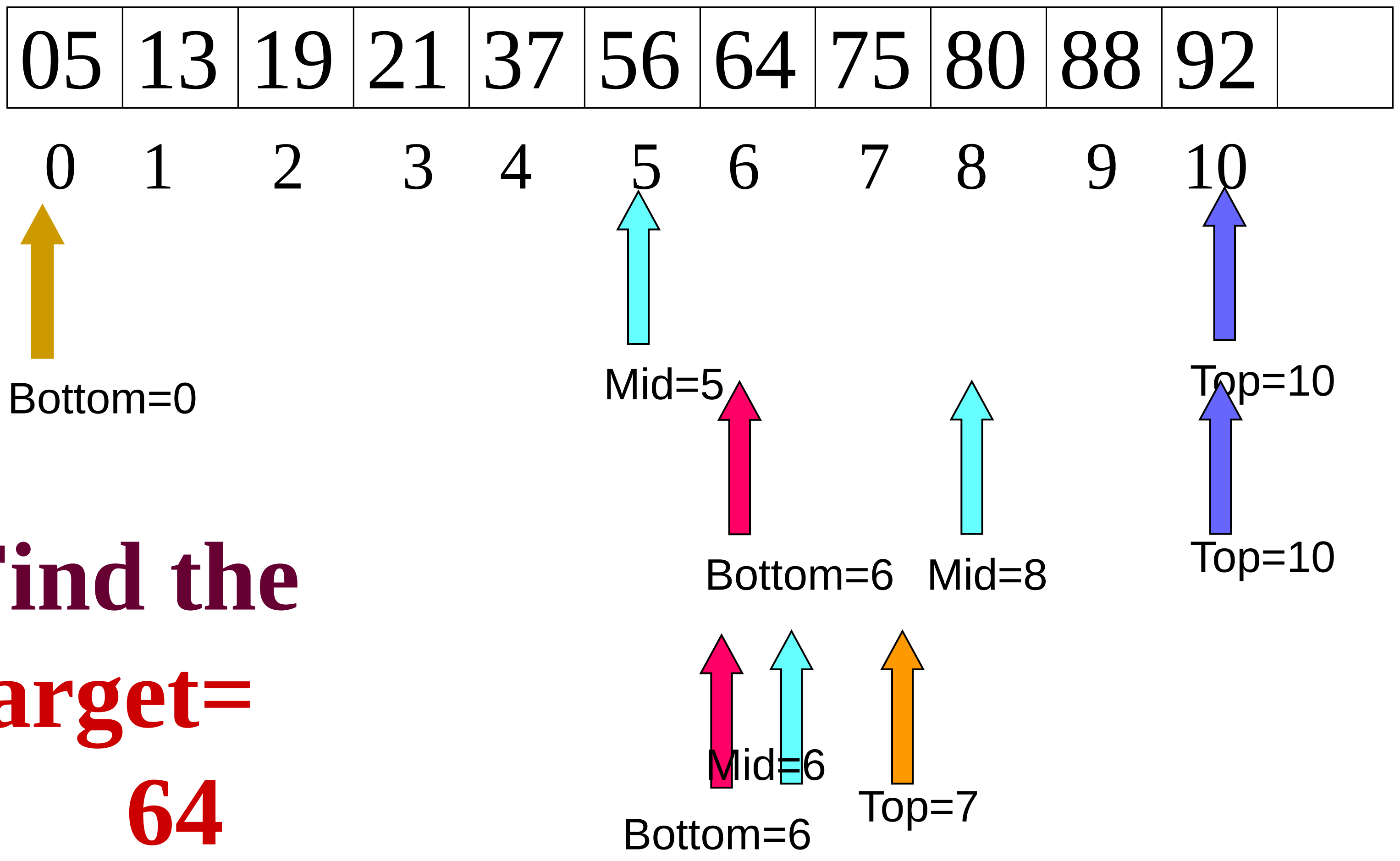
Recognizing Equality in Binary Search

- Method: Check at each stage to see if the target has been found.

```
    Error_code  recursive_binary_2(const Ordered_list &the_list,
     const Key &target, int bottom, int top, int &position)
     /* Pre: The indices bottom to top define the range in the list to
     search for the target .
     Post: If a Record in the range from bottom to top in the list has key
     equal to target , then position locates one such entry, and a code
     of success is returned. Otherwise,not_present is returned, and
     position is undefined.
     Uses: recursive_binary_2 , together with methods from the classes
     Ordered_list and Record . */
{
```



