NIGER DELTA UNIVERSITY

WILBERFORCE ISLAND,
BAYELSA STATE.

NAME:	GODSPOWER CELESTINE OGHENEMINE
MATRIC NUMBER:	UG/17/1437
DEPARTMENT:	COMPUTER SCIENCES
FACULTY:	SCIENCES
COURSE CODE:	CMP 421
COURSE TITLE:	ALGORITHM

*Implement a binary search and a sequential search with any programming language?

Python Program for Binary Search

1. Compare x with the middle element.
2. If x matches with the middle element, we return the mid index.
3. Else If x is greater than the mid element, then x can only lie in right half subarray after the mid element. So we recur for the right half.
4. Else (x is smaller) recur for the left half.

```
# Python 3 program for recursive binary search.
# Modifications needed for the older Python 2 are found in comments.

# Returns index of x in arr if present, else -1
def binary_search(arr, low, high, x):

    # Check base case
    if high >= low:

        mid = (high + low) // 2

        # If element is present at the middle itself
        if arr[mid] == x:
            return mid

        # If element is smaller than mid, then it can only
        # be present in left subarray
        elif arr[mid] > x:
            return binary_search(arr, low, mid - 1, x)

        # Else the element can only be present in right subarray
        else:
            return binary_search(arr, mid + 1, high, x)

    else:
        # Element is not present in the array
        return -1

# Test array
arr = [ 2, 3, 4, 10, 40 ]
x = 10

# Function call
```

```
result = binary_search(arr, 0, len(arr)-1, x)

if result != -1:
    print("Element is present at index", str(result))
else:
    print("Element is not present in array")
```

Output:
Element is present at index 3
Iterative:

Python3

```
# Iterative Binary Search Function
# It returns index of x in given array arr if present,
# else returns -1
def binary_search(arr, x):
    low = 0
    high = len(arr) - 1
    mid = 0

    while low <= high:

        mid = (high + low) // 2

        # If x is greater, ignore left half
        if arr[mid] < x:
            low = mid + 1

        # If x is smaller, ignore right half
        elif arr[mid] > x:
            high = mid - 1

        # means x is present at mid
        else:
            return mid

    # If we reach here, then the element was not present
    return -1


# Test array
arr = [ 2, 3, 4, 10, 40 ]
x = 10

# Function call
result = binary_search(arr, x)
```

```
  if result != -1:
     print("Element is present at index", str(result))
  else:
     print("Element is not present in array")
```

Output:

Element is present at index 3

```
Python search program
LINEAR OR SEQUENTIAL SEARCH


lst = []
num = int(input("Enter size of list: \t"))
for n in range(num):
 numbers = int(input("Enter any number: \t"))
 lst.append(numbers)

x = int(input("\nEnter number to search: \t"))

found = False

for i in range(len(lst)):
 if lst[i] == x:
 found = True
 print("\n%d found at position %d" % (x, i))
 break
if not found:
 print("\n%d is not in list" % x)
```