Feature Number: 2

Feature Name: Virtual Machine

Programmer: Peter Giovi

Tester: Collin Hughes

- **VirtualMachineClass** - VMClass will be called from the game with a filepath to be used for the Player Object.
    - GiveFilepath Method
        - Will Read in the file provided into an array of strings
        - VMClass will be called from the PlayerObjectClass (GetInstruction Method)
            - PlayerObjectClass will send an input struct with any relevant data
                - The programmer will have to choose what information to use and how to use it
                - This may include information such collision objects within a certain angle and distance, the player's velocity and position
            - PlayerObjectClass will be returned an output struct with values for manipulating the player that frame
                - Return a value for acceleration, steering and braking
    - GetInstruction Method
        - Will Read in the input data
            - Will be an array of structs and additional struct
                - Struct will contain position and velocity of objects within the field of view
                - Struct will also contain position and velocity of player
        - Will increment to the next string in the array provided by the GiveFilepath Method
        - Will Process the current instruction based off the interpreted string and the current input struct
        - Will return a struct with what the Player Object should do
        - PlayObjectClass will call these methods in the OnUpdate method

- **Interpretive Language**
        - Programmer will have access to any data points through the interface,

- The Language will include commands to compare data points, send data points to the Player Object and perform basic mathematical functions that might be relevant for the coder to have.

Command	Action
add; x; y;	Adds x + y
sub; x; y;	Subtracts x + y
Mul; x; y;	Multiplies x + y
Div; x; y;	Divides x + y
Acc; x;	Accelerates x
Trn; x;	Turns x
Brk; x;	Brakes x
Madd; x; y;	Adds memory[x] + memory[y]
Msub; x; y;	Subtracts memory[x] - memory[y]
MMul; x; y;	Multiply memory[x] * memory[y]
Mdiv; x; y;	Divide memory[x] / memory[y]
Macc; x;	Accerates memory[x]
Mtrn; x;	Turns memory[x]
Mbrk; x;	Brakes memory[x]
Mvi; x;	Moves tempMath into memory[x]
Mvo; x;	Moves memory[x] into tempMath