CMSC 21
2nd Semester AY 2022-2023
LECTURE 11
Jhoanna R. Olana - 202211140

Item 1:

Implement the following using structures:

Slope:

$$m = \frac{y_1 - y_2}{x_1 - x_2} = \frac{y_2 - y_1}{x_2 - x_1}$$

Midpoint:

$$\left(\frac{x_1 + x_2}{2}, \frac{y_1 + y_2}{2} \right)$$

Slope Intercept Form:

$$y = mx + b$$

Distance between two points:

$$d = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$$

```c
/*
This program determines the midpoint and distance of two inputted points and the slope and the equation of the line
passing through those points.
Written in 2023 by Jhoanna Olana
*/
#include <stdio.h>
#include <math.h>

/*
Holds the characteristics of two given points and the line that passes through them.
*/
struct line {
    struct point {
        float x;
        float y;
    } point1, point2;
    struct midpoint {
        float x;
        float y;
    } midpoint;
    float slope;
    float distance;
};

float solveSlope(struct line line1);
void solveMidpoint(struct line line1);
float solveDistance(struct line line1);
void slopeIntForm(struct line line1);

```

```c
int main() {
    // Creation of a variable "l" with a structure type "line".
    struct line l;

    // Prompting and reading the coordinates of two points.
    printf("%s %s\n\n", "This program determines the midpoint and distance of two inputted points and the slope",
    "and the equation of the line passing through those points.");

    printf("Enter x and y coordinate for the 1st point (separated by spaces): ");
    scanf("%f %f", &l.point1.x, &l.point1.y);

    printf("Enter x and y coordinate for the 2nd point (separated by spaces): ");
    scanf("%f %f", &l.point2.x, &l.point2.y);

    // Debug purpose -- to see if read inputs are correct
    // printf("Point1: (%d, %d)", l.point1.x, l.point1.y);
    // printf("%f", l.point1.y - l.point2.y);

    // Function call for the characteristics of the line that passes through the two given points.
    printf("Slope: %.2f\n", solveSlope(l));
    printf("Distance: %.2f\n", solveDistance(l));
    solveMidpoint(l);
    slopeIntForm(l);

    // Debug purpose -- to confirm that midpoint wasn't directly changed by the func.
    // printf("\nAfter func call, midpoint: (%.2f, %.2f)", l.midpoint.x, l.midpoint.y);

    return 0;
};
```

```c

/*
Solves and returns the slope of the line1 by accessing each nested members (lineM --> pointN --> n - coordinate).
Numerator and denominator are seperated to be read clearly.
*/
float solveSlope(struct line line1) {
    float slope, slope_num, slope_denom;

    slope_num = line1.point1.y - line1.point2.y;

    slope_denom = line1.point1.x - line1.point2.x;

    slope = slope_num / slope_denom;

    return slope;
}

/*
Same thing with solveSlope functionn except that it solves and prints the midpoint and some members of line1 (lineM.midpoint.n)
are used to hold (not changed) the values of each midpoint coordinates.
*/
void solveMidpoint(struct line line1) {
    line1.midpoint.x = (line1.point1.x + line1.point2.x) / 2.0;

    line1.midpoint.y = (line1.point1.y + line1.point2.y) / 2.0;

    printf("Midpoint: (%.2f, %.2f)\n", line1.midpoint.x, line1.midpoint.y);
}

```

```c
/*
Same thing with solveSlope function except that it solves for the distance of the points and uses the math library for squaring(pow)
and getting the square root(sqrt).
*/
float solveDistance(struct line line1) {
    float distance, x_squared, y_squared;

    x_squared = pow((line1.point1.x - line1.point2.x), 2.0);

    y_squared = pow((line1.point1.y - line1.point2.y), 2.0);

    distance = sqrt(x_squared + y_squared);

    return distance;
}

/*
Same thing with solveDistance function except that it solves for various elements, uses some already defined func (solveSlope) to solve
for some elements (b == y-intercept), and prints an equation.
*/
void slopeIntForm(struct line line1) {
    // y = mx + b
    float b, m;

    m = solveSlope(line1);
    b = (-m * line1.point1.x ) + line1.point1.y;

    printf("Equation of the line: y = %.2fx + %.2f", m, b);
}
```

```
106    for some elements (b == y-intercept), and prints an equation.
107    */
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

```
PS C:\Users\olana> cd "c:\Users\olana\OneDrive - University of the Philip
lana } ; if ($?) { .\lec13_olana }
This program determines the midpoint and distance of two inputted points

Enter x and y coordinate for the 1st point (separated by spaces): 1 2
Enter x and y coordinate for the 2nd point (separated by spaces): 3 4
Slope: 1.00
Distance: 2.83
Midpoint: (2.00, 3.00)
Equation of the line: y = 1.00x + 1.00
PS C:\Users\olana\OneDrive - University of the Philippines\1st Year - 2nd
```

Github Link: [CMSC21/Lecture13 at main · jrolana/CMSC21 · GitHub]