COS10007

Week 8 Prac

Question 1

Linear search checks each item in a list one by one to see if it matches,

it doesn't require a sorted list.

Something like this:

for(int i = 0; i < array\_len; i++) {

if(array[i] == searchterm) {

// Do something.

}

}

Binary search checks the midpoint of a sorted list and jumps back or forward by

half the remainder to find the search term.

Something like this:

bool search(int s) {

divide(int left, int right) {

if(array[midpoint] == searchterm) {return true;}

if(left == right) {return false;}

int midpoint = (left + right) / 2;

if(array[midpoint] > searchterm) {

return divide(midpoint, right);

} else {

return divide(left, midpoint);

}

}

return divide(0, array\_len - 1);

}

// Returns bool result.

search(searchterm);

Question 2

#include <stdio.h>

#include <time.h>

#include <stdlib.h>

#include <string.h>

#define ARRAY\_LEN 10

#define MAX\_VAL 64

// Array to search.

int array[ARRAY\_LEN];

void sort\_array(void) {

for(int i = 0; i < ARRAY\_LEN; i++) {

int min = MAX\_VAL;

size\_t min\_index = 0;

for(int j = i; j < ARRAY\_LEN; j++) {

if(array[j] < min) {

min = array[j];

min\_index = j;

}

}

int temp = array[i];

array[i] = min;

array[min\_index] = temp;

}

}

void print\_array(void) {

printf("?");

for(int i = 0; i < ARRAY\_LEN; i++) {

printf("\t%d", array[i]);

}

printf("\n");

}

// Search each element in array, return first matching value found.

int linear\_search(int \* a, size\_t a\_len, int key) {

for(int i = 0; i < a\_len; i++) {

if(a[i] == key) {

return i;

}

}

return -1;

}

// Search by dividing the list in half each time, return the first matching value found.

int binary\_search(int \* a, size\_t a\_len, int key) {

int divide(int l, int r) {

int m = (l + r) / 2;

if(a[m] == key) {return m;}

if(l >= r) {return -1;}

// Don't check m because it's already checked at the start of this function.

if(key < a[m]) {

// Go left.

return divide(l, m - 1);

} else {

// Go right.

return divide(m + 1, r);

}

// Catchall in case something goes horribly wrong.

return -1;

}

return divide(0, a\_len - 1);

}

int main(void) {

srandom(time(NULL));

for(int i = 0; i < ARRAY\_LEN; i++) {array[i] = random() % MAX\_VAL;}

printf("Initial Array\n\n");

print\_array();

printf("\nSorted Array\n\n");

sort\_array();

print\_array();

printf( "\n-------------------------------------------"

"---------------------------------------\n\n");

printf("Linear Search\n\n");

print\_array();

for(int i = 0; i < linear\_search(array, ARRAY\_LEN, 32) + 1; i++) {

printf("\t");

}

printf("^\n\n");

printf("Binary Search\n\n");

print\_array();

for(int i = 0; i < binary\_search(array, ARRAY\_LEN, 32) + 1; i++) {

printf("\t");

}

printf("^\n\n");

return 0;

}

