Course: ENSF 337 - Fall 2020

Lab #: 5

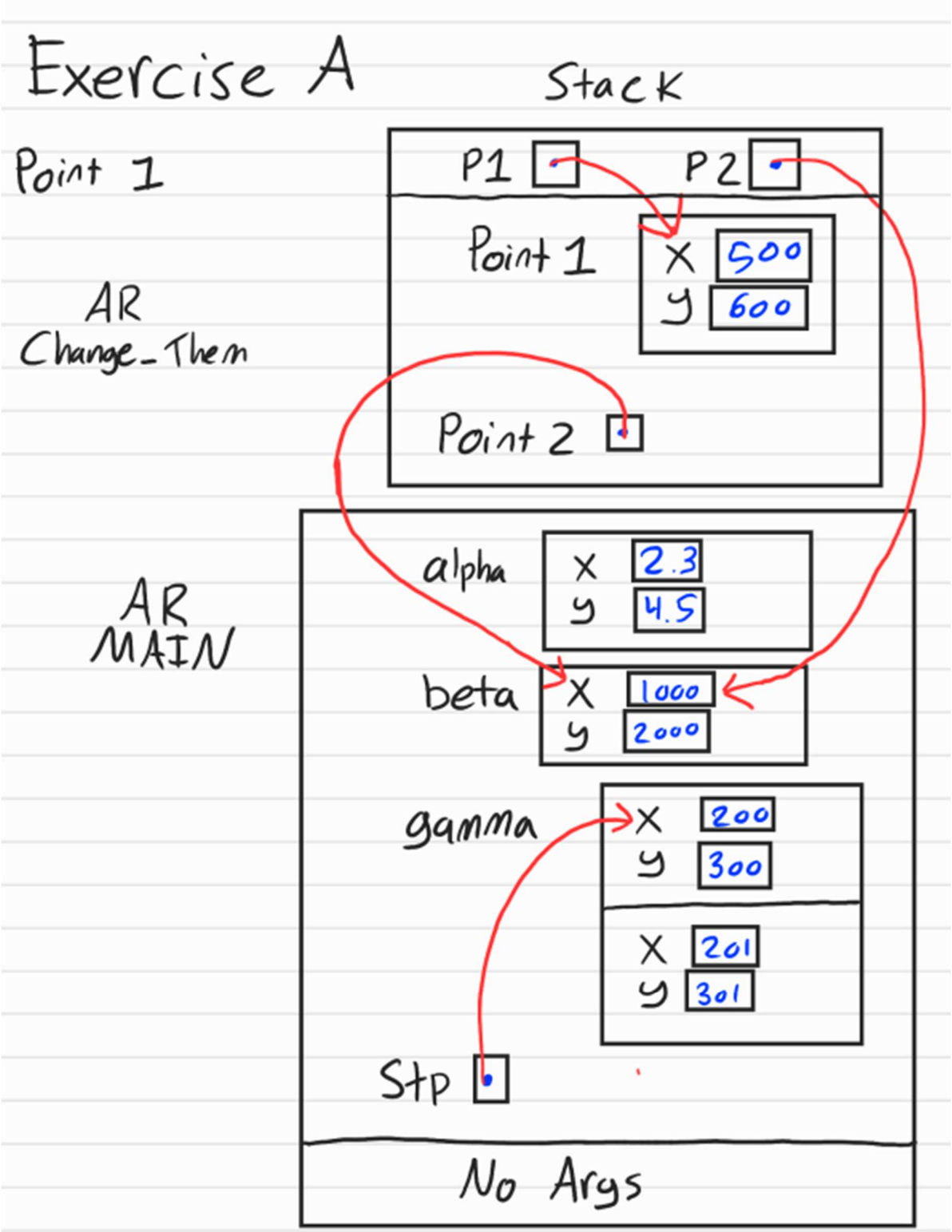
Instructor: M. Moussavi

Student Name: Alexander Sembrat (30089188) and Matthew Ho (30052684)

