Pre-Lab 4 Report John Santiago

ECE 100-001 Teammates:

Prof. Oruklu Lab Date 9/26/17

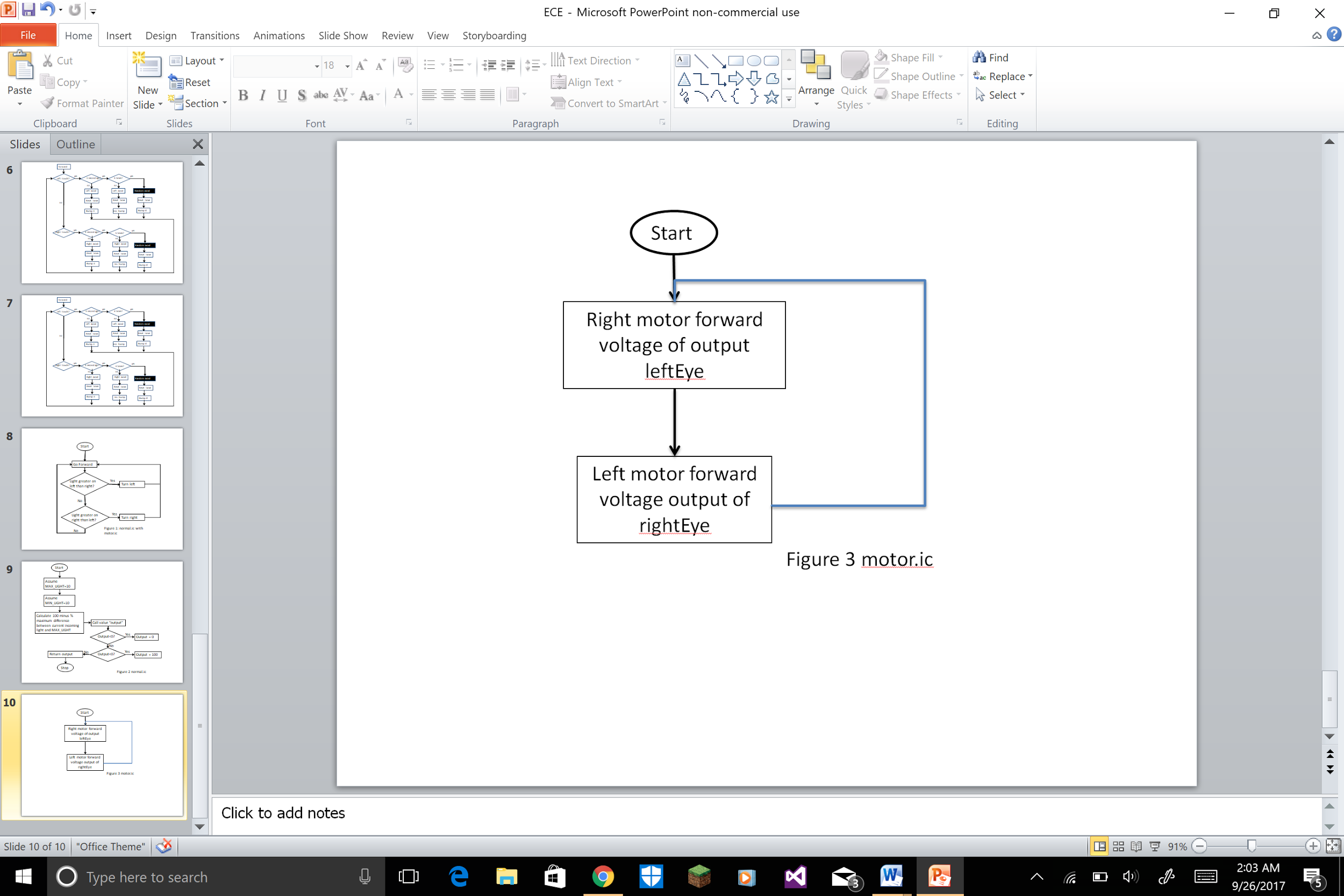