Find the

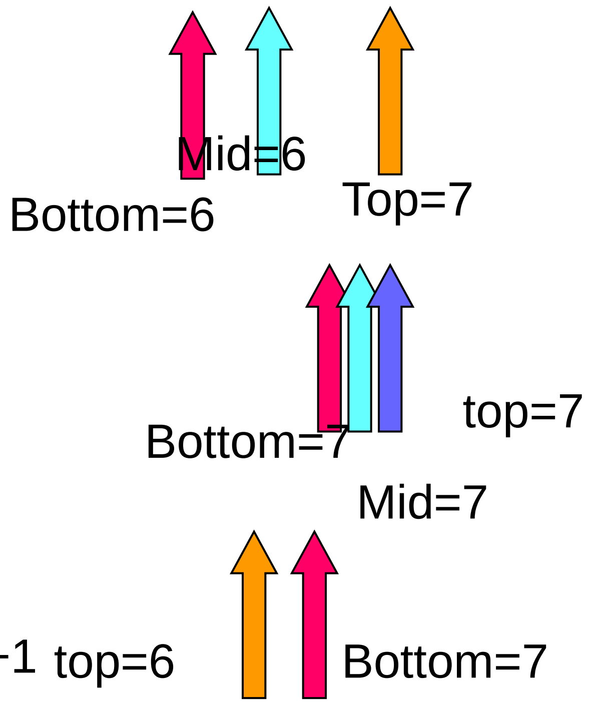
target=

64

如果 target=data
如果 target<data top=mid-1
如果 target>data bottom=mid+1

05	13	19	21	37	56	64	75	80	88	92	
0	1	2	3	4	5	6	7	8	9	10	

Find the target= 68



如果 target==data
如果 target<data top=mid-1
如果 target>data bottom=mid+1

Recognizing Equality in Binary Search

```
{
Record data;
if (bottom <= top) {
        int mid = (bottom + top)/2;
        the_list.retrieve(mid, data);
        if (data == target) {
                position = mid;
                return success;
                }
      else if (data < target)
                return recursive_binary_2(the_list, target, mid+1, top, position);
              else
                return recursive_binary_2(the_list, target, bottom, mid - 1, position);
     }
else return not_present;
}
```

0	1	1	1	4
0	1	2	3	4

寻找target=1的过程

Level0
（在run_recursive_binary_2中调用而进入）
recursive_binary_2(the_list,1,0,4,position)

Level1:　　（上一层与1比后进入该层）

执行至判断data==target成功后返回

```
    if (bottom <= top) {
    int mid = (bottom + top)/2;
    the_list.retrieve(mid, data);
    if (data == target) {
         position = mid;
         return success;
        }
  else if (data < target)
         return
         recursive_binary_2(the_list,
         target, mid+1, top, position);
        else
         return
         recursive_binary_2(the_list,
         target, bottom, mid - 1, position);
   }
else return not_present;
```

Nonrecursive Version
（非递归算法）

```
Error_code binary_search_2(const Ordered_list &the_list,const Key
&target, int &position)
/* Post: If a Record in the list has key equal to target , then position locates
one such entry and a code of success is returned. Otherwise,not present is
returned and position is undefined.
Uses: Methods for classes Ordered_list and Record . */
{
    Record data;
    int bottom = 0, top = the list.size( ) - 1;
    while (bottom <= top) {
        position = (bottom + top)/2;
        the_list.retrieve(position, data);
        if (data == target) return success;
        if (data < target) bottom = position + 1;
        else top = position - 1;
    }
return not_present; }
```