```
using System;
using System.Collections.Generic;
using System.Linq;
using Microsoft.Xna.Framework;
using Microsoft.Xna.Framework.Audio;
using Microsoft.Xna.Framework.Content;
using Microsoft.Xna.Framework.GamerServices;
using Microsoft.Xna.Framework.Graphics;
using Microsoft.Xna.Framework.Input;
using Microsoft.Xna.Framework.Media;

namespace FinalProject
{
    /// <summary>
    /// This is the main type for your game
    /// </summary>
    public class Game1 : Microsoft.Xna.Framework.Game
    {
        GraphicsDeviceManager graphics;
        SpriteBatch spriteBatch;

        KeyboardState keyboardState;

        Texture2D background;
        Rectangle backgroundRect;

        Texture2D[] playerOneDirections;
        Texture2D playerOneSprite;
        Rectangle PlayerOneRec;

        Texture2D[] playerTwoDirections;
        Texture2D playerTwoSprite;
        Rectangle PlayerTwoRec;

        Texture2D[] playerThreeDirections;
        Texture2D playerThreeSprite;
        Rectangle PlayerThreeRec;

        Texture2D[] playerFourDirections;
        Texture2D playerFourSprite;
        Rectangle PlayerFourRec;

        Vector2 onePos, twoPos, threePos, fourPos;

        SpriteFont title;

        Boolean isOneAlive = false;
        Boolean isTwoAlive = false;
        Boolean isThreeAlive = false;
        Boolean isFourAlive = false;

        Texture2D healthSprite;
        Rectangle healthRect;
        SpriteFont healthFont;
        int health;

        Texture2D speedTexture;
        List<Rectangle> speedRects;

        Texture2D healthOrbTexture;
        List<HealthOrbs> healthOrbList = new List<HealthOrbs>();
        Vector2 healthOrbPos = new Vector2(600, 300);

        Texture2D obstTexture;
        List<Rectangle> obstRects;
        Texture2D missileUp, missileDown, missileRight, missileLeft;

        List<missile> missileList;
        bool drawMissile;

        bool isMissileUpAlive1 = true;
        bool isMissileDownAlive1, isMissileRightAlive1, isMissileLeftAlive1 = false;
        bool isMissileUpAlive2 = true;
        bool isMissileDownAlive2, isMissileRightAlive2, isMissileLeftAlive2 = false;
        bool isMissileUpAlive3 = true;
        bool isMissileDownAlive3, isMissileRightAlive3, isMissileLeftAlive3 = false;
        bool isMissileUpAlive4 = true;
        bool isMissileDownAlive4, isMissileRightAlive4, isMissileLeftAlive4 = false;


        int playerOnefacing = 1;
        int playerTwofacing = 1;
        int playerThreefacing = 1;
        int playerFourfacing = 1;

        int numPlayers;

        bool chooseplayers = true;

        SpriteFont timerFont;
        float timer = 0;

        SoundEffect gun, die;

        KeyboardState tempkeyboardState;


        public Game1()
        {
            graphics = new GraphicsDeviceManager(this);
            graphics.PreferredBackBufferWidth = 1100;
            graphics.PreferredBackBufferHeight = 660;
            Content.RootDirectory = "Content";
        }


        protected override void Initialize()
        {


            base.Initialize();
        }


        protected override void LoadContent()
        {
            // Create a new SpriteBatch, which can be used to draw textures.
