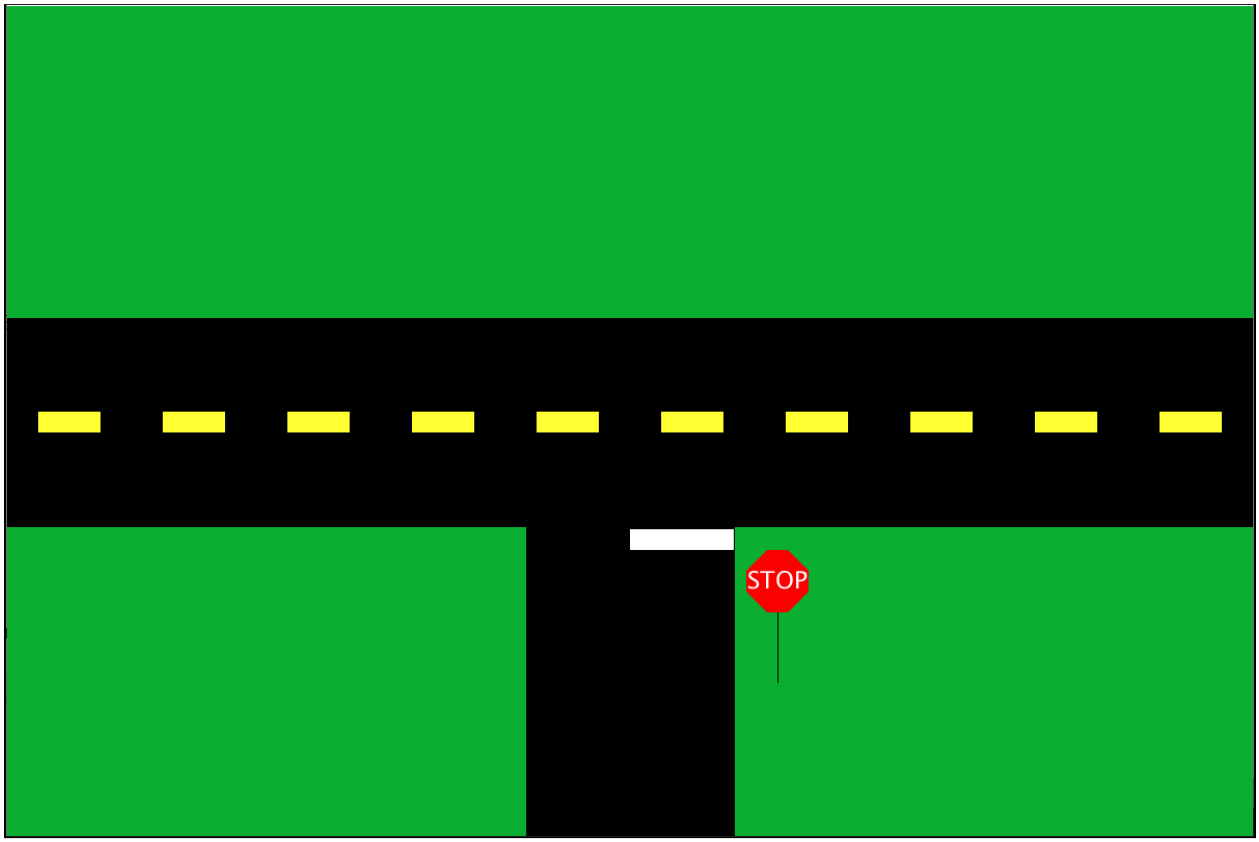
# Signal Before Turning Program

*By Christopher Pawlus, Hasinuz Zaman 10/10/21*
*AP CSA, Mr. Hans*

Our decision in the pair project was signalling before turning. Its purpose is to give people who drive or are learning to drive an understanding of one of the many rules in the NYS driving handbook which is signaling before a turn. At first, we thought of a couple of ways we could do this project from an animation or just a simple art piece. Our original plan was to create a four-way intersection with a car that approaches and turns with its signal. As we progressed our plans changed and we decided to create a simple background with a T-intersection with a stop sign and have a car do an animation for signaling. Then we decided that we would have a mini animation of a car approaching the stop sign, stopping, signalling before turning, and continue to finish executing the program. Down below are some of the significant pieces of code in our program:

