**Linear search using python:**

What is linear? - progressing from one stage to another in a single series of steps; sequential.

Write a code to search an element from a list.

Let’s imagine a list and 5,8,4,6,9,2 and if you want to search 9

In python you can do it with certain function to search for it, what if you want to do it by yourself i.e,. manually by using loop. What this loop will do is it will start comparing with first element for 9 and if it doesn’t match it will go to the second number and so on until it finds the number 9 with the index value of 4.

Diagram

Description automatically generated

In order to search for 9 let’s use the below code with while loop in the function. Value is there in the list and the value is found.

Graphical user interface, application, Teams

Description automatically generated

What if we give a new value which is not in the list, update the n value and when we run the code it will print ***‘not found’*** . Run the code and check the output.

* Now we are saying that we found the value, but we are not saying where it is ? we also have to maintain the position.

Graphical user interface, text, application, Word

Description automatically generated

Graphical user interface, application

Description automatically generated

* Now if we want to know the position as per our counting which is the human way, we can add pos+1 in the print statement***(“found at “, pos +1)*** and observe the output.

That’s how we do the liner search, ***and you can modify the code from while loop to for loop.***

**Binary Search in Python:**

Let’s perform another type of search which is binary. Why do we need to know about binary search – Let’s say you want to perform a search on 1000 values to find a single value which sits after 800/900 and if we want to search it using linear search which is using a loop we need to run the loop 800/900 times depending on the position. In binary values all the values needs to be ***sorted*** which is NOT the case with linear search.

Graphical user interface, application, Teams

Description automatically generated

Graphical user interface, application

Description automatically generated

Application

Description automatically generated with medium confidence

Graphical user interface, application

Description automatically generated

Now the value is smaller or bigger than the mid value. Here in our case we are searching for 45 which means 8 is smaller than the 45 hence we will update the lower bound value to

Diagram, text

Description automatically generated

Teams

Description automatically generated with low confidence

Graphical user interface, application

Description automatically generated

Graphical user interface, application

Description automatically generated

Graphical user interface, application

Description automatically generated

Is it a good way of doing the search like this – Yes, it is we are skipping the values in the initial level, if the list is big it will skip lot of values.

Let’s do it with code.

L = 0 #lower bound zero

U = len(list) – 1 # upper bound is 1 less than the length

Apply a loop : lower bound should always be <= upper bound and find the mid value as shown below. Lower+upper//2 # // will give integer division

Graphical user interface, website

Description automatically generated

Check for another value

Graphical user interface, website

Description automatically generated

For the value which is not present.

Graphical user interface, website

Description automatically generated

def search(list, n):

    i = 0

    while i < len(list):

        if list[i] == n:

            return True

        i += 1

    return False

list = [5,8,4,6,9,2]

n = 10

if search(list, n):

    print("found")

else:

    print("not found")

pos = -1

def search(list, n):

    i = 0

    while i < len(list):

        if list[i] == n:

            globals()["pos"] = i

            return True

        i += 1

    return False

list = [5,8,4,6,9,2]

n = 9

if search(list, n):

    print("found at ", pos+1)

else:

    print("not found")

pos = -1

def search(list, n):

    l = 0

    u = len(list) - 1

    while l <= u:

        mid = (l+u) // 2

        if list[mid] == n:

            globals()["pos"] = mid

            return True

        else:

            if list[mid] < n:

                l = mid+1

            else:

                u = mid-1

list = [4,7,8,12,45,99]

n = 99

if search(list, n):

    print("found at ", pos+1)

else:

    print("not found")