            spriteBatch = new SpriteBatch(GraphicsDevice);

            backgroundRect = new Rectangle(0, 0, 1100, 660);
            background = Content.Load<Texture2D>("jungle background");

            healthOrbTexture = Content.Load<Texture2D>("OrbRed");

            health = 10;
            healthSprite = Content.Load<Texture2D>("Pixel");
            healthRect = new Rectangle(0, 0, health * 2, 25);
            healthFont = Content.Load<SpriteFont>("Impact");


            speedTexture = Content.Load<Texture2D>("back");
            speedRects = new List<Rectangle>();

            obstTexture = Content.Load<Texture2D>("BrickWallCAG");
            obstRects = new List<Rectangle>();

            missileList = new List<missile>();

            Random random = new Random();
            for (int i = 0; i < 25; i++)
            {
                obstRects.Add(new Rectangle(random.Next(0, 1100), random.Next(0, 660), obstTexture.Width / ↙
    4, obstTexture.Height / 4));
            }

            title = Content.Load<SpriteFont>("Title");

            //missiles
            missileUp = Content.Load<Texture2D>("bullet 1");
            missileDown = Content.Load<Texture2D>("bullet 3");
            missileRight = Content.Load<Texture2D>("bullet 2");
            missileLeft = Content.Load<Texture2D>("bullet 4");

            playerOneDirections = new Texture2D[4];
            playerOneDirections[0] = Content.Load<Texture2D>("soldier red back");
            playerOneDirections[1] = Content.Load<Texture2D>("soldier red front");
            playerOneDirections[2] = Content.Load<Texture2D>("soldier red left");
            playerOneDirections[3] = Content.Load<Texture2D>("soldier red right");
            playerOneSprite = playerOneDirections[1];
            PlayerOneRec = new Rectangle((int)onePos.X, (int)onePos.Y, playerOneSprite.Width,          ↙
    playerOneSprite.Height);


            playerTwoDirections = new Texture2D[4];
            playerTwoDirections[0] = Content.Load<Texture2D>("back");
            playerTwoDirections[1] = Content.Load<Texture2D>("front");
            playerTwoDirections[2] = Content.Load<Texture2D>("left");
            playerTwoDirections[3] = Content.Load<Texture2D>("right");
            playerTwoSprite = playerTwoDirections[1];
            PlayerTwoRec = new Rectangle((int)twoPos.X, (int)twoPos.Y, playerTwoSprite.Width,          ↙
    playerTwoSprite.Height);

            playerThreeDirections = new Texture2D[4];
            playerThreeDirections[0] = Content.Load<Texture2D>("soldier blue back");
            playerThreeDirections[1] = Content.Load<Texture2D>("soldier blue front");
            playerThreeDirections[2] = Content.Load<Texture2D>("soldier blue left");
            playerThreeDirections[3] = Content.Load<Texture2D>("soldier blue right");
            playerThreeSprite = playerThreeDirections[1];
            PlayerThreeRec = new Rectangle((int)threePos.X, (int)threePos.Y, playerThreeSprite.Width,  ↙
    playerThreeSprite.Height);

            playerFourDirections = new Texture2D[4];
            playerFourDirections[0] = Content.Load<Texture2D>("soldier orange back");
            playerFourDirections[1] = Content.Load<Texture2D>("soldier orange front");
            playerFourDirections[2] = Content.Load<Texture2D>("soldier orange left");
            playerFourDirections[3] = Content.Load<Texture2D>("soldier orange right");
            playerFourSprite = playerFourDirections[1];
            PlayerFourRec = new Rectangle((int)fourPos.X, (int)fourPos.Y, playerFourSprite.Width,      ↙
    playerFourSprite.Height);

            //New Vector2 Palyer Posisitions !!!!!!!!!!!
            onePos = new Vector2(50, 50);
            twoPos = new Vector2(graphics.GraphicsDevice.Viewport.Width - 50, 50);
            threePos = new Vector2(50, graphics.GraphicsDevice.Viewport.Height - 50);
            fourPos = new Vector2(graphics.GraphicsDevice.Viewport.Width - 50, graphics.GraphicsDevice. ↙
    Viewport.Height - 50);
```

```
            //timer stuff
            timerFont = Content.Load<SpriteFont>("timerFont");

            die = Content.Load<SoundEffect>("HealthLossSound");
            gun  = Content.Load<SoundEffect>("GunSound");

        }

        /// <summary>
        /// UnloadContent will be called once per game and is the place to unload
        /// all content.