## Background



For our background, we decided to keep it simple by going with a light green background and two roads. For our background, we used the background(); function with certain RGB values that look like grass. After that, we moved on to making the road and we took two rectangles and used the rectangle mode (rectMode(CENTER);) as CENTER. We then took smaller rectangles and filled them with yellow to make the traffic lines that separate the lanes and cars.

```
size(1200,800);
rectMode(CENTER);

//background art
background(10,175,50);//background
fill(0,0,0);//pavement
rect(600,400,1200,200);// horizontal rectangle
rect(600,600,200,600);// vertical rectangle

//yellow lines
fill(255,255,50);
noStroke();
rect(60,400,60,20);
rect(180,400,60,20);
rect(300,400,60,20);
rect(420,400,60,20);
rect(540,400,60,20);
rect(660,400,60,20);
rect(780,400,60,20);
rect(900,400,60,20);
rect(1020,400,60,20);
rect(1140,400,60,20);

//white approach line and stop sign
fill(255,255,255);
rect(650,513,100,20);
fill(255,255,255);
```

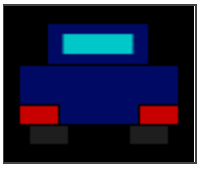
After that, we created the stop sign with two rectangles and four triangles using the rect(); and triangle(); functions. They were all filled in with the color red and we used the textSize(); and text(); functions to create text on the stop sign. Lastly, we created a line to represent the pole of the sign itself.

```
// start stop sign
fill(255,255,255);
fill(250,0,0);
noStroke();
rect(742,553,60,20);//vertical rect
rect(742,553,20,60);//horizontal rect

//triangles
fill(250,0,0);
triangle(732,523,712,543,732,543);//top left tri
fill(250,0,0);
triangle(752,523,772,543,752,543);// top right tri
fill(250,0,0);
triangle(752,583,772,563,752,563);// bottom right tri
fill(250,0,0);
triangle(732,583,712,563,732,563);// bottom left tri

//stop text and post
stroke(0);
textSize(23);
line(742,583,742,650);
fill(255,255,255);
text("STOP",713,560);
// end stop sign
```

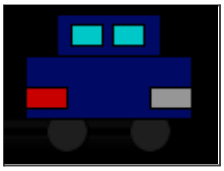
## Car Models

For our car models, we decided to make two models with one being a view from the rear of the car and the other from the side of the car. We also made it a 1D animation/picture. We tried to keep it as simple and realistic as possible so we made two blue cars with wheels, taillights, headlights, windows/windshield, and the body. The first version was the view from the rear (left image) and the second version was the view from the left side (right image) with code.

```
/* car body v1
fill(0,10,100);
stroke(0);
rect(650,t1,80,30);
noStroke();
rect(650,t2,50,20);
fill(0,200,200);
// windshield
rect(650,t2,35,10);
fill(200,0,0);
stroke(0);
//blinkers
rect(620,t3,20,10);
rect(680,t3,20,10);
//wheels
fill(30,30,30);
rect(625,t4,20,10);
rect(675,t4,20,10);
noStroke();
*/
```



```
/* car body v2
// wheels
stroke(0);
fill(30,30,30);
circle(x1,476,20);
circle(x2,476,20);
// body
fill(0,10,100);
rect(x3,455,80,30);
rect(x3,430,50,20);
fill(0,200,200);
// side windows
rect(x4,430,15,10);
rect(x5,430,15,10);
// blinkers & headlights
fill(200,0,0);
rect(x6,460,20,10);
fill(150,150,150);
rect(x7,460,20,10);
noStroke();
*/
```

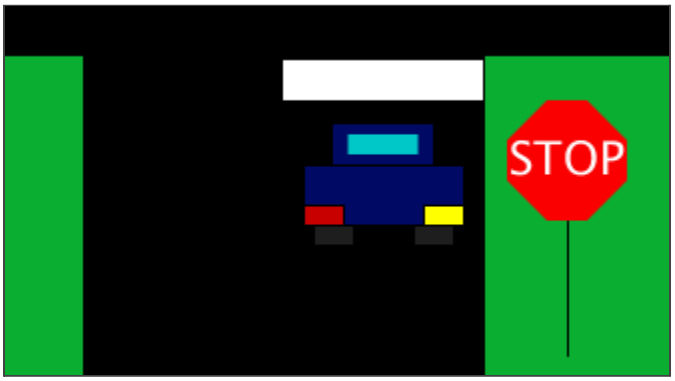


Here are the x and y values for the initial starting values of the car models. X values are x1-x7. Y values are t1-t4.

```
int t1 = 700;
int t2 = 675;
int t3 = 710;
int t4 = 720;
int x1 = 730;
int x2 = 770;
int x3 = 750;
int x4 = 740;
int x5 = 760;
int x6 = 720;
int x7 = 780;
```

## Signaling Animation

For our animation we had some errors and bugs come our way with using the delay(); function where there would be a time halt in the program for a certain time. After we realized that this function wouldn't work, we checked for different solutions and approaches and the solution for it was using the built-in functions called keyPressed() and keyReleased(). Every time the E key was pressed it would be used as user input and then have the colors of the rear tail light change from red to yellow as many times the user presses and releases the key, but the program counts a limit of 6 blinks allowing the car to turn after it reaches a value of 6 for one of the variables.

