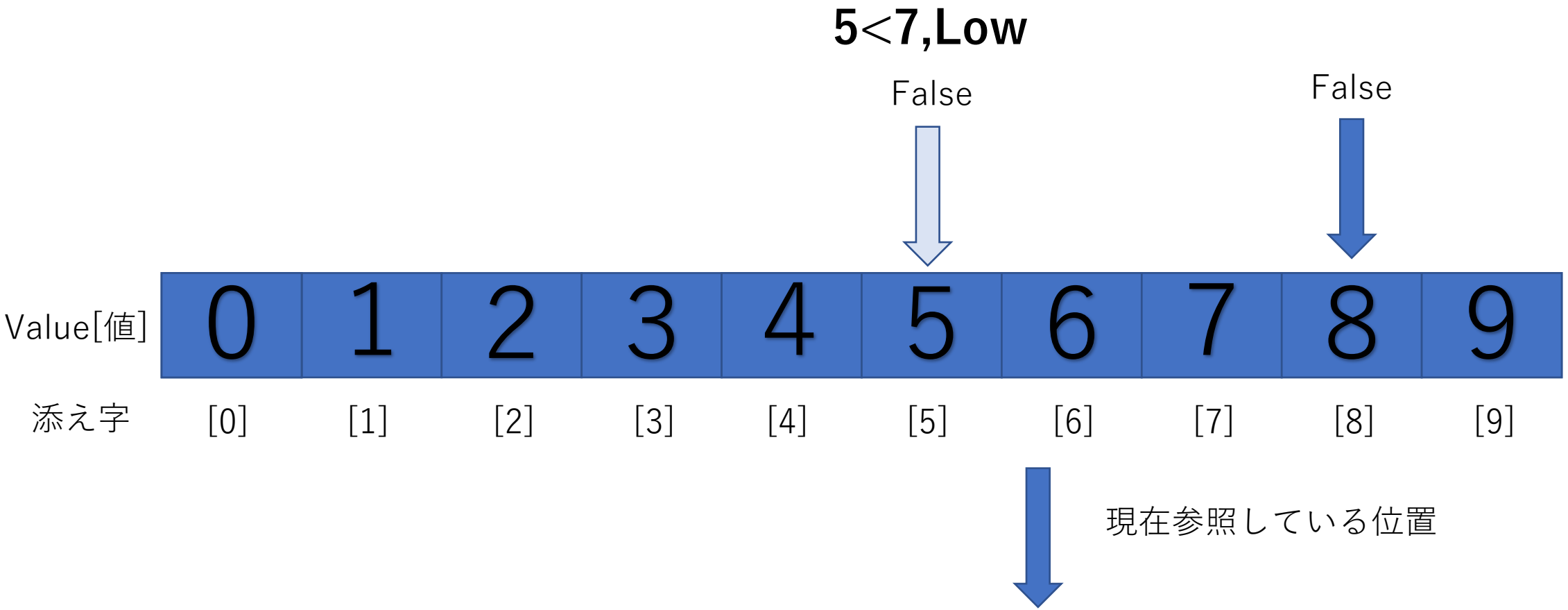
数字の7を探すプログラムを考える。

二分探索



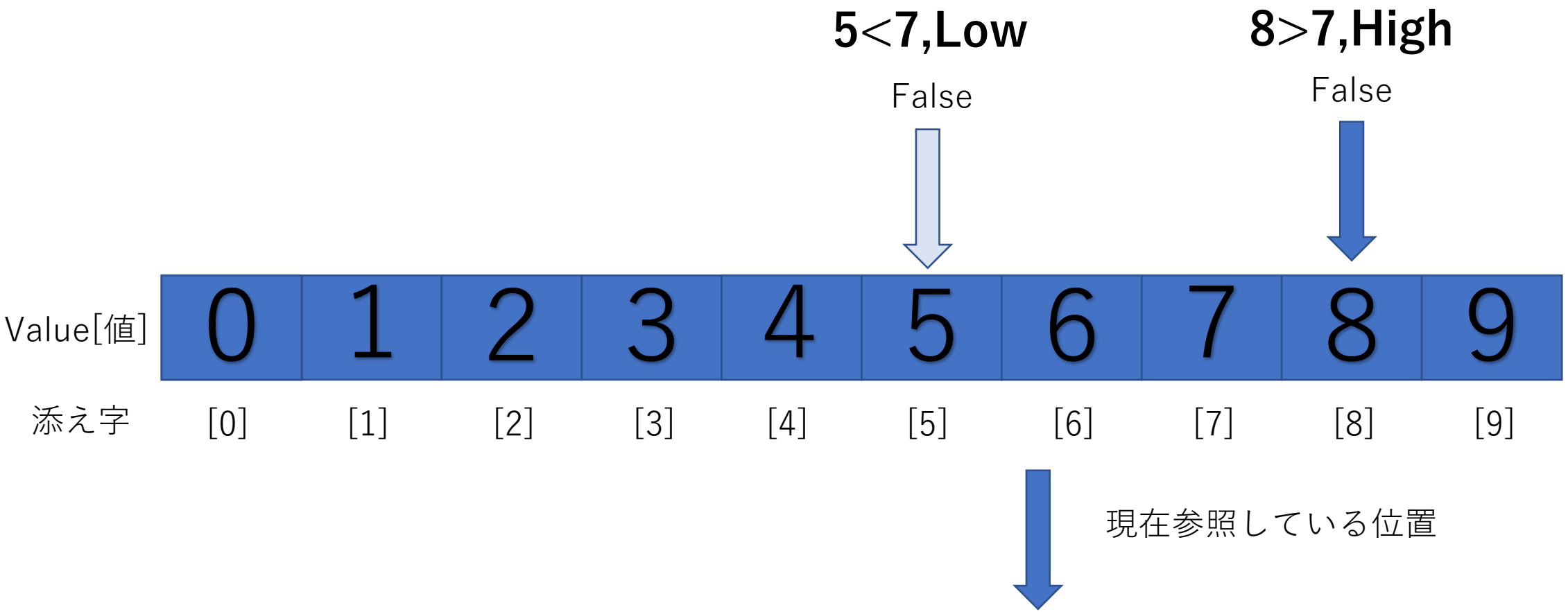
数字の7を探すプログラムを考える。

二分探索



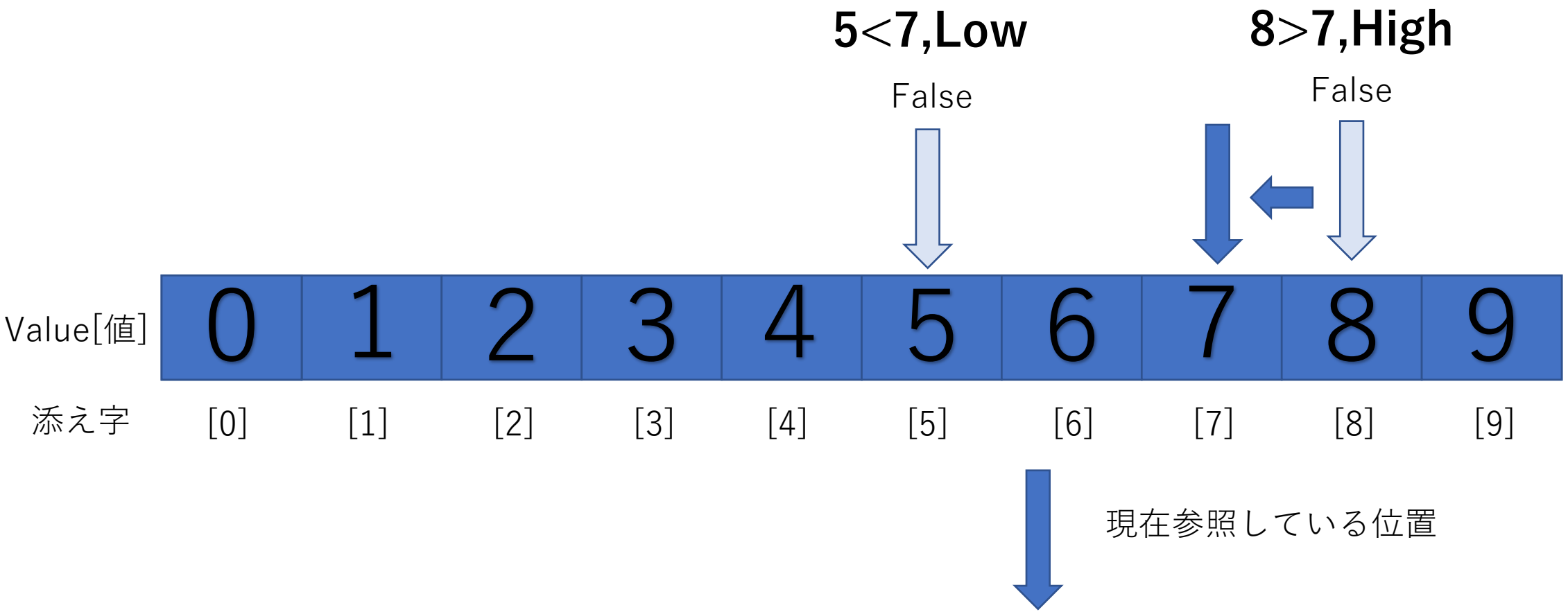
数字の7を探すプログラムを考える。

二分探索



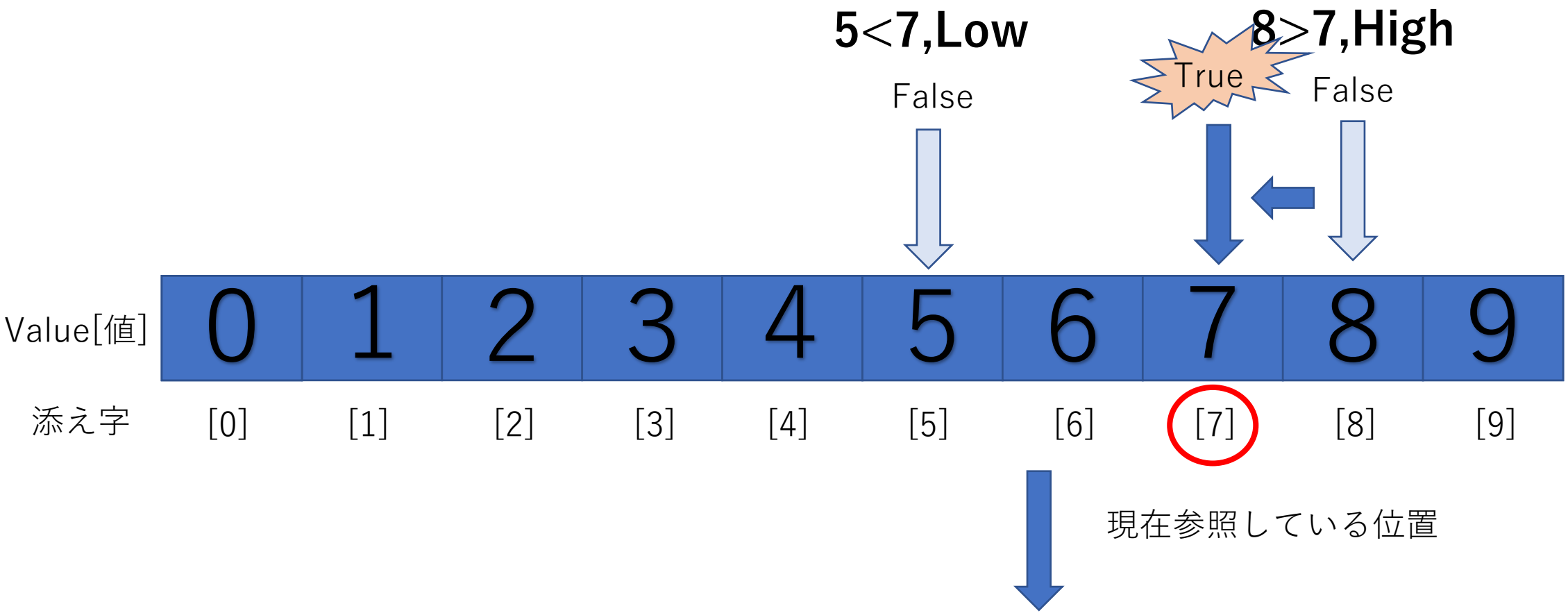
数字の7を探すプログラムを考える。

二分探索



数字の7を探すプログラムを考える。

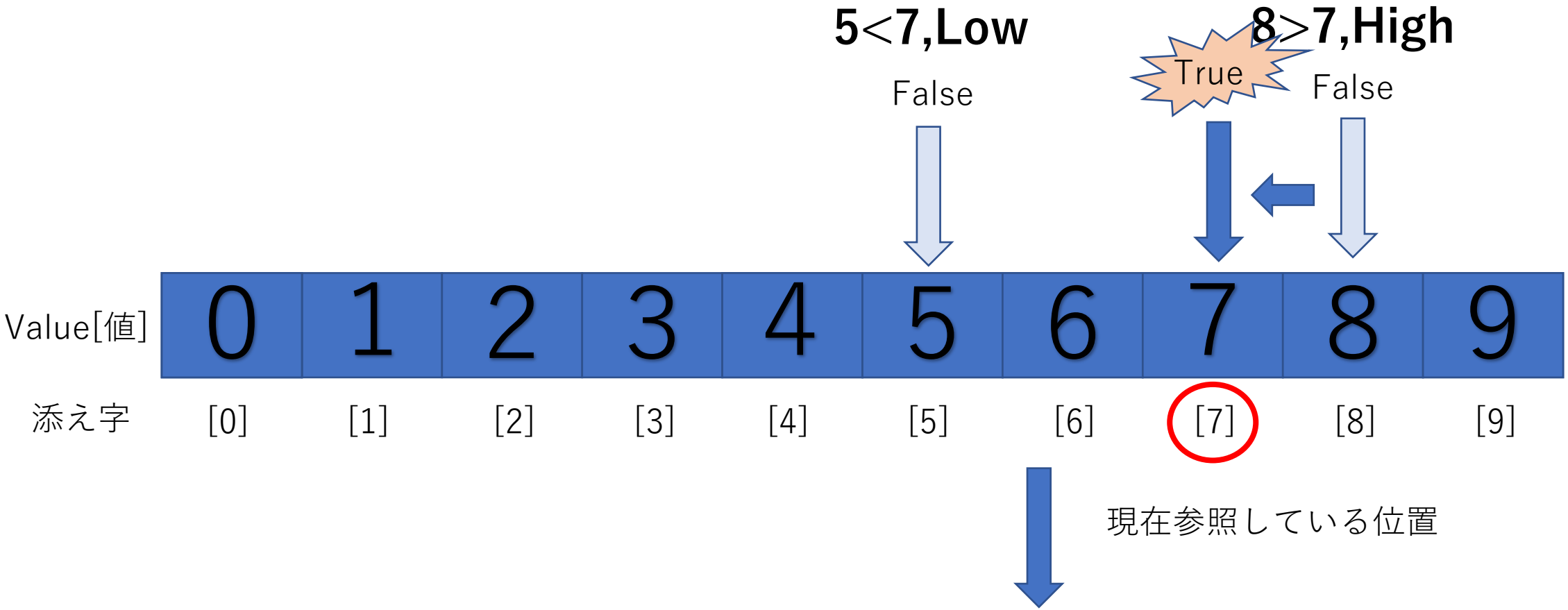
二分探索



数字の7を探すプログラムを考える。

二分探索

比較回数(計算量)は3回



前知識

とあるリストを整列するプログラム
を考える

Nはリストの要素数
バブルソート　最悪計算量　N^2