        /// </summary>
        protected override void UnloadContent()
        {
            // TODO: Unload any non ContentManager content here
        }

        private void PlayerOneMovement()
        {
            float tempX = onePos.X;
            float tempY = onePos.Y;


            if (Keyboard.GetState().IsKeyDown(Keys.W) && tempY > 0)
            {
                tempY -= 3;
                playerOneSprite = playerOneDirections[0];
                playerOnefacing = 1;
            }
            else if (Keyboard.GetState().IsKeyDown(Keys.S) && (tempY + PlayerOneRec.Height) < 660)
            {
                tempY += 3;
                playerOneSprite = playerOneDirections[1];
                playerOnefacing = 2;
            }
            else if (Keyboard.GetState().IsKeyDown(Keys.A) && tempX > 0)
            {
                tempX -= 3;
                playerOneSprite = playerOneDirections[2];
                playerOnefacing = 3;
            }
            else if (Keyboard.GetState().IsKeyDown(Keys.D) && (tempX + PlayerOneRec.Width) < 1100)
            {
                tempX += 3;
                playerOneSprite = playerOneDirections[3];
                playerOnefacing = 4;
            }
            Rectangle temprec = new Rectangle((int)tempX, (int)tempY, playerOneSprite.Width,      ↙
    playerOneSprite.Height);
            if (!WillICollide(temprec))
            {
                onePos.X = tempX;
                onePos.Y = tempY;
                PlayerOneRec = temprec;
            }
        }

        private void PlayerTwoMovement()
        {
            float tempX = twoPos.X;
            float tempY = twoPos.Y;

            if (Keyboard.GetState().IsKeyDown(Keys.I) && tempY > 0)
            {
                tempY -= 3;
                playerTwoSprite = playerTwoDirections[0];
                playerTwofacing = 1;
            }
            else if (Keyboard.GetState().IsKeyDown(Keys.K) && (tempY + PlayerTwoRec.Height) < 660)
            {
                tempY += 3;
                playerTwoSprite = playerTwoDirections[1];
                playerTwofacing = 2;
            }
            else if (Keyboard.GetState().IsKeyDown(Keys.J) && tempX > 0)
            {
                tempX -= 3;
                playerTwoSprite = playerTwoDirections[2];
                playerTwofacing = 3;
            }
            else if (Keyboard.GetState().IsKeyDown(Keys.L) && (tempX + PlayerTwoRec.Width) < 1100)
            {
                tempX += 3;
                playerTwoSprite = playerTwoDirections[3];
                playerTwofacing = 4;
            }
            Rectangle temprec = new Rectangle((int)tempX, (int)tempY, playerTwoSprite.Width,        ↙
    playerTwoSprite.Height);
            if (!WillICollide(temprec))
            {
                twoPos.X = tempX;
                twoPos.Y = tempY;
                PlayerTwoRec = temprec;
            }
        }

        private void PlayerThreeMovement()
        {
            float tempX = threePos.X;
            float tempY = threePos.Y;

            if (Keyboard.GetState().IsKeyDown(Keys.Up) && tempY > 0)
            {
                tempY -= 3;
                playerThreeSprite = playerThreeDirections[0];
                playerThreefacing = 1;
            }
            else if (Keyboard.GetState().IsKeyDown(Keys.Down) && (tempY + PlayerThreeRec.Height) < 660)
            {
                tempY += 3;
                playerThreeSprite = playerThreeDirections[1];
                playerThreefacing = 2;
            }
            else if (Keyboard.GetState().IsKeyDown(Keys.Left) && tempX > 0)
            {
                tempX -= 3;
                playerThreeSprite = playerThreeDirections[2];
                playerThreefacing = 3;
            }
            else if (Keyboard.GetState().IsKeyDown(Keys.Right) && (tempX + PlayerThreeRec.Width) < 1100)
            {
                tempX += 3;
                playerThreeSprite = playerThreeDirections[3];
                playerThreefacing = 4;
            }

            Rectangle temprec = new Rectangle((int)tempX, (int)tempY, playerThreeSprite.Width,      ↙
    playerThreeSprite.Height);
            if (!WillICollide(temprec))
            {
                threePos.X = tempX;
                threePos.Y = tempY;
                PlayerThreeRec = temprec;
            }
        }

        private void PlayerFourMovement()
        {
            float tempX = fourPos.X;
            float tempY = fourPos.Y;


            if (Keyboard.GetState().IsKeyDown(Keys.NumPad8) && tempY > 0)
            {
                tempY -= 3;
                playerFourSprite = playerFourDirections[0];
                playerFourfacing = 1;
            }
            else if (Keyboard.GetState().IsKeyDown(Keys.NumPad5) && (tempY + PlayerFourRec.Height) < 660)
            {
                tempY += 3;
                playerFourSprite = playerFourDirections[1];
                playerFourfacing = 2;
            }
            else if (Keyboard.GetState().IsKeyDown(Keys.NumPad4) && tempX > 0)
            {
                tempX -= 3;
