1. (Watch Gameplay.mp4 file)

2a. The programming language is a version of javascript. The overall purpose of the program is to lay out the fundamentals that would be used in a scrolling-shooter game (think space invaders or a scrolling bullet hell game). The video first illustrates movement which is based off of the arrow keys and enemy movement. It then shows bullet shooting, bullet movement, and cycling through multiple bullets. Next it shows bullet collisions with enemy as well as bullet teleportation and deactivation off-screen. It finally shows the enemy teleporting back to the top of the screen to simulate scrolling and the fighter colliding with wall bounds.

2b. The first major obstacle I had to tackle in my program was specifically the bullet array. In my program I knew that I needed to be able to cycle through 10 different bullets but it was difficult figuring out a way to do that. I tried using individual objects and concluded upon using an object that had arrays within it which would let me point towards individual properties of any numbered bullet I wanted to. The way I concluded upon using the array inside an object was through collaborative development. This was a major struggle for me because I also did not know how to use for loops at the time and spent lots of time debugging and figuring out them. The array within an object turned out to be the best solution to this because of the fact that it’s an object which allows me to easily point to its individual properties (i.e width, height) along with the array allowing me to easily pick which bullet I want to be referring to. The other problem I had a struggle with was my attempt to simplify a function which had 10 lines which all set the property of the bullets x and y to properly move them on screen. The problem was firstly I had to use a toString() function to change the i value in the for loop to a string, and then secondly I discovered after much wasted time that I was reusing a global variable already used in a for loop resulting in the function only setting the position of bullet 0.

function bulletCollide(obj1, obj2) {

for (i = 0; i < 10; i++) {

if ((bullet.X[i] < obj2.X + obj2.W) && (bullet.X[i] + bulletInfo.W > obj2.X)) {

if ((bullet.Y[i] < obj2.Y + obj2.H) && (bullet.Y[i] + bulletInfo.H > obj2.Y)) {

return(true);

}

}

}

}

2d.

This abstraction greatly helped manage the complexity of the program. It is a basic object collision function specialized for collisions involving the bullet elements. It does this by including a loop which goes through every single bullet (0 – 10) and then selecting which bullet is colliding with the obj (in this case enemy or top of the screen). This simplifies the process of handling bullet collision down to just calling a function whenever there is a collision detected. It uses the value of X and Y for bullet which allows for 10 different numbers to be inputted for each as well as obj2 width, height, x, and y values to allow any new objects for the bullets to collide with to easily check whether they did collide or not. The bullet.X[i] is an object with an array built in to simplify the cycling of bullets 1-10. The for loop automatically selects which bullet is colliding avoiding me having to copy the code 10 different times and call every function just to check for every possible bullet element collision.

3.

var moveRight = false;

var moveLeft = false;

var moveUp = false;

var moveDown = false;

var bulletIndex = -1;

var bulletCount = 30;

var i;

var fighter = {

X : getProperty("fighter","x"),

Y : getProperty("fighter","y"),

W : getProperty("fighter","width"),

H : getProperty("fighter","height"),

};

var enemy = {

X : getProperty("enemy","x"),

Y : getProperty("enemy","y"),

W : getProperty("enemy","width"),

H : getProperty("enemy","height"),

};

var bullet = {

X : [400,400,400,400,400,400,400,400,400,400],

Y : [-3,-3,-3,-3,-3,-3,-3,-3,-3,-3],

A : [0,0,0,0,0,0,0,0,0,0],

};

var bulletInfo = {

W : getProperty("bullet0","width"),

H : getProperty("bullet0","height"),

};

var rightWall = {

X : 320,

Y : 0,

H : 450,

W : 0,

};

var leftWall = {

X : 0,

Y : 0,

H : 450,

W : 0,

};

var topWall = {

X : 0,

Y : -30,

H : 0,

W : 320,

};

var boundaryWall = {

X : 0,

Y : 125,

H : 0,

W : 320,

};

var bottomWall = {

X : 0,

Y : 450,

H : 0,

W : 320,

};

function update() {

bulletUpdate();

enemy.Y = enemy.Y + 3;

if (moveRight) {

fighter.X = fighter.X + 2.5;