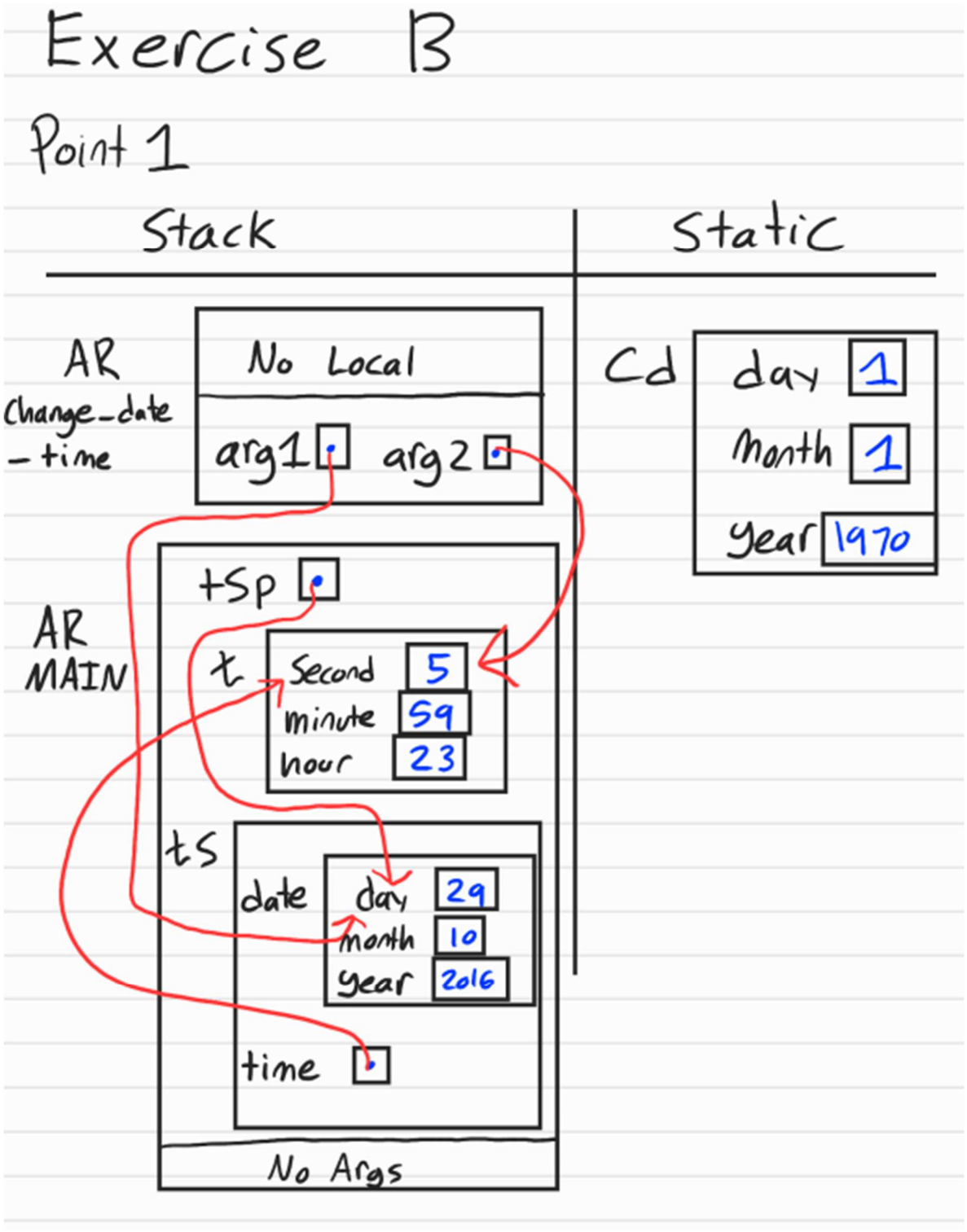
Lab Section: B01

Submission Date: October 19th 2020

Exercise A


Exercise A
Stack
Point 1
P1
P2
Point 1
X 500
y 600
AR
Change_Them
Point 2
alpha
X 2.3
y 4.5
AR
MAIN
beta
X 1000
y 2000
gamma
X 200
y 300
X 201
y 301
Stp
No Args

Exercise B


Exercise B
Point 1
Stack
Static
AR
Change_date
- time
No Local
arg1
arg2
Cd
day
1
month
1
year
1970
tSp
AR
MAIN
t
Second
5
minute
59
hour
23
tS
date
day
29
month
10
year
2016
time
No Args

Exercise D

```c
// ENSF 337 - fall 2020 - lab 5 - exercise D
// Matthew Ho(30052684) Alexander Sembrat (30089188)
// M. Moussavi
// lab5exe_D.c
// Date Submitted 10/19/2020

#include <stdio.h>
#include <stdlib.h>
#include "lab5exe_D.h"

int main(void) {
    char input_filename[30] = "lab5exe_D.txt";
    char output_filename[30]= "lab5exe_D_output.txt";
    IntVector intVec;
    intVec.number_of_data = 0;

    read_text_file(&intVec, input_filename);

    display_single_column(&intVec);

    display_multiple_column(&intVec, 5, output_filename);

    return 0;
}

void read_text_file (IntVector* vec, const char* input_filename){
    int nscan;
    FILE *fp = fopen (input_filename, "r");

    if(fp == NULL){
        fprintf(stdout, "Sorry cannot open the text file %s.\n", input_filename);
        exit(1);
    }

    do{
        nscan = fscanf(fp,"%d", &vec->storage[vec->number_of_data]); //this looks first at which file you want, then it looks at what type it is looking for, and then the location of the object is determ
        if(nscan == 1)// this checks if the value that was checked matches the one that is determined
            (vec->number_of_data)++;
        else if (nscan != EOF){
            fprintf(stderr, "Invalid data in %s.\n", input_filename);
            exit(1);
        }
    }while ((nscan != EOF) & (vec->number_of_data < MAX_CAPACITY));

    fclose(fp);
}

void display_single_column(const IntVector* intV){
    int i;
    for (i = 0; i < intV ->number_of_data; i++ )
        printf("%10d\n", intV ->storage[i]);
}

void display_multiple_column(const IntVector *intV, int col, const char* output_filename)
{
    FILE* Output = fopen(output_filename, "w");
    int i = 0;

    while(i < (*intV).number_of_data)
    {
        for(int j = 0; j < col && i < (*intV).number_of_data; j++, i++)
        {
            fprintf(Output,"%10d  ", intV -> storage[i]);
        }

        fprintf(Output, "\n");

    }
}
```