                playerFourSprite = playerFourDirections[2];
                playerFourfacing = 3;
            }
            else if (Keyboard.GetState().IsKeyDown(Keys.NumPad6) && (tempX + PlayerFourRec.Width) < 1100)
            {
                tempX += 3;
                playerFourSprite = playerFourDirections[3];
                playerFourfacing = 4;
            }

            //PlayerFourRec = new Rectangle ((int)fourPos.X, (int)fourPos.Y, playerFourSprite.Width,       ↙
    playerFourSprite.Height);

            Rectangle temprec = new Rectangle((int)tempX, (int)tempY, playerFourSprite.Width,            ↙
    playerFourSprite.Height);
            if (!WillICollide(temprec))
            {
                fourPos.X = tempX;
                fourPos.Y = tempY;
                PlayerFourRec = temprec;
            }
        }


        private void MissileAndWallCollision()
        {

            foreach (missile missile in missileList)
            {
                if (missile.getVisable())
                {

                    for (int o = 0; o < obstRects.Count(); o++)
                    {
                        if (missile.getMissileRec().Intersects(obstRects[o]))
                        {
                            missile.setVisable(false);
                            break;
                        }
                    }
                }
            }

        }



        private void MissileAndPlayerCollision()
        {

            foreach (missile missile in missileList)
            {
                if (missile.getVisable())
                {
                        if (PlayerOneRec.Intersects(missile.getMissileRec()) )
                        {
                            isOneAlive = false;
                            die.Play();
                        }
                        if (PlayerTwoRec.Intersects(missile.getMissileRec()))
                        {
                            isTwoAlive = false;
                            die.Play();
                        }
                        if (PlayerThreeRec.Intersects(missile.getMissileRec()))
                        {
                            isThreeAlive = false;
                            die.Play();
                        }
                        if (PlayerFourRec.Intersects(missile.getMissileRec()))
                        {
                            isFourAlive = false;
                            die.Play();
                        }
                }
            }
        }



        //private void PlayerAndWallCollision()
        //{
        //    for (int i = 0; i < obstRects.Count(); i++)
        //        if (playerOneRec.Intersects(obstRects[i]))
        //        {

        //        }
        //}

        private bool WillICollide(Rectangle rec)
        {

            for (int i = 0; i < obstRects.Count(); i++)
                if (rec.Intersects(obstRects[i]))
                {
                    return true;
                }
            return false;
        }

        private void ChoosePlayers()
        {
            if (chooseplayers)
            {

                if (keyboardState.IsKeyDown(Keys.F2))
                {
                    numPlayers = 2;
                    isOneAlive = true;
                    isTwoAlive = true;
                    chooseplayers = false;
                }
                if (keyboardState.IsKeyDown(Keys.F3))
                {
                    numPlayers = 3;
                    isOneAlive = true;
                    isTwoAlive = true;
                    isThreeAlive = true;
                    chooseplayers = false;
                }
                if (keyboardState.IsKeyDown(Keys.F4))
                {
                    numPlayers = 4;
                    isOneAlive = true;
                    isTwoAlive = true;
                    isThreeAlive = true;
                    isFourAlive = true;
                    chooseplayers = false;
                }
            }
        }
        private void Reset()
        {
            timer = (float)0.00;
            chooseplayers = true;
            isOneAlive = false;
            isTwoAlive = false;
            isThreeAlive = false;
            isFourAlive = false;
        }

        private void HealthPlacememnt(float time)
        {
            Random rand = new Random();
            int randX = rand.Next(100, 800);
            int randY = rand.Next(100, 800);
            HealthOrbs healthOrb = new HealthOrbs(healthOrbTexture, healthOrbPos);

            if (time % 5 == 0.0 && time != 0)
            {
                healthOrbList.Add(new HealthOrbs(healthOrbTexture, new Vector2(randX, randY)));
            }
        }

        private void CleanUpMissile()
        {

          //  foreach (missile missile in missileList)
            for (int i = 0; i < missileList.Count ; i ++  )
            {
                if ((missileList[i].getMissileRec().X > 1100 || missileList[i].getMissileRec().Y > 660 ||   ↙
    missileList[i].getMissileRec().X < 0 || missileList[i].getMissileRec().Y < 0) ||
                    (missileList[i].getVisable() == false))
