Course: Programming Fundamental – ENSF 337

Lab #: Lab 5  
Instructor: Mansouri Habibabadi  
Student Name: Drew Hengehold

Lab Section: B01

Date submitted: October 19, 2022

UCID: 30151823

**Exercise A**

**Diagram

Description automatically generated**

**Diagram

Description automatically generated**

**Diagram

Description automatically generated**

**Exercise B**

/\*

\* File: lab5exB.c

\* ENSF 337, lab 5, Exercise B

\* Debugged by Drew Hengehold

\*/

#include <stdio.h>

#include <stdlib.h>

// This is a simple C program that is supposed to create an array of type double,

// (dyanamically on the heap), filling it with some arbitrary numbers and then

// using the array as needed. But the program doesn't do any thing useful becaues

// some flaws in the main function and improper design of the function

// create\_array.

void create\_array(double \*\* x, unsigned int n);

void populate\_array(double \*array, int n);

int main(void) {

printf("\nProgram started...\n");

double \* array = NULL;

int n = 20;

create\_array(&array, n);

if((double\*)array != NULL) {

populate\_array(array, n);

// displays half of the values of the array

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

printf("%f\n", \*array++);

}

array = 0;

// According to C standard, the program's behaviour, after the following

// call to the function free is considered "Undefined" and needs to be fixed.

free(array);

}

printf("Program terminated...\n");

return 0;

}

// THE FOLLOWING FUNCTION IS NOT PROPERLY DESINGED AND NEEDS TO BE FIXED

void create\_array(double \*\* x, unsigned int n) {

\*x = malloc(n\* sizeof(double));

if(x == NULL){

printf("Sorry Memory Not Available. Program Terminated.\n");

exit(1);

}

}

void populate\_array(double \*array, int n) {

int i;

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

array[i] = (i + 1) \* 100;

}

**Text

Description automatically generated**

**Exercise C**

**Diagram

Description automatically generated**

**Exercise D**

**Diagram

Description automatically generated**

**Exercise E**

**Source code(lab5exE.c)**

/\* File: lab5exE.c

\* ENSF 337 - lab 5 - Exercise E

\*/

// Author of mid point, swap and display : Drew Hengehold

#include "lab5exE.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)\n");

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, 0.0, 0.0};

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

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

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

for(int i = 0; i<2;i++)

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

return middle;

}

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

{

Point temp = {"Temp", 0.0, 0.0, 0.0};

temp = \*p2;

p2 = p1;

p1 = &temp;

}

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

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 = sqrt(pow((p1->x-p2->x),2)+pow((p1->y-p2->y),2)+pow((p1->z-p2->z),2));

return distance;

}

Text

Description automatically generated

**Exercise F**

**Source Code lab5exF.c**

// lab5exF.c

// ENSF 337, Exercise F

// Autor of reverse and search: Drew Hengehold

#include "lab5exF.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)

{

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

{

if(struct\_array[n].label[0] == label[0])

{

if(struct\_array[n].label[1]==label[1])

{

return n;

}

}

}

return -1;

}

void reverse (Point \*a, int n)

{

int i, j = 9, k = 0;

Point temp = {"",0,0,0};

while(k<5)

{

for(i = 0; i<2; i++)

{

temp.label[i] = a[k].label[i];

}

temp.x = a[k].x;

temp.y = a[k].y;

temp.z = a[k].z;

for(i=0;i<2;i++)

{

a[k].label[i]=a[j].label[i];

}

a[k].x=a[j].x;

a[k].y=a[j].y;

a[k].y=a[j].y;

for(i =0; i<2;i++)

{

a[j].label[i]=temp.label[i];

}

a[j].x=temp.x;

a[j].y=temp.y;

a[j].y=temp.y;

j--;

k++;

for(i =0; i<2; i++)

{

temp.label[i] = '\0';

temp.x = 0;

temp.y = 0;

temp.z = 0;

}

}

}

**Text

Description automatically generated**