5 pts

# Chess Board

1. Initialize 32 Rectangles for all chess pieces and 1 for board and put them in a Rectangle[]
2. Load 12 Texture2Ds for all chess pieces and 1 for board
3. If escape is pressed during Update, this.Exit()
4. Loop through all Rectangles and Draw the appropriate Texture2D

# Cards

1. Create a Card class
   1. Set up a constructor that takes in a Texture2D and Rectangle
   2. Have a Draw method to make the piece
2. Initialize 2 Cards through the constructor
3. Draw cards using Rectangles and Texture2Ds

# War

1. Create a Card class
   1. Set up constructor that takes in a Texture2D, Rectangle, and values of card value and suit
   2. Set up Draw method for the card
2. Initialize 52 Cards through the constructor and store it in a Card[]
3. Use the provided Shuffle method to rearrange
4. Draw the first 2 Cards in the List onto the screen
5. With a SpriteText, DrawString the Card suits and values

# Star Field

1. Create a Star class
   1. Set up constructor that takes in a Texture2D, Rectangle, xSpeed, ySpeed, and Color
   2. Set up Draw method for the star
   3. Set up Update method for the star to make it move on each iteration
2. Initialize 5 Stars through the constructor with random parameters and store them in a Star[]
3. Update and Draw all of them in a loop and check if they are out of bounds
4. If they are out of bounds, initialize a new random Star

# 

# Space Invaders #1

1. Create an Alien class
   1. Set up constructor that takes in Texture2D and Rectangle
   2. Make a static setSpeed method to change direction of movement
   3. Make an Update method to make Aliens move on each iteration
   4. Make a moveDown method to change all Aliens Rectangle.Y
   5. Make a Draw method to draw the Alien
2. Initialize 10 Aliens in a Alien[] through a loop
3. Set up Alien speed
4. Draw and Update all Aliens
5. When left Alien reaches left edge, setSpeed in opposite direction and moveDown
6. When right Alien reaches right edge, setSpeed in opposite direction and moveDown
7. When Aliens reach bottom edge, only setSpeed in opposite direction when Aliens reach left or right edge

10 pts

# Death Blossom

1. Create a Missile class
   1. Set up constructor that takes in a Rectangle, ints for speed and angle, and Texture2D
   2. Make an Update method that uses speed and angle to move Missile
   3. Make a Draw method that draws Missile with correct orientation
2. Initialize KeyboardState, List<Missile>, and Missile Texture2D
3. Check if space has been pressed and make Gunstar firing true
4. Create and add Missile to List<Missile> for each game Update
5. Update every Missile in List<Missile>
6. If Missile is out of the screen, remove it from List<Missile>
7. Draw all Missile in List<Missile>

# Popper

1. Check for screen end bounds and change velocities for each kernel accordingly
2. Initialize int of timers
3. In nested for loops
   1. Check if a kernel Intersects with another kernel and both their images are unpoppedTex and both timers are 0
   2. If true, change images to poppedTex and set timers for 45 sec
4. Create a reverse loop
   1. If timer of any kernel == 1
      1. Remove kernel from all lists - timers, velocities, images, kernels
   2. If timer of kernel > 1
      1. timer--
5. Initialize Random and int genTimer
6. Use Random to make an int spawn time between 1 and 5 sec (avg. 3)
7. When int spawn time == 0
   1. Use Random to find x and y for kernel Rectangle in kernels
   2. Use Random to find xSpeed and ySpeed of the new kernel and put it in velocities
   3. Set images to unpoppedTex

# Bread Crumbs

1. Initialize Rectangle[], Texture2D, and Vector2 speed for all Bread Crumbs
2. Initialize and use Random to set ints xSpeed and ySpeed for Vector2 speed
3. Initialize 11 Rectangles in Rectangle[] where each is smaller than the other
4. Update all Rectangles in Rectangle[] by making them move using speed
5. If Rectangles in Rectangle[] hit walls, make them rebound
6. Draw all Rectangles in Rectangle[]

30 pts

# Light Cycle

1. Create a Cycle class
   1. Set up constructor for List<Rectangle> for walls and ints speed and angle and Rectangle for the cycle
   2. Make addTexture method to add Texture2D for cycle and wall
   3. Make turnRight and turnLeft method to change cycle direction and angle
      1. If right, add PI/2 radians and change speed as needed
      2. If left, subtract PI/2 radians and change speed as needed
   4. Make Update method to move cycle and create Rectangle wall behind and add it to List<Rectangle>
   5. Draw all List<Rectangle> and Cycle (rotate based on direction)
2. Initialize 4 Rectangles for window borders, Random, 2 Cycles, and int scene
3. Load needed Texture2D - wall, cycle, loading screen, game background, etc.
4. Create conditional using int scene in Update
   1. If scene == 0
      1. Wait till space is pressed to go to scene = 1
   2. If scene == 1
      1. Goes to scene == 2 after 180 s
   3. If scene == 2
      1. If A/D is pressed, move Cycle 1 with turnLeft/turnRight
      2. If Left/Right Arrow is pressed, move Cycle turnLeft/turnRight
      3. If Cycle 1 or Cycle 2 Intersect with window border Rectangles, List<Rectangle> walls, or each other, game ends
      4. Wait till the game ends for scene = 3
   4. If scene == 3
      1. Wait till space is pressed to Initialize again
5. Create conditional using int scene in Draw
   1. If scene == 0
      1. Draw loading screen and message “Press Space”
   2. If scene == 1
      1. Draw message “Game starting in 3… 2… 1…” using gameTime over 180 sec
   3. If scene == 2
      1. Draw both Cycles, wall border, and background Texture2D
   4. If scene == 3
      1. Draw message “Player 1 or Player 2 won” and “Press Space”