                {

                    missileList.Remove(missileList[i]);
                }
            }
        }

        private void OneMissileMaker()
        {
            //missile update
            keyboardState = Keyboard.GetState();
            if ((keyboardState.IsKeyDown(Keys.LeftControl) && tempkeyboardState.IsKeyUp(Keys.LeftControl))  ↙
    && fourPos.Y > 0)
            {

                missile M = new missile();

                M.setDirection(playerOnefacing);
                M.setSpeed(7);
                //M.setPosition(onePos);

                if (playerOnefacing == 1 && isOneAlive)
                {
                    //missileRec = new Rectangle((int)onePos.X, (int)onePos.Y -15, missileUp.Width,         ↙
    missileUp.Height);
                    //drawMissile = true;
                    M.setMissileSprite(missileUp);
                    M.setPosition(new Vector2((int)onePos.X, (int)onePos.Y - 15));

                }
                if (playerOnefacing == 2 && isOneAlive)
                {
                    //missileRec = new Rectangle((int)onePos.X, (int)onePos.Y + 25 , missileUp.Width,       ↙
    missileUp.Height);
                    //drawMissile = true;
                    M.setMissileSprite(missileDown);
                    M.setPosition(new Vector2((int)onePos.X, (int)onePos.Y + 25));
                }
                if (playerOnefacing == 3 && isOneAlive)
                {
                    //missileRec = new Rectangle((int)onePos.X - 25, (int)onePos.Y, missileUp.Width,        ↙
    missileUp.Height);
                    //drawMissile = true;
                    M.setMissileSprite(missileLeft);
                    M.setPosition(new Vector2((int)onePos.X - 15, (int)onePos.Y));
                }
                if (playerOnefacing == 4 && isOneAlive)
                {
                    //missileRec = new Rectangle((int)onePos.X + 25, (int)onePos.Y, missileUp.Width,        ↙
    missileUp.Height);
                    //drawMissile = true;
                    M.setMissileSprite(missileRight);
                    M.setPosition(new Vector2((int)onePos.X + 15, (int)onePos.Y));
                }

                missileList.Add(M); //ads each missile to missile list

                keyboardState = tempkeyboardState;
            }

        }
        private void TwoMissileMaker()
        {
            //missile update
            if (Keyboard.GetState().IsKeyDown(Keys.Space) && twoPos.Y > 0)
            {

                missile M = new missile();

                M.setDirection(playerTwofacing);
                M.setSpeed(7);
                //M.setPosition(onePos);

                if (playerTwofacing == 1 && isTwoAlive)
                {
                    //missileRec = new Rectangle((int)onePos.X, (int)onePos.Y -15, missileUp.Width,         ↙
    missileUp.Height);
                    //drawMissile = true;
                    M.setMissileSprite(missileUp);
                    M.setPosition(new Vector2((int)twoPos.X, (int)twoPos.Y - 15));

                }
                if (playerTwofacing == 2 && isTwoAlive)
                {
                    //missileRec = new Rectangle((int)onePos.X, (int)onePos.Y + 25 , missileUp.Width,      ↙
    missileUp.Height);
                    //drawMissile = true;
                    M.setMissileSprite(missileDown);
                    M.setPosition(new Vector2((int)twoPos.X, (int)twoPos.Y + 25));
                }
                if (playerTwofacing == 3 && isTwoAlive)
                {
                    //missileRec = new Rectangle((int)onePos.X - 25, (int)onePos.Y, missileUp.Width,       ↙
    missileUp.Height);
                    //drawMissile = true;
                    M.setMissileSprite(missileLeft);
                    M.setPosition(new Vector2((int)twoPos.X - 15, (int)twoPos.Y));
                }
                if (playerTwofacing == 4 && isTwoAlive)
                {
                    //missileRec = new Rectangle((int)onePos.X + 25, (int)onePos.Y, missileUp.Width,       ↙
    missileUp.Height);
                    //drawMissile = true;
                    M.setMissileSprite(missileRight);
                    M.setPosition(new Vector2((int)twoPos.X + 15, (int)twoPos.Y));
                }

                missileList.Add(M); //ads each missile to missile list

            }
        }
        private void ThreeMissileMaker()
        {
            //missile update
            if (Keyboard.GetState().IsKeyDown(Keys.RightControl ) && threePos.Y > 0)
            {

                missile M = new missile();

                M.setDirection(playerThreefacing );
                M.setSpeed(7);
                //M.setPosition(onePos);

                if (playerThreefacing  == 1 && isThreeAlive )
                {
                    //missileRec = new Rectangle((int)onePos.X, (int)onePos.Y -15, missileUp.Width,        ↙
    missileUp.Height);
                    //drawMissile = true;
                    M.setMissileSprite(missileUp);
                    M.setPosition(new Vector2((int)threePos.X, (int)threePos.Y - 15));