マージソート　最悪計算量　$N \log N$

計算量オーダーについて

・計算量について議論する上で便利なものさし
・入力サイズをNとしたとき

$O(N)$　はN　に比例した計算量
$O(N^2)$　はN^2　に比例した計算量
$O(\log N)$は$\log N$に比例した計算量

を示す。

計算量オーダーの考え方は、
前提：nはとても大きい数として考える。
例、計算量$3n^2 + 5n + 100$に対して

・最高次数の項以外を落とす
・係数を無視する

計算量オーダーの考え方は、
前提：nはとても大きい数として考える。
例、計算量$3n^2 + 5n + 100$に対して

・最高次数の項以外を落とす　$3n^2 + 5n + 100 \rightarrow 3n^2$
・係数を無視する

計算量オーダーの考え方は、
前提：nはとても大きい数として考える。
例、計算量$3n^2 + 5n + 100$に対して

・最高次数の項以外を落とす　$3n^2 + 5n + 100 \rightarrow 3n^2$
・係数を無視する　$3n^2 \rightarrow n^2$

したがって、

計算量$3n^2 + 5n + 100$のプログラムは

計算量オーダー$O(n^2)$のプログラム

として見なせる。

実際の関数の比較
(pythonでします)

制限時間1秒に対する計算の許容限界量

ループ回数	蟻本の記載	現在
10^6	余裕を持って間に合う	
10^7	おそらく間に合う	余裕を持って間に合う
10^8	非常にシンプルな処理でない限り厳しい	おそらく間に合う
10^9		非常にシンプルな処理でない限り厳しい

実際の計算量

$\log n$	n	$n \log n$	n^2	n^3	2^n	$n!$
2	**5**	12	25	130	30	120
3	**10**	33	100	1,000	1,024	3,628,800
4	**15**	59	225	3.375	32,768	-
4	**20**	86	400	8,000	1,048,576	-
5	**25**	116	625	15,625	-	-
5	**30**	147	900	27,000	-	-
7	**100**	664	10,000	1,000,000	-	-
8	**300**	2,468	90,000	27,000,000	-	-
10	**1,000**	9,966	1,000,000	-	-	-
13	**10,000**	132,877	100,000,000	-	-	-
16	**100,000**	1,660,964	-	-	-	-
20	**1,000,000**	19,931,568	-	-	-	-

参考文献

・計算量オーダーの求め方を総整理！ ～ どこから log が出て来るか ～

https://qiita.com/drken/items/872ebc3a2b5caaa4a0d0