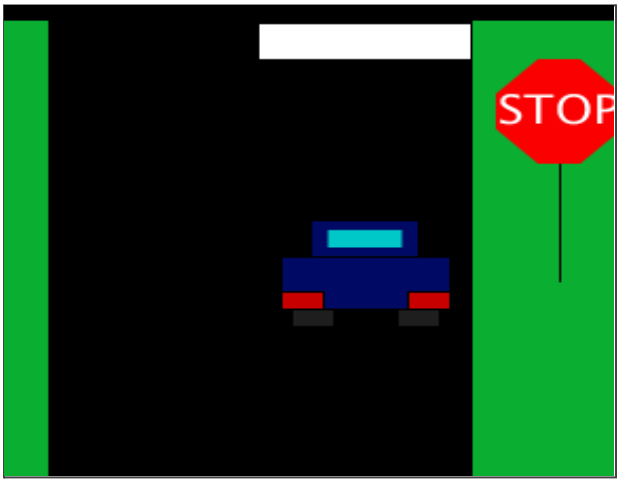



```
void keyReleased() {
  if((motion == false) && (keyCode == 69)) {
    fill(200,0,0);
    rect(680,t3,20,10);
    signal +=1;
  }
}
```

```
if((keyCode == 69)&&(motion == false)) {
  fill(255,255,0);
  rect(680,t3,20,10);
}
```

## User Controls

User controls stood out to us because user input controls can make our program more interactive and interesting. In the program, the user has three control options for certain parts of the program and those controls are the W, E, and D keys. When the program first starts up there's just a background, but when the user presses the W key the rear-end view of the car can be seen. When W is pressed the car moves up by a pixel and when W is held down it moves up by a pixel continuously. Something that is important about this part is the starting values of each shape. We checked the exact x and y values for the shapes and stored them in variables and allowed them to be moved by either subtracting or adding one value (A pixel) from the variables for them to move.



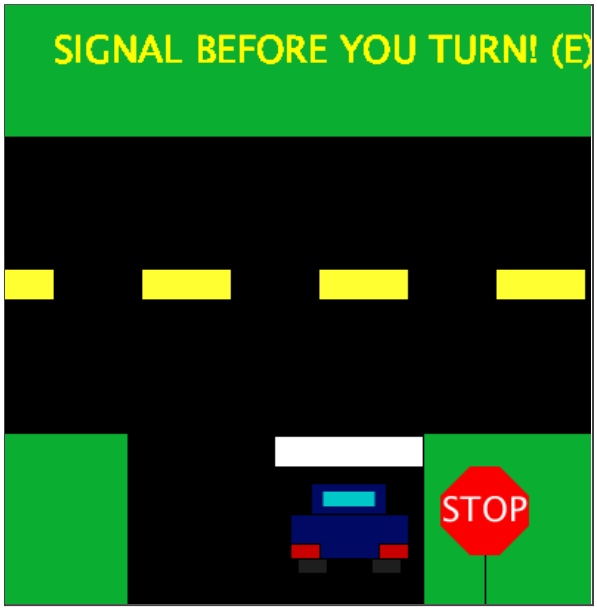

```
if((keyCode == 87)&&(motion == true)) {
  //car body v1
  t1 = t1 - move;
  fill(0,10,100);
  stroke(0);
  rect(650,t1,80,30);
  noStroke();
  t2 = t2 - move;
  rect(650,t2,50,20);
  fill(0,200,200);
  // windshield
  rect(650,t2,35,10);
  fill(200,0,0);
  stroke(0);
  //blinkers
  t3 = t3 - move;
  rect(620,t3,20,10);
  rect(680,t3,20,10);
  //wheels
  fill(30,30,30);
  t4 = t4 - move;
  rect(625,t4,20,10);
  rect(675,t4,20,10);
  noStroke();
}
```

```
int t1 = 700;
int t2 = 675;
int t3 = 710;
int t4 = 720;
int x1 = 730;
int x2 = 770;
int x3 = 750;
int x4 = 740;
int x5 = 760;
int x6 = 720;
int x7 = 780;
```

After the car reaches a certain point on the y-axis, the program stops the W key control and allows the E key for interaction. We then thought about how we can allow the user to be the blinker of the car by releasing and pressing the E key to act as a blinker and once it reaches a certain amount of times blinked, which is 6, it will then allow the car to turn with the D key. There are also messages (SIGNAL BEFORE YOU TURN!, TURN (D), YOU SIGNALLED BEFORE YOU TURNED, THANK YOU!) about this showing the controls for the user and what they are supposed to do.