                }
                if (playerThreefacing  == 2 && isThreeAlive )
                {
                    //missileRec = new Rectangle((int)onePos.X, (int)onePos.Y + 25 , missileUp.Width,      ↙
    missileUp.Height);
                    //drawMissile = true;
                    M.setMissileSprite(missileDown);
                    M.setPosition(new Vector2((int)threePos.X, (int)threePos.Y + 25));
                }
                if (playerThreefacing == 3 && isThreeAlive )
                {
                    //missileRec = new Rectangle((int)onePos.X - 25, (int)onePos.Y, missileUp.Width,        ↙
    missileUp.Height);
                    //drawMissile = true;
                    M.setMissileSprite(missileLeft);
                    M.setPosition(new Vector2((int)threePos.X - 15, (int)threePos.Y));
                }
                if (playerThreefacing  == 4 && isThreeAlive )
                {
                    //missileRec = new Rectangle((int)onePos.X + 25, (int)onePos.Y, missileUp.Width,        ↙
    missileUp.Height);
                    //drawMissile = true;
                    M.setMissileSprite(missileRight);
                    M.setPosition(new Vector2((int)threePos.X + 15, (int)threePos.Y));
                }

                missileList.Add(M); //ads each missile to missile list

            }
        }
        private void FourMissileMaker()
        {
            //missile update
            if (Keyboard.GetState().IsKeyDown(Keys.NumPad0 ) && twoPos.Y > 0)
            {

                missile M = new missile();

                M.setDirection(playerFourfacing );
                M.setSpeed(7);
                //M.setPosition(onePos);

                if (playerFourfacing  == 1 && isFourAlive )
                {
                    //missileRec = new Rectangle((int)onePos.X, (int)onePos.Y -15, missileUp.Width,        ↙
    missileUp.Height);
                    //drawMissile = true;
                    M.setMissileSprite(missileUp);
                    M.setPosition(new Vector2((int)fourPos.X, (int)fourPos.Y - 15));

                }
                if (playerFourfacing  == 2 && isFourAlive )
                {
                    //missileRec = new Rectangle((int)onePos.X, (int)onePos.Y + 25 , missileUp.Width,        ↙
    missileUp.Height);
                    //drawMissile = true;
                    M.setMissileSprite(missileDown);
                    M.setPosition(new Vector2((int)fourPos.X, (int)fourPos.Y + 25));
                }
                if (playerFourfacing  == 3 && isFourAlive )
                {
                    //missileRec = new Rectangle((int)onePos.X - 25, (int)onePos.Y, missileUp.Width,        ↙
    missileUp.Height);
                    //drawMissile = true;
                    M.setMissileSprite(missileLeft);
                    M.setPosition(new Vector2((int)fourPos.X - 15, (int)fourPos.Y));
                }
                if (playerFourfacing  == 4 && isFourAlive )
                {
                    //missileRec = new Rectangle((int)onePos.X + 25, (int)onePos.Y, missileUp.Width,        ↙
    missileUp.Height);
                    //drawMissile = true;
                    M.setMissileSprite(missileRight);
                    M.setPosition(new Vector2((int)fourPos.X + 15, (int) fourPos.Y));
                }

                missileList.Add(M); //ads each missile to missile list
            }
        }

        private void OneMissileChecker()
            {
            //missile update
            if (Keyboard.GetState().IsKeyDown(Keys.LeftControl) && fourPos.Y > 0)
            {
                if (playerOnefacing == 1)
                {
                    isMissileUpAlive1 = true;
                }
                if (playerOnefacing == 2)
                {
                    isMissileDownAlive1 = true;
                }
                if (playerOnefacing == 3)
                {
                    isMissileLeftAlive1 = true;
                }
                if (playerOnefacing == 4)
                {
                    isMissileRightAlive1 = true;
                }
            }
            }
        private void TwoMissileChecker()
        {
            //missile update
            if (Keyboard.GetState().IsKeyDown(Keys.Space ) && twoPos.Y > 0)
            {
                if (playerTwofacing  == 1)
                {
                    isMissileUpAlive2 = true;
                }
                if (playerTwofacing  == 2)
                {
                    isMissileDownAlive2 = true;
                }
                if (playerTwofacing  == 3)
                {
                    isMissileLeftAlive2 = true;
                }
                if (playerTwofacing  == 4)
                {
                    isMissileRightAlive2 = true;
                }
            }
            }
        private void ThreeMissileChecker()
        {
            //missile update
            if (Keyboard.GetState().IsKeyDown(Keys.RightControl  ) && threePos.Y > 0)
            {
                if (playerThreefacing   == 1)
                {
                    isMissileUpAlive3 = true;
