**Data Structure**

**&**

**Algorithm**

**Class IX**

**Lab 10**

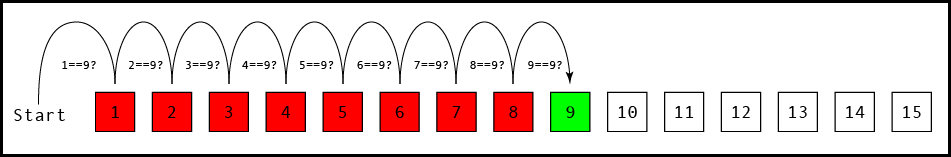
|  |
| --- |
| Lab Objectives:Linear SearchBinary Search |

# 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.

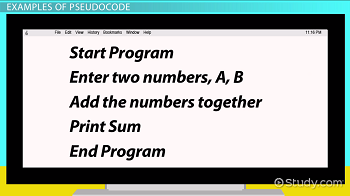
## Every item is checked and if a match is found then that particular item is returned, otherwise the search continues till the end of the data collection.

## Suppose you need to find 9 among the numbers 1 to 15:



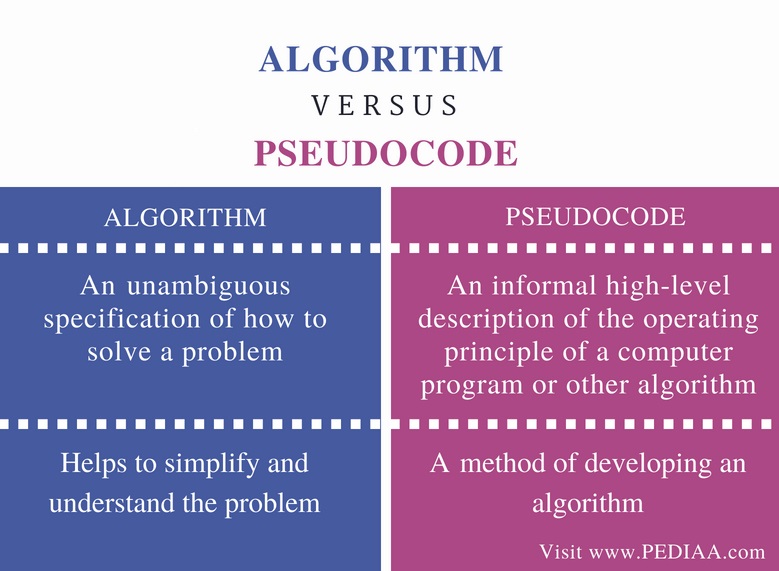
# Pseudocode

## Pseudocode is an informal high-level description of the operating principle of a computer program or other algorithm.

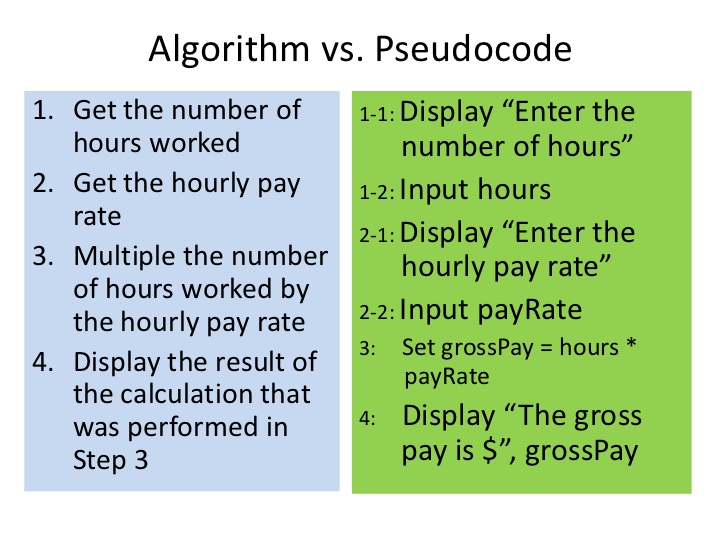


## It uses the structural conventions of a normal programming language, but is intended for human reading rather than machine reading.

# Algorithm vs Pseudocode

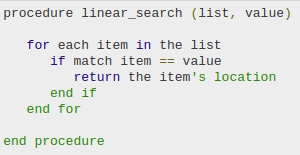


# Writing style of Algorithm and Pseudocode

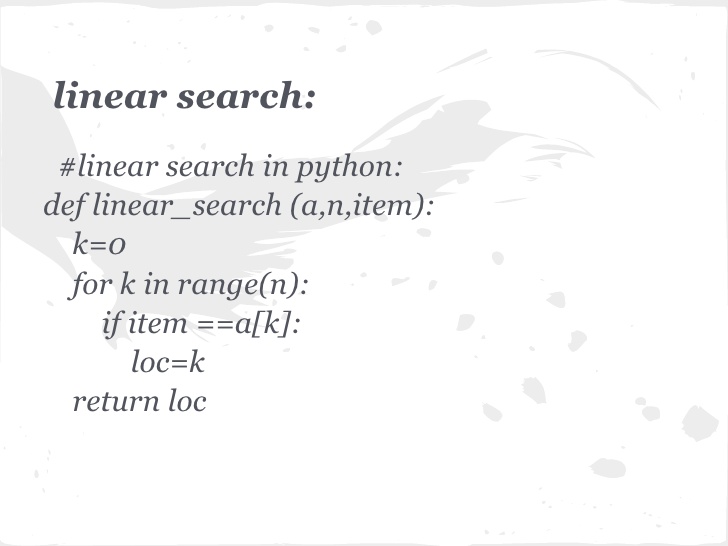


# 

# Pseudocode for Linear Search



# Algorithm for Linear Search

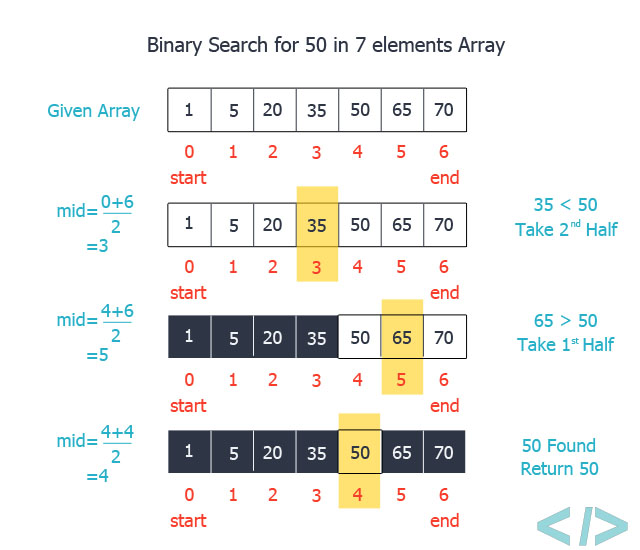


# Binary Search

## Binary search is an efficient algorithm for finding an item from a sorted list of items.

## 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.



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

mid = low + (high - low) / 2

## IMG_256Here it is, 0 + (9 - 0 ) / 2 = 4 (integer value of 4.5). So, 4 is the mid of the array.

## 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.

## IMG_256

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

low = mid + 1

mid = low + (high - low) / 2

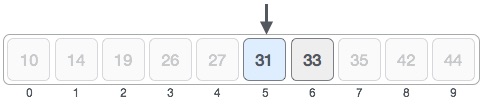
## Our new mid is 7 now. We compare the value stored at location 7 with our target value 31.

## IMG_256

## 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.



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



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



## 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.