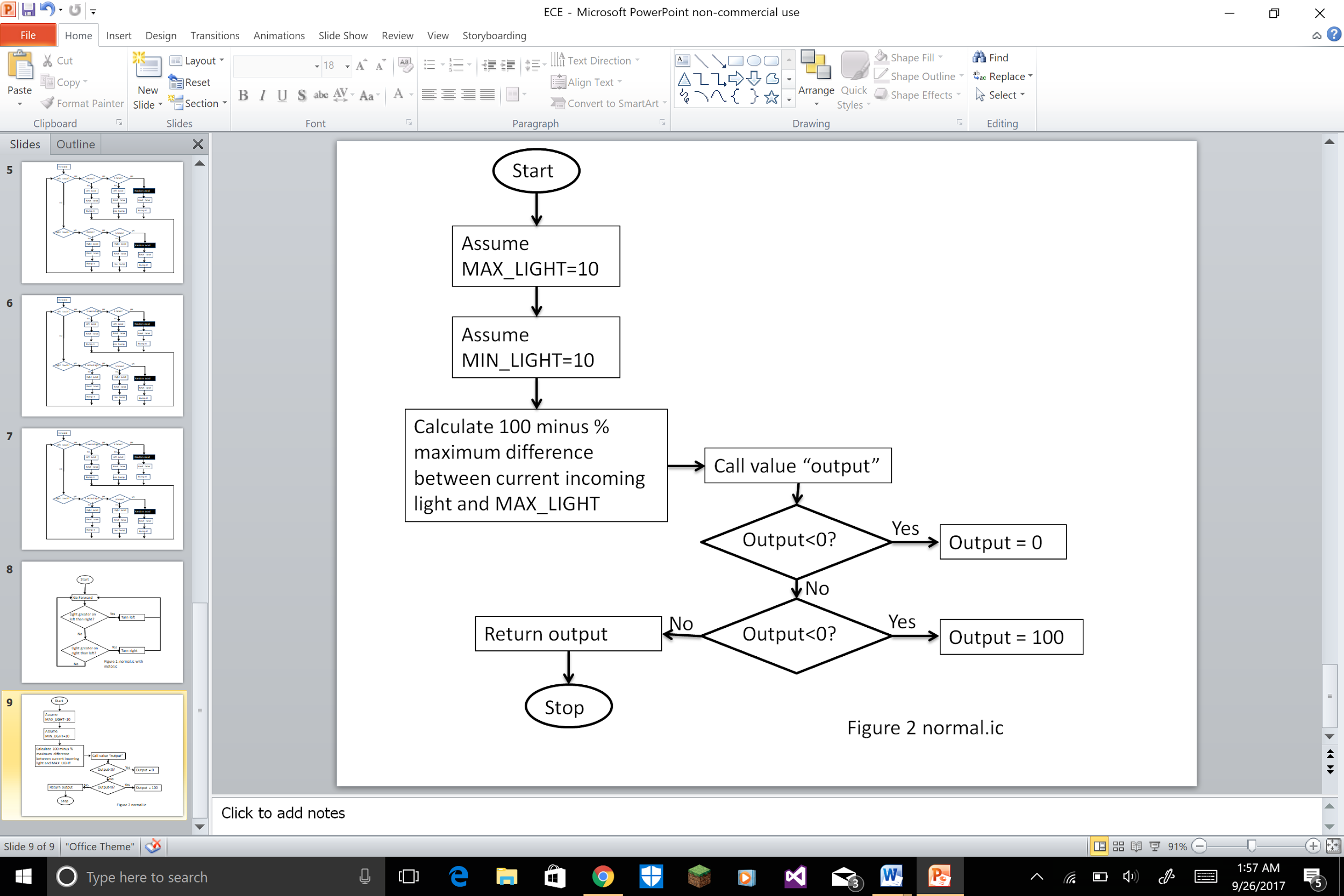
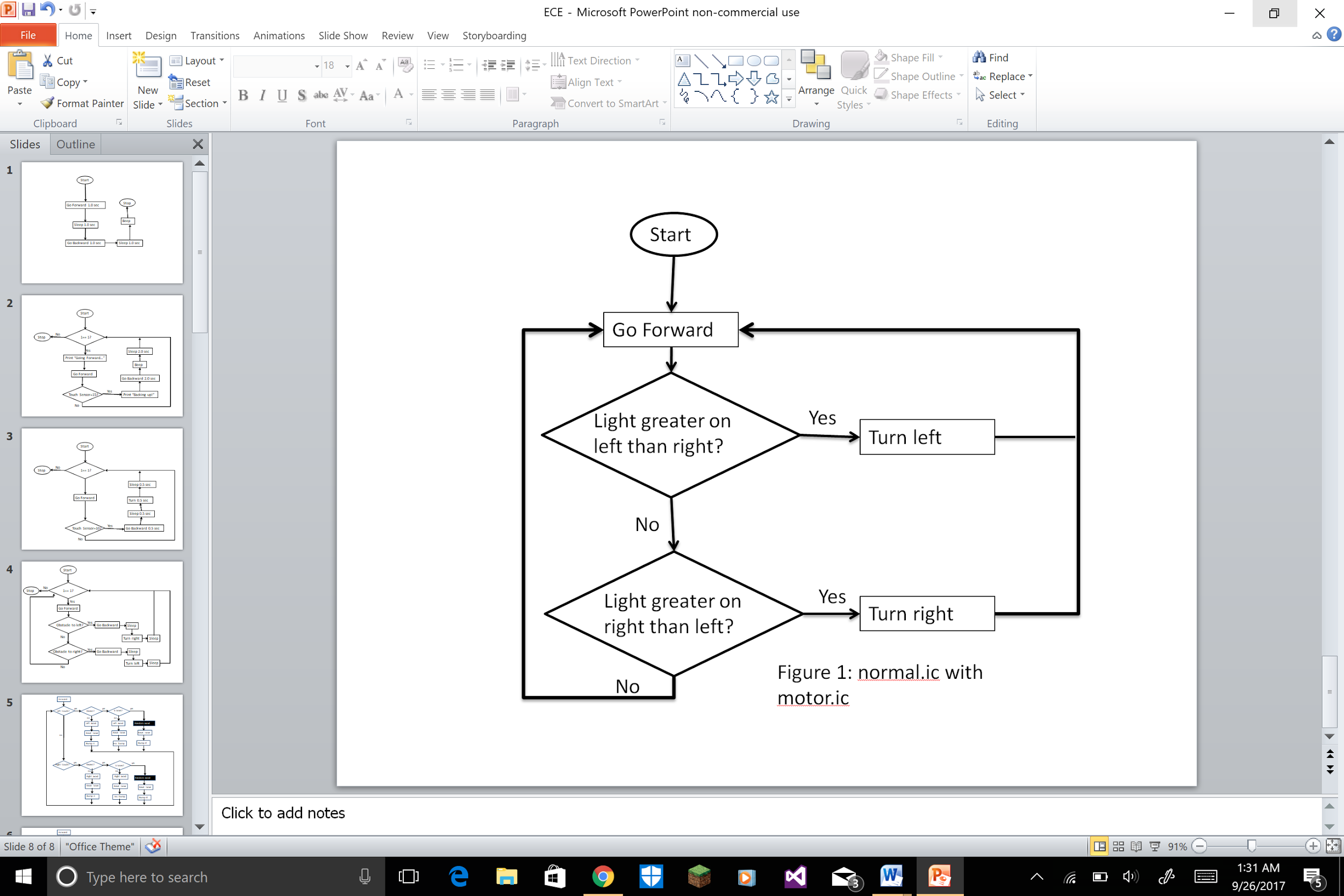
TA: Due Date 9/30/17

