Course: Programming Fundamental –ENSF 337

Lab #: Lab 5

Instructor: M. Moussavi

Student Name: Jiho Kim

Lab Section: B01

Date submitted: Oct 14, 2019

EXERCISE C

#include <stdio.h>

// Students should add the defintion of macro LARGEST\_OF\_THREE here.

#define LARGEST\_OF\_THREE(a, b, c) (( (a>b)? a:b) < c)? c : ( (a>b)? a:b)

int main(void)

{

double x = 0.300, y = 0.500, z= 0.999;

double largest = LARGEST\_OF\_THREE(x, y, z);

printf("\nTest 1: the largest value is %f", largest);

printf("\nTEST 2: the lagerst value is %f", LARGEST\_OF\_THREE(345, 99.8, 10));

printf("\nTEST 3: the lagerst value is %f", LARGEST\_OF\_THREE(4, 9.8, 10.0));

printf("\nTEST 4: the lagerst value is %f", LARGEST\_OF\_THREE(4.5, 4.5, 4.5));

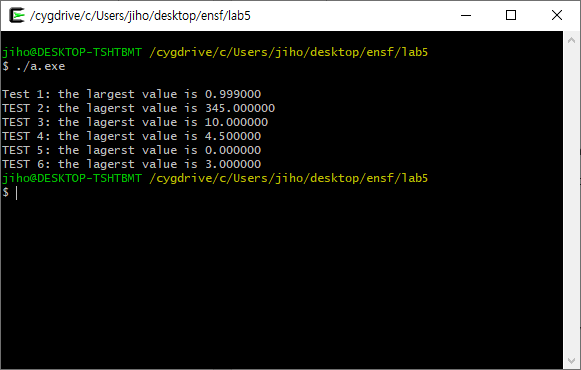
printf("\nTEST 5: the lagerst value is %f", LARGEST\_OF\_THREE(0.0, 0.0, 0.0));

printf("\nTEST 6: the lagerst value is %f",

LARGEST\_OF\_THREE(x \* 10, y \* 2 , z \* 3));

return 0;

}



EXERCISE D

/\* File: lab5exD.c

\* ENSF Fall 2019- lab 5 - Exercise D

\*/

#include "lab5exD.h"

#include <stdio.h>

#include <math.h>

#include <string.h>

int main(void)

{

Point alpha = { "A1", 2.3, 4.5, 56.0} ;

Point beta = { "B1", 25.9, 30.0, 97.0 } ;

printf ("Display the values in alpha, and beta: ");

display\_struct\_point(alpha);

display\_struct\_point(beta);

Point \*stp = &alpha;

printf ("\n\nDisplay the values in \*stp: ");

display\_struct\_point(\*stp);

Point gamma = mid\_point(stp, &beta, "M1");

printf ("\n\nDisplay the values in gamma after calling mid\_point function.");

printf ("Expected result is: M1 <14.10, 17.25, 76.50>");

printf("\n\nThe actual result of calling mid\_point function is: ");

display\_struct\_point(gamma);

swap (stp, &beta);

printf ("\n\nDisplay the values in \*stp, and beta after calling swap function.");

printf ("Expected to be:\nB1 <25.90, 30.00, 97.00>\nA1 <2.30, 4.50, 56.00>");

printf("\n\nThe actual result of calling swap function is: ");

display\_struct\_point(\*stp);

display\_struct\_point(beta);

printf("\n\nThe distance between alpha and beta is: %.2f. ", distance(&alpha, &beta));

printf ("(Expected to be: 53.74)");

printf("\nThe distance between gamma and beta is: %.2f. ", distance(&gamma, &beta));

printf ("(Expected to be: 26.87)");

return 0;

}

void display\_struct\_point(const Point x)

{

printf("\n%s <%.2lf, %.2lf, %.2lf>", x.label, x.x, x.y, x.z);

}

Point mid\_point(const Point\* p1, const Point\* p2, const char\* label)

{

// This function is incomplete and must be completed by the students

// YOU ARE NOT ALLOWED TO USE ANY STRING LIBRARY FUNCTIONS IN THIS FUNCTION

Point middle = {"?", 0, 0};

for(int i = 0; i < 10 || label[i] == '\0'; i++){

middle.label[i] = label[i];

}

middle.x = (p2->x - p1->x)/2 + p1->x;

middle.y = (p2->y - p1->y)/2 + p1->y;

middle.z = (p2->z - p1->z)/2 + p1->z;

return middle;

}

void swap(Point\* p1, Point \*p2)

{

Point p3;

p3.x = p1->x;

p1->x = p2->x;

p2->x = p3.x;

p3.y = p1->y;

p1->y = p2->y;

p2->y = p3.y;

p3.z = p1->z;

p1->z = p2->z;

p2->z = p3.z;

return;

}

double distance(const Point\* p1, const Point\* p2)

