**Day 4 Assignment**

Q1. the Binary Search algorithm, it is suggested to calculate the mid as

beg + (end - beg) / 2 instead of (beg + end) / 2. Why is it so?

It is used for large set of values where we can find mid very easily.

Q2. Write the algorithm/function for Ternary Search.

#include <stdio.h>

int ternarySearch(int l, int r, int key, int arr[])

{

if (r >= l) {

// Find the mid1 and mid2

int mid1 = l + (r - l) / 3;

int mid2 = r - (r - l) / 3;

// Check if key is present at any mid

if (arr[mid1] == key) {

return mid1;

}

if (arr[mid2] == key) {

return mid2;

}

// Since key is not present at mid check in which region it is present then repeat the Search operation in that region

if (key < ar[mid1]) {

// The key lies in between l and mid1

return ternarySearch(l, mid1 - 1, key, arr);

}

else if (key > ar[mid2]) {

// The key lies in between mid2 and r

return ternarySearch(mid2 + 1, r, key, arr);

}

else {

// The key lies in between mid1 and mid2

return ternarySearch(mid1 + 1, mid2 - 1, key, ar);

}

}

// Key not found

return -1;

}

// Driver code

int main()

{

int l, r, p, key;

// Get the array

// Sort the array if not sorted

int arr[] = { 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 };

// Starting index

l = 0;

// length of array

r = 9;

// Checking for 5

// Key to be searched in the array

key = 5;

// Search the key using ternarySearch

p = ternarySearch(l, r, key, arr);

// Print the result

printf("Index of %d is %d\n", key, p);

// Checking for 50

// Key to be searched in the array

key = 50;

// Search the key using ternarySearch

p = ternarySearch(l, r, key, arr);

// Print the result

printf("Index of %d is %d", key, p);

}