**PROGRAM 1**

**AIM – Write an algorithm and program to implement Linear Search.**

**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

**i) Using arrays**

**SOURCE CODE:-**

#include <bits/stdc++.h>

using namespace std;

int search(intarr[], int n, int x)

{

inti;

for (i = 0; i< n; i++)

if (arr[i] == x)

returni;

return -1;

}

int main()

{

intarr[] = { 3, 4, 1, 7, 5 };

int n = sizeof(arr) / sizeof(arr[0]);

int x = 4;

int index = search(arr, n, x);

if (index == -1)

cout<< "Element is not present in the array";

else

cout<< "Element found at position " << index;

return 0;

}

**(ii) Using Linked List**

**Source Code -**

#include <bits/stdc++.h>

using namespace std;

class Node

{

public:

int key;

Node\* next;

};

void push(Node\*\* head\_ref, intnew\_key)

{ Node\* new\_node = new Node();

new\_node->key = new\_key;

new\_node->next = (\*head\_ref);

(\*head\_ref) = new\_node;

}

bool search(Node\* head, int x)

{

Node\* current = head; // Initialize current

while (current != NULL)

{

if (current->key == x)

return true;

current = current->next;

}

return false;

}

int main()

{

Node\* head = NULL;

int x = 21;

push(&head, 10);

push(&head, 30);

push(&head, 11);

push(&head, 21);

push(&head, 14);

search(head, 21)? cout<<"Yes" : cout<<"No";

return 0;

}