}

if (moveLeft) {

fighter.X = fighter.X - 2.5;

}

if (moveUp) {

fighter.Y = fighter.Y - 2.5;

}

if (moveDown) {

fighter.Y = fighter.Y + 2.5;

}

if (isCollide(enemy, bottomWall)) {

enemy.Y = 0;

enemy.X = randomNumber(30, 285);

}

setPosition("fighter", fighter.X, fighter.Y);

setPosition("enemy", enemy.X, enemy.Y);

}

function enemyUpdate() {

bulletCollision();

}

function shootBullet() {

bulletCount = bulletCount - 1;

bullet.A[bulletIndex] = 1;

bullet.X[bulletIndex] = fighter.X + 20;

bullet.Y[bulletIndex] = fighter.Y - 30;

if (bulletIndex >= 9) {

bulletIndex = 0;

}

else {

bulletIndex = bulletIndex + 1;

}

console.log(bulletIndex);

//bulletCD();

}

onEvent("screen1", "keydown", function(event) {

if (event.key == 'Right') {

moveRight = true;

}

if (event.key == 'Left') {

moveLeft = true;

}

if (event.key == 'Up') {

moveUp = true;

}

if (event.key == 'Down') {

moveDown = true;

}

if (isCollide(fighter,rightWall)) {

moveRight = false;

}

if (isCollide(fighter,leftWall)) {

moveLeft = false;

}

if (isCollide(fighter,boundaryWall)) {

moveUp = false;

}

if (isCollide(fighter, bottomWall)) {

moveDown = false;

}

});

onEvent("screen1", "keyup", function(event) {

if (event.key == 'Right') {

moveRight = false;

}

if (event.key == 'Left') {

moveLeft = false;

}

if (event.key == 'Up') {

moveUp = false;

}

if (event.key == 'Down') {

moveDown = false;

}

if (isCollide(fighter,rightWall)) {

moveRight = false;

}

if (isCollide(fighter,leftWall)) {

moveLeft = false;

}

if (isCollide(fighter, boundaryWall)) {

moveUp = false;

}

if (isCollide(fighter, bottomWall)) {

moveDown = false;

}

});

onEvent("screen1", "click", function() {

shootBullet();

});

function setBulletPos() {

var bulletString;

var stringI;

var i3;

for (i3 = 0; i3 < 10; i3++) {

stringI = i3.toString();

bulletString = "bullet" + stringI;

setPosition(bulletString, bullet.X[i3], bullet.Y[i3]);

}

}

function bulletCollision() {

var i5;

for (i5 = 0; i5 < 10; i5++) {

if (bulletCollide(bullet.Y[i5], enemy)) {

enemy.Y = 0;

enemy.X = randomNumber(30, 285);

}

}

}

function bulletUpdate() {

var i1;

for (i1 = 0; i1 < 10; i1++) {

if (bullet.Y[i1] < topWall.Y) {

bullet.A[i1] = 0;

}

if (bullet.A[i1] == 1) {

bullet.Y[i1] = bullet.Y[i1] - 4;

}

setBulletPos();

}

}

//This original isCollide collision function is written by Mr. Saathoff

function isCollide(obj1, obj2) {

if ((obj1.X < obj2.X + obj2.W) && (obj1.X + obj1.W > obj2.X)) {

if ((obj1.Y < obj2.Y + obj2.H) && (obj1.Y + obj1.H > obj2.Y)) {

return(true);

}

}

}

//------------//

function bulletCollide(obj1, obj2) {

for (i = 0; i < 10; i++) {

if ((bullet.X[i] < obj2.X + obj2.W) && (bullet.X[i] + bulletInfo.W > obj2.X)) {

if ((bullet.Y[i] < obj2.Y + obj2.H) && (bullet.Y[i] + bulletInfo.H > obj2.Y)) {

return(true);

}

}

}

}

timedLoop(10, update);

timedLoop(8, enemyUpdate);