```
front \leftarrow front + 1
```

return true

end procedure

Implementation of dequeue() in C programming language -

Example

```
int dequeue() {
    if(isempty())
    return 0;
```

```
int data = queue[front];
front = front + 1;
```

return data;

}

For a complete Queue program in C programming language, please click here.

Data Structure and Algorithms Linear Search

Linear search is a very simple search algorithm. In this type of search, a sequential search is made over all items one by one. Every item is checked and if a match is found then that particular item is returned, otherwise the search continues till the end of the data collection. Linear Search Animation Algorithm Linear Search (Array A, Value x)

Step 1: Set i to 1

Step 2: if i > n then go to step 7

Step 3: if A[i] = x then go to step 6

Step 4: Set i to i + 1

Step 5: Go to Step 2

Step 6: Print Element x Found at index i and go to step 8

Step 7: Print element not found

Step 8: Exit

Pseudocode

procedure linear\_search (list, value)

for each item in the list if match item == value return the item's location end if end for

end procedure

To know about linear search implementation in C programming language, please click-here.

## Data Structure and Algorithms Binary Search

Binary search is a fast search algorithm with run-time complexity of  $O(\log n)$ . This search algorithm works on the principle of divide and conquer. For this algorithm to work properly, the data collection should be in the sorted form.

Binary search looks for a particular item by comparing the middle most item of the collection. If a match occurs, then the index of item is returned. If the middle item is greater than the item, then the item is searched in the sub-array to the left of the middle item. Otherwise, the item is searched for in the sub-array to the right of the middle item. This process continues on the sub-array as well until the size of the subarray reduces to zero.

### How Binary Search Works?

For a binary search to work, it is mandatory for the target array to be sorted. We shall learn the process of binary search with a pictorial example. The following is our sorted array and let us assume that we need to search the location of value 31 using binary search.

### Binary search

First, we shall determine half of the array by using this formula

mid = low + (high - low) / 2

Here it is, 0 + (9 - 0) / 2 = 4 (integer value of 4.5). So, 4 is the mid of the array.

Binary search

Now we compare the value stored at location 4, with the value being searched, i.e. 31. We find that the value at location 4 is 27, which is not a match. As the value is greater than 27 and we have a sorted array, so we also know that the target value must be in the upper portion of the array.

Binary search

We change our low to mid + 1 and find the new mid value again.

low = mid + 1

mid = low + (high - low) / 2

Our new mid is 7 now. We compare the value stored at location 7 with our target value 31.

#### Binary search

The value stored at location 7 is not a match, rather it is more than what we are looking for. So, the value must be in the lower part from this location.

Binary search

Hence, we calculate the mid again. This time it is 5.

Binary search

We compare the value stored at location 5 with our target value. We find that it is a match.

Binary search

We conclude that the target value 31 is stored at location 5.

Binary search halves the searchable items and thus reduces the count of comparisons to be made to very less numbers.

Pseudocode

The pseudocode of binary search algorithms should look like this –

Procedure binary\_search

 $A \leftarrow sorted array$ 

 $n \gets size \ of \ array$ 

 $x \leftarrow$  value to be searched

Set lowerBound = 1 Set upperBound = n

while x not found if upperBound < lowerBound EXIT: x does not exists.

```
set midPoint = lowerBound + ( upperBound - lowerBound ) / 2
```

```
if A[midPoint] < x
set lowerBound = midPoint + 1
```

```
if A[midPoint] > x
set upperBound = midPoint - 1
```

```
if A[midPoint] = x
```

EXIT: x found at location midPoint

end while

# end procedure

To know about binary search implementation using array in C programming language, please click here.

# Data Structure - Interpolation Search

Interpolation search is an improved variant of binary search. This search algorithm works on the probing position of the required value. For this algorithm to work properly, the data collection should be in a sorted form and equally distributed.