**BINARY SEARCH**

#include <stdio.h>

int binarySearch(int arr[], int size, int target) {

int left = 0, right = size - 1;

while (left <= right) {

int mid = left + (right - left) / 2;

if (arr[mid] == target)

return mid;

else if (arr[mid] < target)

left = mid + 1;

else

right = mid - 1;

}

return -1;

}

int main() {

int arr[] = {3, 5, 9, 11, 13, 17 };

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

int target = 9;

int result = binarySearch(arr, size, target);

if (result != -1)

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

else

printf("Element not found\n");

return 0;

}

A computer screen with a black screen

AI-generated content may be incorrect.

**QUICK SORT**

#include <stdio.h>

void quickSort(int arr[], int left, int right) {

if (left >= right) return;

int pivot = arr[right];

int i = left;

int j;

for (j = left; j < right; j++) {

if (arr[j] < pivot) {

int temp = arr[i];

arr[i] = arr[j];

arr[j] = temp;

i++;

}

}

int temp = arr[i];

arr[i] = arr[right];

arr[right] = temp;

quickSort(arr, left, i - 1);

quickSort(arr, i + 1, right);

}

int main() {

int arr[] = {7, 5, 9, 4, 10};

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

int i;

quickSort(arr, 0, n - 1);

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

printf("%d ", arr[i]);

printf("\n");

return 0;

}

