1. Implement Linear Search

#include <stdio.h>

int linear\_search(int arr[], int n, int target) {

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

if (arr[i] == target) {

return i;

}

}

return -1;

}

int main() {

int my\_array[] = {5, 2, 9, 1, 7, 6};

int array\_size = sizeof(my\_array) / sizeof(my\_array[0]);

int target\_element = 7;

int result = linear\_search(my\_array, array\_size, target\_element);

if (result != -1) {

printf("Element %d found at index %d.\n", target\_element, result);

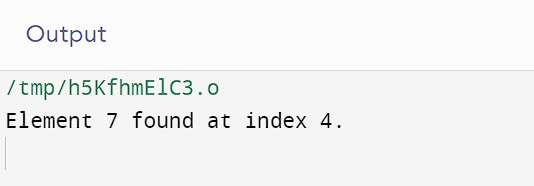
} else {

printf("Element %d not found in the array.\n", target\_element);

}

return 0;

}



1. Implement Binary Search

#include <stdio.h>

int binary\_search(int arr[], int n, int target) {

int low = 0, high = n - 1;

while (low <= high) {

int mid = (low + high) / 2;

if (arr[mid] == target) {

return mid;

} else if (arr[mid] < target) {

low = mid + 1;

} else {

high = mid - 1;

}

}

return -1;

}

int main() {

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

int array\_size = sizeof(sorted\_array) / sizeof(sorted\_array[0]);

int target\_element = 7;

int result = binary\_search(sorted\_array, array\_size, target\_element);

if (result != -1) {

printf("Element %d found at index %d\n", target\_element, result);

} else {

printf("Element %d not found in the array\n", target\_element);

}

return 0;

}