                }
                if (playerThreefacing   == 2)
                {
                    isMissileDownAlive3 = true;
                }
                if (playerThreefacing   == 3)
                {
                    isMissileLeftAlive3 = true;
                }
                if (playerThreefacing   == 4)
                {
                    isMissileRightAlive3 = true;
                }
            }
            }
        private void FourMissileChecker()
        {
            //missile update
            if (Keyboard.GetState().IsKeyDown(Keys.NumPad0  ) && twoPos.Y > 0)
            {
                if (playerFourfacing   == 1)
                {
                    isMissileUpAlive4 = true;
                }
                if (playerFourfacing   == 2)
                {
                    isMissileDownAlive4 = true;
                }
                if (playerFourfacing   == 3)
                {
                    isMissileLeftAlive4 = true;
                }
                if (playerFourfacing   == 4)
                {
                    isMissileRightAlive4 = true;
                }
            }

        }

        private void OneMissileDirection()
        {
            foreach (missile M in missileList)
            {

                if (M.getVisable())
                {
                    if (M.getDirection() == 1 && isMissileUpAlive1)
                    {
                        M.setPosition(new Vector2(M.GetPosition().X, M.GetPosition().Y - 5));
                    }
                    if (M.getDirection() == 2 && isMissileDownAlive1)
                    {
                        M.setPosition(new Vector2(M.GetPosition().X, M.GetPosition().Y + 5));

                    }
                    if (M.getDirection() == 3 && isMissileLeftAlive1)
                    {
                        M.setPosition(new Vector2(M.GetPosition().X - 5, M.GetPosition().Y));
                    }
                    if (M.getDirection() == 4 && isMissileRightAlive1)
                    {
                        M.setPosition(new Vector2(M.GetPosition().X + 5, M.GetPosition().Y));
                    }
                }
            }
        }
        private void TwoMissileDirection()
        {
            foreach (missile M in missileList)
            {

                if (M.getVisable())
                {
```

```
                if (M.getDirection() == 1 && isMissileUpAlive2)
                {
                    M.setPosition(new Vector2(M.GetPosition().X, M.GetPosition().Y - 5));
                }
                if (M.getDirection() == 2 && isMissileDownAlive2)
                {
                    M.setPosition(new Vector2(M.GetPosition().X, M.GetPosition().Y + 5));

                }
                if (M.getDirection() == 3 && isMissileLeftAlive2)
                {
                    M.setPosition(new Vector2(M.GetPosition().X - 5, M.GetPosition().Y));
                }
                if (M.getDirection() == 4 && isMissileRightAlive2)
                {
                    M.setPosition(new Vector2(M.GetPosition().X + 5, M.GetPosition().Y));
                }
            }
        }
    }
    private void ThreeMissileDirection()
    {
        foreach (missile M in missileList)
        {

            if (M.getVisable())
            {
                if (M.getDirection() == 1 && isMissileUpAlive3)
                {
                    M.setPosition(new Vector2(M.GetPosition().X, M.GetPosition().Y - 5));
                }
                if (M.getDirection() == 2 && isMissileDownAlive3)
                {
                    M.setPosition(new Vector2(M.GetPosition().X, M.GetPosition().Y + 5));

                }
                if (M.getDirection() == 3 && isMissileLeftAlive3)
                {
                    M.setPosition(new Vector2(M.GetPosition().X - 5, M.GetPosition().Y));
                }
                if (M.getDirection() == 4 && isMissileRightAlive3)
                {
                    M.setPosition(new Vector2(M.GetPosition().X + 5, M.GetPosition().Y));
                }
            }
        }
    }
    private void FourMissileDirection()
    {
        foreach (missile M in missileList)
        {

            if (M.getVisable())
            {
                if (M.getDirection() == 1 && isMissileUpAlive4)
                {
                    M.setPosition(new Vector2(M.GetPosition().X, M.GetPosition().Y - 5));
                }
                if (M.getDirection() == 2 && isMissileDownAlive4)
                {
                    M.setPosition(new Vector2(M.GetPosition().X, M.GetPosition().Y + 5));

                }
                if (M.getDirection() == 3 && isMissileLeftAlive4)
                {
                    M.setPosition(new Vector2(M.GetPosition().X - 5, M.GetPosition().Y));
                }
                if (M.getDirection() == 4 && isMissileRightAlive4)
                {
                    M.setPosition(new Vector2(M.GetPosition().X + 5, M.GetPosition().Y));
                }
            }
        }
    }

    protected override void Update(GameTime gameTime)