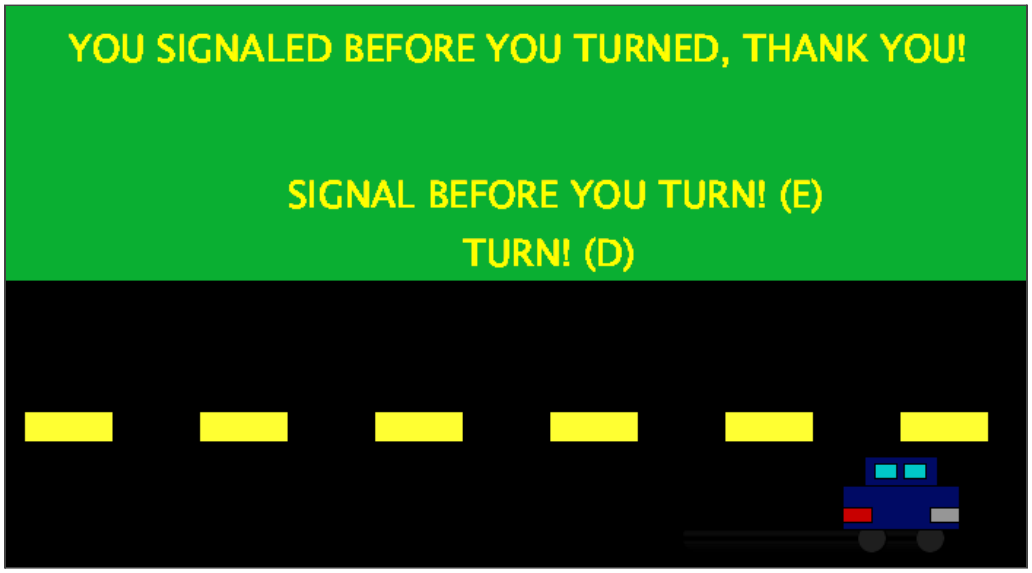
```
if((keyCode == 69)&&(motion == false)) {
  fill(255,255,0);
  rect(680,t3,20,10);
}
```

```
void keyReleased() {
  if((motion == false) && (keyCode == 69)) {
    fill(200,0,0);
    rect(680,t3,20,10);
    signal +=1;
  }
}
```




Lastly, when the user interacts with the controls and triggers the correct amount of times the car blinker blinks they will then be allowed to turn, but they would always have to look both ways to make sure no cars are coming their way in real life. A message saying ‘TURN (E)’ pops up

allowing the user to turn into the road after signaling and after the road is clear. The D key has the same properties as the W key, but moving the new car from the view of its side, going right on the screen. The program stops after all three messages have been displayed and a certain value is reached on the second animation.




```
if((keyCode == 68)&&(turn == true)) {
  //car body v2
  // wheels
  x1 = x1 + move;
  x2 = x2 + move;
  stroke(0);
  fill(30,30,30);
  circle(x1,476,20);
  circle(x2,476,20);
  // body
  fill(0,10,100);
  x3 = x3 + move;
  rect(x3,455,80,30);
  rect(x3,430,50,20);
  fill(0,200,200);
  // side windows
  x4 = x4 + move;
  x5 = x5 + move;
  rect(x4,430,15,10);
  rect(x5,430,15,10);
  // blinkers & headlights
  fill(200,0,0);
  x6 = x6 + move;
  rect(x6,460,20,10);
  fill(150,150,150);
  x7 = x7 + move;
  rect(x7,460,20,10);
  noStroke();
 }
```

## Challenges

Some of the challenges we've faced were the animation of the program and user input. For our animation, we faced a bug with the delay(); function where the colors of the taillight were not changing. We quickly tried to find a different solution which was using key presses and key releases from a certain key. The other challenge we faced was user input controls. We came towards a problem of having other controls being used when they weren't supposed to. So, we had to create boolean values and set them as true or false at certain parts of the program so that they can be used to allow the other conditional statements for the user controls to work as intended.