{

// This function is incomplete and must be completed by the students

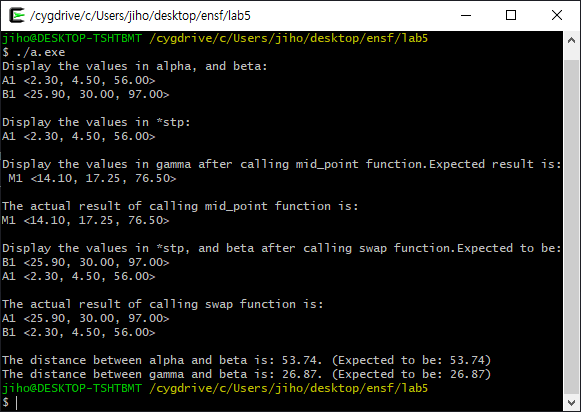
// NOTE: IN THIS FUNCTION YOU ARE NOT ALLOWED TO USE THE ARROW OPERATOR ->

double distance;

distance = sqrt(pow(((\*p2).x - (\*p1).x) , 2) + pow(((\*p2).y - (\*p1).y) , 2) + pow(((\*p2).z - (\*p1).z) , 2) );

return distance;

}



Exercise E

// lab5exE.c

// ENSF 337- Fall 2019, Exercise E

#include "lab5exE.h"

#include <stdio.h>

#include <math.h>

#include<string.h>

int main(void)

{

Point struct\_array[10];

int i;

int position;

populate\_struct\_array(struct\_array, 10);

printf("\nArray of Points contains: \n");

for(i=0; i < 10; i++)

display\_struct\_point(struct\_array[i], i);

printf("\nTest the search function");

position = search(struct\_array, "v0", 10);

if(position != -1)

printf("\nFound: struct\_array[%d] contains %s", position,

struct\_array[position].label);

else

printf("\nstruct\_array doesn't have label: %s.", "v0");

position = search(struct\_array, "E1", 10);

if(position != -1)

printf("\nFound: struct\_array[%d] contains %s", position,

struct\_array[position].label);

else

printf("\nstruct\_array doesn't have label: %s.", "E1");

position = search(struct\_array, "C5", 10);

if(position != -1)

printf("\nFound: struct\_array[%d] contains %s", position,

struct\_array[position].label);

else

printf("\nstruct\_array doesn't have label: %s.", "C5");

position = search(struct\_array, "B7", 10);

if(position != -1)

printf("\nFound: struct\_array[%d] contains %s", position,

struct\_array[position].label);

else

printf("\nstruct\_array doesn't have label: %s.", "B7");

position = search(struct\_array, "A9", 10);

if(position != -1)

printf("\nFound: struct\_array[%d] contains %s", position,

struct\_array[position].label);

else

printf("\nstruct\_array doesn't have label: %s.", "A9");

position = search(struct\_array, "E11", 10);

if(position != -1)

printf("\nFound: struct\_array[%d] contains %s", position,

struct\_array[position].label);

else

printf("\nstruct\_array doesn't have label: %s.", "E11");

position = search(struct\_array, "M1", 10);

if(position != -1)

printf("\nFound: struct\_array[%d] contains %s", position,

struct\_array[position].label);

else

printf("\nstruct\_array doesn't have label: %s.", "M1");

printf("\n\nTesting the reverse function:");

reverse(struct\_array, 10);

printf("\nThe reversed array is:");

for(i=0; i < 10; i++)

display\_struct\_point(struct\_array[i], i);

return 0;

}

void display\_struct\_point(const Point x , int i)

{

printf("\nstruct\_array[%d]: %s <%.2lf, %.2lf, %.2lf>\n",

i, x.label, x.x, x.y, x.z);

}

void populate\_struct\_array(Point\* array, int n)

{

int i;

char ch1 = 'A';

char ch2 = '9';

char ch3 = 'z';

for( i = 0; i < 10; i++)

{

/\* generating some random values to fill them elements of the array: \*/

array[i].x = (7 \* (i + 1) % 11) \* 100 - i /2;

array[i].y = (7 \* (i + 1) % 11) \* 120 - i / 3;

array[i].z = (7 \* (i + 1) % 11) \* 150 - i /4;

if(i % 2 == 0)

array[i].label[0] = ch1++;

else

array[i].label[0] = ch3--;

array[i].label[1] = ch2--;

array[i].label[2] = '\0';

}

}

int search(const Point\* struct\_array, const char\* label, int n)

{

// Students should complete the definiton of this function

// NOTE: YOU ARE NOT ALLOWED TO USE LIBRARY FUNCTION strcmp IN THIS FUNCTION

int counter = 0;

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

while(label[counter] != '\0'){

if(label[counter] == struct\_array[i].label[counter]){

counter++;

} else{

break;

}

if(label[counter] == '\0')

return i;

}

counter = 0;

}

return -1;

}

void reverse (Point \*a, int n)

{

// Students should complete the definiton of this function

Point b;

for(int i = 0; i < n/2; i++){

b = a[i];

a[i] = a[n-i-1];

a[n-i-1] = b;

}

return;

}