    {
        // Allows the game to exit
        if (GamePad.GetState(PlayerIndex.One).Buttons.Back == ButtonState.Pressed)
            this.Exit();

        keyboardState = Keyboard.GetState();

        if (keyboardState.IsKeyDown(Keys.Escape))
        Reset();

        ChoosePlayers();
        MissileAndWallCollision();
        CleanUpMissile();
        MissileAndPlayerCollision();

        if (isOneAlive)
        {
            OneMissileMaker();
            OneMissileChecker();
            OneMissileDirection();
        }

        if (isTwoAlive)
        {
            TwoMissileMaker();
            TwoMissileChecker();
            TwoMissileDirection();
        }
        if (isThreeAlive)
        {
            ThreeMissileMaker();
            ThreeMissileChecker();
            ThreeMissileDirection();
        }

        if(isFourAlive )
        {
            FourMissileMaker();
            FourMissileChecker();
            FourMissileDirection();
        }


        //movement
        if (isOneAlive)
        {
            PlayerOneMovement();
        }
        if (isTwoAlive)
        {
            PlayerTwoMovement();
        }
        if (isThreeAlive)
        {
            PlayerThreeMovement();
        }
```

```
            if (isFourAlive)
            {
                PlayerFourMovement();
            }

            //PlayerAndWallCollision();
            HealthPlacememnt(timer);


            //timer code
            if (!chooseplayers)
            {
                timer += (float)gameTime.ElapsedGameTime.TotalSeconds;
            }

            base.Update(gameTime);

        }

        /// <summary>
        /// This is called when the game should draw itself.
        /// </summary>
        /// <param name="gameTime">Provides a snapshot of timing values.</param>


        private void StartScreen()
        {
            spriteBatch.Draw(background, backgroundRect, Color.White);
            spriteBatch.DrawString(title, "Survival Island", new Vector2(283, 230), Color.Black);
            spriteBatch.DrawString(timerFont, "F2 = 2 players \nF3 = 3 players \nF4 = 4 players", new ↙
    Vector2(300, 330), Color.Black);
        }

        protected override void Draw(GameTime gameTime)
        {
            GraphicsDevice.Clear(Color.CornflowerBlue);

            spriteBatch.Begin();
            StartScreen();

            if (!chooseplayers)
            {
                spriteBatch.Draw(background, backgroundRect, Color.White);
                if (isOneAlive)
                {
                    spriteBatch.Draw(playerOneSprite, onePos, Color.White);
                }
                if (isTwoAlive)
                {
                    spriteBatch.Draw(playerTwoSprite, twoPos, Color.White);
                }
                if (isThreeAlive)
                {
                    spriteBatch.Draw(playerThreeSprite, threePos, Color.White);
                }
                if (isFourAlive)
                {
                    spriteBatch.Draw(playerFourSprite, fourPos, Color.White);
                }

                foreach (Rectangle rect in obstRects)
                    spriteBatch.Draw(obstTexture, rect, Color.White);


                //spriteBatch.Draw(healthSprite, healthRect, Color.Red);
                //foreach (HealthOrbs healthorb in healthOrbList)
                //{

                //    healthorb.Draw(spriteBatch);
                //}


                //timer drawing
                spriteBatch.DrawString(timerFont, "The time is: " + timer.ToString("0.00"), new Vector2(500 ↙
    , 0), Color.Black);
            }



            if (missileList != null)
            {

                foreach (missile M in missileList)
                {

                    if (M.getVisable())
                    {

                        if (M.getDirection() == 1)
                        {
                            spriteBatch.Draw(missileUp, new Rectangle((int)M.GetPosition().X, (int)M.      ↙
    GetPosition().Y, missileUp.Width, missileUp.Height), Color.White);
                        }
                        if (M.getDirection() == 2)
                        {
                            spriteBatch.Draw(missileDown, new Rectangle((int)M.GetPosition().X, (int)M.    ↙
    GetPosition().Y, missileDown.Width, missileDown.Height), Color.White);
                        }
                        if (M.getDirection() == 3)
                        {
                            spriteBatch.Draw(missileLeft, new Rectangle((int)M.GetPosition().X, (int)M.    ↙
    GetPosition().Y, missileLeft.Width, missileLeft.Height), Color.White);
                        }
                        if (M.getDirection() == 4)
                        {
                            spriteBatch.Draw(missileRight, new Rectangle((int)M.GetPosition().X, (int)M.   ↙
    GetPosition().Y, missileRight.Width, missileRight.Height), Color.White);
                        }
                    }


                }
            }





            spriteBatch.End();



            base.Draw(gameTime);
        }
    }
}
```