lab5exe_D_output - Notepad

File Edit Format View Help

```
       234        678        999        234
        33         22         99        222
        45         56         44         77
        92         91         81         73
        19         18         17        666
       555          1          3          6
```

Ln 1, Col 1 | 100% | Unix (LF) | UTF-8

Exercise E

```c
// File: lab5exE.c
// ENSF Fall 2020 - lab 5 - Exercise E
// Matthew Ho(30052684) Alexander Sembrat (30089188)
// M. Moussavi
// Date Submitted 10/19/2020

#include "lab5exE.h"
#include <stdio.h>
#include <math.h>
#include <string.h>

int main(void)
{
    Point alpha = { "A1", 2.3, 4.5, 56} ;
    Point beta = { "B1", 25.9, 30.0, 97} ;
    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);
}
```

```c

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};
    int i = 0;

    while((*label) != '\0')
    {
        (middle).label[i] = (*label);
        i++;
        label++;
    }

    middle.x = ((*p1).x + (*p2).x)/2;
    middle.y = ((*p1).y + (*p2).y)/2;
    middle.z = ((*p1).z + (*p2).z)/2;

    return middle;
}

void swap(Point* p1, Point *p2)
{
    Point storage = {"storage", 0, 0, 0};
    int i = 0;

    (storage).x = (*p1).x;
    (storage).y = (*p1).y;
    (storage).z = (*p1).z;

    (*p1).x = (*p2).x;
    (*p1).y = (*p2).y;
    (*p1).z = (*p2).z;

    (*p2).x = (storage).x;
    (*p2).y = (storage).y;
    (*p2).z = (storage).z;


    while((*p1).label[i]  != '\0' && (*p2).label[i] != '\0')
    {
        (storage).label[i] = (*p1).label[i];
        (*p1).label[i] = (*p2).label[i];
        (*p2).label[i] = (storage).label[i];
        i++;
    }

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

```c
}

double distance(const Point* p1, const Point* p2)
{
    double distance;
    distance = sqrt(pow((*p2).x -(*p1).x, 2) + pow((*p2).y -(*p1).y, 2) + pow((*p2).z - (*p1).z, 2));
    return distance;
}
```

Header File

```c
// File: lab5exE.c
// ENSF Fall 2020 - lab 5 - Exercise E
// Matthew Ho(30052684) Alexander Sembrat (30089188)
// M. Moussavi
// Date Submitted 10/19/2020


#ifndef lab5ExE_h
#define lab5ExE_h

/* a structure that represents a point on a Cartesian coordinates system. */
typedef struct point
{
    char label[10];  // a label for a point
    double x;        // x coordinate for point in a Cartesian coordinate system
    double y;        // y coordinate for point in a Cartesian coordinate system
    double z;
} Point;

Point mid_point(const Point* p1, const Point* p2, const char* label);
/* REQUIRES:
 *   p1 and p2 point to Point objects
 * PROMISES:
 *   returns an object of Point that its x and y coordinates are the middle-
 *   point of those objects that p1 and p2 are pointing to. The returned
 *   object's label will be the copy of argument label.
 */

void swap(Point* p1, Point *p2);
/* REQUIRES:
 *   p1 and p2 point to objects of Point
 * PROMISES:
 *   swaps the values of data members in the two objects *p1 and *p2.
 */

void display_struct_point(const Point x);

double distance (const Point* a, const Point* b);
/* REQUIRES:
 *   a and b point to objects of Point
 * PROMISES:
 *   returns the distance between objects that a and b are pointing to.
 */
#endif /* lab5ExE_h */
```

