ASSIGNMENT NO:13

# TITLE: LINEAR SEARCH AND SENTINAL SEARCH

# PROBLEM DEFINITION:

Write C++ program to store roll numbers of student in array who attended training program in random order. Write function for searching whether particular student attended training program or not using linear search and sentinel search.

# PREREQUISITES:

Stored array if random numbers we enter sort pt by any method.

# LEARNING OBJECTIVES:

Sorting and searching array method.

# THEORY:

**Sequential \linear search:**

* Linear search is usually very simple to implement and is practical when the list has only a few elements, or when performing a single search on unordered list.
* When many values have to be searched in some list in order to use a faster method.
* Even though theory other search algorithm may be faster than linear search, in practice even any medium sized array might be infeasible to use anything else.

**Algorithm:**

Steps:

1. Start
2. Let N be the size of list. Let target the element searched be the variable.
3. Read elements from user and store array A[50]
4. Let flag=0, position=0.

For (i = 0to n)

begin

if (key==A[i])

flag=1;

break;

else

position++

end

if(flag==1)

cout<<”element found”<<A[position];

else

cout<<”element not found”;

1. Stop.

Eg:

6 11 32 1 10 43 21

Key=1

Flag=0

Pos=0

1. for (i=0)

Key!=6

Pos=1

1. for (i=1)

key!= 11

pos = 1

1. for (i=2)

key!=32

pos =3

1. for (i=3)

key =1

flag1

break;

display

element found :1

**Advantages of linear search:**

1. The primary advantage is its simplicity, conceptually easy , and easy to understand and implement and also very forward..

2. From an operational standpoint, linear search also very resource efficient as it does not copying partitioning the array search and thus memory efficient.

3. It operates equally well on both sorted and unsorted data.

**Disadvantages of linear search:**

1. It has very poor O (n) general efficiency.
2. The performance of algorithm scales linearly with the size of the input.
3. For general case, linear search is considered slower than many other search methods.

**Sentinel search:**

1. The simpler form search is sentinel search.
2. This search is applicable to a table either as an array or as a linked list.
3. Best code requires only one comparison so O (1). Worst case requires n comparisons so O (n). Average case requires (n=1)/2 comparisons so again O(n)

**Algorithm:**

1. Start.
2. Read input data from user and store in array eg.A[50]
3. Let n be the size of the list.
4. Let target element to be searched be ‘key’ variable. Let pos=0 and flag=0.
5. For(i=0 to n)

begin

if (key==A[i])

flag =1

cout<<”element found”

else if (flag==0)

cout<<”element not found”

End

1. Stop.

Eg:

6 11 15 10 3 21 37

Key=10

For i=0

6! =key

For i=1

11! =key

For i=2

15! =key

For i=3

Key=10

Element found: 10

**Advantages of sentinel search:**

1. It is simpler searching method, implementation wise and conceptually.
2. It is memory efficient as it does not require partitioning.
3. Work well both on sorted and unsorted arrays and linked list.

**Disadvantages of sentinel search:**

1. Requires too many passes hence general efficiency is low.
2. More time complexity than other sorting algorithm.

**Application:**

1. For sorting combination problems.
2. Construction of paging algorithm for many storage hierarchies.