**Problem Statement**

The goal is to create a robot to find its way through the maze using light sensors efficiently which can be useful for search and rescue missions. To start, the robot should complete the maze by following a tape path.

**Investigation/Research**

In the book, Robotic Explorations by Fred G Martin, explains how to code a robot to follow a tape path using light sensors. However, before coding, the way light sensors work should be understood. To understand how light sensors work, a situation must be provided. Say, the robot is following a straight tape path that suddenly curves to the right. The amount of light hitting the right sensor should be greater than that of the left sensor. The Handy Bug should start turning to the right by slowing the right motor down. For the Handy Bug to perform such actions, a combination of “normal.ic” and “motor.ic” programs can be used as shown in Figure 1. The program, “normal.ic” Figure 2 (Martin 2001, p.84), assigns an integer value to each sensor indicating how much light there is. If the value for the leftEye is greater than the rightEye, then “motor.ic” Figure 3 (Martin 2001, p.85) will respond by slowing down the left motor and vice versa.



**Proposal**

For the robot to navigate the maze using light sensors, the Handy Board should use the programs “motor.ic” and “normal.ic” but more complex. Some goals to accomplish this is to first run the Handy Bug with motor.ic and normal.ic. Then work on its reliability by editing the code and the framework of the robot. Finally, we perfect the robot’s efficiency with the coding and make any modifications to its hardware.

**References**

1. Martin, Fred G. 2001. Robotic Explorations: A Hands-On Introduction to Engineering. New Jersey: Prentice Hall.

2.Oruklu, Erdal. 2017. *ECE 100 Lecture Notes.* Chicago: Illinois Institute of Technology, Electrical and Computer Engineering Department.

**Appendix**

**Normal.ic**

//normal.ic original

//converts light sensor readings to 0-100 power levels

int normalize(int light)

{

int MAX\_LIGHT=10;

int MIN\_LIGHT=200;

int output=100-((MAX\_LIGHT-light)/(MAX\_LIGHT-MIN\_LIGHT))\*100;

if (output<0)

output=0;

if (output>100)

output=100;

return output;

}

**Motor.ic**

void main()

{

while(1)

{

motor(LEFT\_MOTOR, normalize(rightEye);

motor(RIGHT\_MOTOR, normalize(analog((leftEye));

}

}