CMSC 21 Lecture 13 (Structures) Assignment

montinola_13ass.c

C: > Users > admin > Documents > Jules > BSCS 2nd Semester > CMSC 21 - Lec > Lecture 13 Assignment > montinola_13ass.c > getSlopeInterceptForm(line)

```c
#include <stdio.h>
#include <stdlib.h> // to use abs() function
#include <math.h> // to use pow() and sqr() functions

// structure for line with two points, line as its tag
struct line
{
    // nested structure for points, with members as x and y coordinates and float for both data types
    struct point
    {
        float x;
        float y;
    } point1, point2; // declaring point1 and point2 as variables since it both will have x and y coordinates

    // variables that are required as the members for structure line, data types are all float
    float *midpoint; // make member midpoint as pointer in order to return an array
    float slope;
    float distance;
};

// float function with struct line variable as its formal parameter in order to access x and y values
float solveSlope(struct line line1)
{
    // access values of x and y of 2 points by accessing members of struc point inside struc line and store quotient to variable slope
    float slope = (line1.point2.y - line1.point1.y) / (line1.point2.x - line1.point1.x);

    // if slope value is equal to 0, call abs() function to remove negative sign
    if (slope == 0)
    {
        slope = abs(slope);
    }

    // return value of slope variable
    return slope;
}

// float function that returns a pointer to the midpoint coordinates with multiple values using array
// struct line variable as its formal parameter in order to access x and y values
float *solveMidpoint(struct line line1)
{
    // initialize a float array with 2 elemetns, declare as static to ensure that its memory is not deallocated when the function returns
    static float midpoint[2];

    // store the midpoint of x coordinates to first index of midpoint array
    midpoint[0] = (line1.point1.x + line1.point2.x) / 2.0;
```

```
    // store the midpoint of y coordinates to second index of midpoint array
    midpoint[1] = (line1.point1.y + line1.point2.y) / 2.0;

    // return array base address by using array name
    return midpoint;
}

// float function with struct line variable as its formal parameter in order to access x and y values
float solveDistance(struct line line1)
{
    // return the value rendered from the arithmetic
    // use of pow() function from math.lib with the formula as its first argument and the value of power to raise the formula to, as its second argument
    return sqrt(pow(line1.point2.x - line1.point1.x, 2) + pow(line1.point2.y - line1.point1.y, 2));
}

// void function since it does not return a value, only prints a statement
// struct line variable as its formal parameter in order to access x and y values
void getSlopeInterceptForm(struct line line1)
{
    // store the value returned by solveSlope() function to variable slope
    float slope = solveSlope(line1);
    // access values of y and x of point1 by accessing members of struc point inside struc line and store answer to variable slope
    // store value rendered from the arithmetic to variable intercept
    float intercept = line1.point1.y - (slope * line1.point1.x);

    // if slope value is equal to 0, there is no need to print x variable in slope-intercept form
    if (slope == 0)
    {
        // display the slope intercept form using the value of intercept
        printf("Slope-intercept form: y = %.2f\n", intercept);
    }
    // else slope is not equal to 0, x variable has a coefficient
    else
    {
        // display the slope intercept form using the value of slope and intercept
        printf("Slope-intercept form: y = %.2fx + %.2f\n", slope, intercept);
    }
}
```

```
int main(){

    // declare structure line variable which is line1
    struct line line1;

    // prompt the user to enter the x and y of the first point
    printf("Enter the x-coordinate and y-coordinate of the first point: ");
    // store the x and y values of first point by accessing x and y inside the struc point which is nested inside the struc line
    scanf("%f %f", &line1.point1.x, &line1.point1.y);

    // prompt the user to enter the x and y of the second point
    printf("Enter the x-coordinate and y-coordinate of the second point: ");
    // store the x and y values of second point by accessing x and y inside the struc point which is nested inside the struc line
    scanf("%f %f", &line1.point2.x, &line1.point2.y);

    // line1 as the actual parameters in the function call since we will perform arithmethic with its points
    line1.slope = solveSlope(line1); // make the member of struc line which is slope as a variable and store the value returned by the solveSlope(line1)
    line1.midpoint = solveMidpoint(line1); // make the member of struc line which is midpoint(a pointer) as a variable and store the array returned by the solveMidpoint(line1)
    line1.distance = solveDistance(line1); // make the member of struc line which is distance as a variable and store the value returned by the solveDistance(line1)

    printf("\nSlope: %.2f\n", line1.slope); // display the value of slope using the value stored at the variable by accessing the member slope of struc line
    printf("Midpoint: (%.2f, %.2f)\n", line1.midpoint[0], line1.midpoint[1]); // display the value of the elements of array midpoint(x, y) variable by accessing the member midpoint which is a pointer of str
    printf("Distance between two points: %.2f\n", line1.distance); // display the value of distance using the value stored at the variable by accessing the member distance of struc line

    // call the function to be able to display the slope intercept form
    getSlopeInterceptForm(line1);

    return 0;
}
```

Output:

```
Enter the x-coordinate and y-coordinate of the first point: 1 1
Enter the x-coordinate and y-coordinate of the second point: 0 1

Slope: 0.00
Midpoint: (0.50, 1.00)
Distance between two points: 1.00
Slope-intercept form: y = 1.00
PS C:\Users\admin>
```

Github Link:

https://github.com/jullyannemontinola/CMSC-21/tree/77d3e12dad8e2f64d792cd385f67492714bdde58/CMSC%2021/Lecture%2013%20Assignment