```
matth@DESKTOP-Q3LAO5I /cygdrive/c/Users/matth/Desktop
$ gcc -Wall lab5exE.c

matth@DESKTOP-Q3LAO5I /cygdrive/c/Users/matth/Desktop
$ ./a.exe
Display the values in alpha, and beta:
A1 <2.30, 4.50, 56.00>
B1 <25.90, 30.00, 97.00>

Display the values in *stp:
A1 <2.30, 4.50, 56.00>

Display the values in gamma after calling mid_point function.Expected result is:
 M1 <14.10, 17.25, 76.50>

The actual result of calling mid_point function is:
M1 <14.10, 17.25, 76.50>

Display the values in *stp, and beta after calling swap function.Expected to be:
B1 <25.90, 30.00, 97.00>
A1 <2.30, 4.50, 56.00>

The actual result of calling swap function is:
B1 <25.90, 30.00, 97.00>
A1 <2.30, 4.50, 56.00>

The distance between alpha and beta is: 53.74. (Expected to be: 53.74)
The distance between gamma and beta is: 26.87. (Expected to be: 26.87)
matth@DESKTOP-Q3LAO5I /cygdrive/c/Users/matth/Desktop
$
```

Exercise F

```c
// ENSF 337- Fall 2020, Exercise F
// File: lab5exF.c
// ENSF Fall 2020 - lab 5 - Exercise F
// Matthew Ho(30052684) Alexander Sembrat (30089188)
// M. Moussavi
// Date Submitted 10/19/2020

#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);
```

```c
    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)
{
    int i = 0;
    int count;
    int true = 0;
    int j;

    while(i < n)
    {
        j = 0;
        count = 0;
        while(label[j] != '\0')
        {
            if(struct_array[i].label[j] == (*label))
            {
                label++;
                count++;
            }
            j++;
        }
        i++;
        if(count == j)
        {
            count = i - 1;
            true = 1;
            i = n;
        }

    }

    if(true == 0)
    {
        return -1;
    }
    else
```

```
    {
        return count;
    }

    // Students should complete the definiton of this function
    // NOTE: YOU ARE NOT ALLOWED TO USE ANY C LIBRARY FUNCTION IN YOUR SOLUTION
}

void reverse (Point *a, int n)
{
    Point store[n];
    int j = 0;
    int i = 0;

    for(i = 0; i < n; i++)
    {
        for(j = 0; j < n; j++)
        {
            store[i].label[j] = (*a).label[j];
        }
        store[i].x = (*a).x;
        store[i].y = (*a).y;
        store[i].z = (*a).z;
        a++;
    }

    for(i = 0; i < n; i++)
    {
        for( j = 0; j < n; j++)
        {
            (*(a-n)).label[j] = store[n-i-1].label[j];
        }
        (*(a-n)).x = store[n-i-1].x;
        (*(a-n)).y = store[n-i-1].y;
        (*(a-n)).z = store[n-i-1].z;
        a++;
    }

    // Students should complete the definiton of this function
}
```

```
matth@DESKTOP-Q3LAO5I /cygdrive/c/Users/matth/Desktop
$ ./a.exe

Array of Points contains:

struct_array[0]: A9 <700.00, 840.00, 1050.00>

struct_array[1]: z8 <300.00, 360.00, 450.00>

struct_array[2]: B7 <999.00, 1200.00, 1500.00>

struct_array[3]: y6 <599.00, 719.00, 900.00>

struct_array[4]: C5 <198.00, 239.00, 299.00>

struct_array[5]: x4 <898.00, 1079.00, 1349.00>

struct_array[6]: D3 <497.00, 598.00, 749.00>

struct_array[7]: w2 <97.00, 118.00, 149.00>

struct_array[8]: E1 <796.00, 958.00, 1198.00>

struct_array[9]: v0 <396.00, 477.00, 598.00>

Test the search function
Found: struct_array[9] contains v0
Found: struct_array[8] contains E1
Found: struct_array[4] contains C5
Found: struct_array[2] contains B7
Found: struct_array[0] contains A9
struct_array doesn't have label: E11.
struct_array doesn't have label: M1.

Testing the reverse function:
The reversed array is:
struct_array[0]: v0 <396.00, 477.00, 598.00>

struct_array[1]: E1 <796.00, 958.00, 1198.00>

struct_array[2]: w2 <97.00, 118.00, 149.00>

struct_array[3]: D3 <497.00, 598.00, 749.00>

struct_array[4]: x4 <898.00, 1079.00, 1349.00>

struct_array[5]: C5 <198.00, 239.00, 299.00>

struct_array[6]: y6 <599.00, 719.00, 900.00>

struct_array[7]: B7 <999.00, 1200.00, 1500.00>

struct_array[8]: z8 <300.00, 360.00, 450.00>

struct_array[9]: A9 <700.00, 840.00, 1050.00>

matth@DESKTOP-Q3LAO5I /cygdrive/c/Users/matth/Desktop
$
```