5 pts

# Bigger

1. Allow screen resizing
2. Initialize old and new KeyboardState
3. If 1 is pressed, change screen dimensions to 400, 200
4. If 2 is pressed, change screen dimensions to 600, 400
5. If 3 is pressed, change screen dimensions to fullscreen
6. old = new KeyboardState
7. If escape is pressed, this.Exit()

# Just a Piece

1. Initialize old and new KeyboardState
2. Initialize Rectangle for displayed piece
3. Initialize Rectangle[ ] for source Rectangles in sprite sheet
4. Initialize int for current index in source Rectangle[ ]
5. Load Texture2D for entire sprite sheet
6. If right is pressed, current++ % 5
7. If left is pressed and current > 0, current-- or current = 4
8. old = new KeyboardState
9. Draw displayed piece using display rectangle and source rectangle [current]

# Stepper

1. Initialize old and new KeyboardState
2. Initialize Rectangle to display object
3. Initialize int for rotation angle
4. Load Texture2D for object
5. If right is pressed, add Pi/4 radians to rotation
6. If left is pressed add Pi/4 radians to rotation
7. old = new KeyboardState
8. Draw object and use Texture2D’s center as origin

# Just a Second

1. Initialize Rectangle for clock hand
2. Load Texture2D with second hand of clock
3. Initialize int for rotation angle and update Pi/180 radians every loop
4. Draw object and use Texture2D’s center as origin

# Sidekick

1. Initialize Rectangles for three objects - stationary in middle of screen with Width/2 and Height/2, one moving with low Y value and one with coincing Y value
2. Initialize Colors for both moving objects
3. Load three Texture2Ds for all objects
4. Change Color to red if X value of either object falls within Rectangle.X or Rectangle.X + Rectangle.Width of stationary object
5. Move both Rectangles.X++
6. Draw stationary object
7. Draw objects with respective colors

10 pts

# Pick a Peck

1. Initialize old and new KeyboardState
2. Initialize Rectangle [ ] for menu option display
3. Initialize Rectangle [ ] for sources within sprite sheet
4. Initialize int for menu option selected
5. Load Texture2D of sprite sheet
6. Check what number key is pressed and change int accordingly
7. old = new KeyboardState
8. Draw menu options with Rectangle [ ] for destination and source Rectangles
9. DrawString for menu option labels that indicate the number user can input
10. Draw chosen option (2x) using Rectangle [int] and source Rectangle [int]

# You’ve Been Targeted

1. Initialize old and new MouseState
2. Initialize Vector2 for center of tank (center of screen - Width/2, Height/2)
3. Initialize Rectangle for projectile
4. Initialize dx and dy
5. Load Texture2D for tank
6. Constantly Update projectile Rectangle.X and Rectangle.Y with dx and dy
7. If new MouseState click && old MouseState not click
   1. Find angle of mouse from tank (tangent, xDistance, yDistance) and set it as int
   2. Calculate dx and dy based on xDistance and yDistance
8. When projectile is outside screen, reset everything
9. old = new MouseState
10. Draw tank using Vector2 and Texture2D with rotation angle in radians
11. Draw projectile Rectangle

# Rock and a Hard Place

1. Initialize new KeyboardState
2. Initialize Rectangle for stationary hard place
3. Initialize Rectangle for moving rock
4. Initialize Color for rock
5. Load Texture2Ds for both objects
6. Check for arrow movements
   1. Up = Rectangle.Y -= 10
   2. Down = Rectangle.Y += 10
   3. Right = Rectangle.X += 10
   4. Left = Rectangle.X -= 10
7. Make new method isOverlapping(Rectangle 1, Rectangle 2)
   1. If 1.X + 1.Width > 2.X and If 1.X < 2.X + 2.Width
   2. If 1.Y + 1.Height > 2.Y and If 1.Y < 2.Y + 2.Width
8. Use if statement to set Color
9. Draw both objects with appropriate Colors

15 pts

# Huff-N-Puff

1. Initialize new KeyboardState
2. Initialize Rectangle for boy and feather
3. Initialize Rectangle [ ] for sources
4. Initialize int for current sprite
5. Initialize int for score
6. Initialize bool for game over
7. Load SpriteFont
8. Load sprite sheet Texture2D
9. Move boy with arrow controls
   1. Left - Rectangle.X -= 2
   2. Right - Rectangle.X += 2
   3. Change source using current
      1. If left pressed after right, set current = 0
      2. If right pressed after left, set current = 5
10. Update feather.Y--
11. If feather.Intersects(boy)
    1. If space is pressed
       1. score++
       2. feather.Y++
       3. feather.X += random(1,20)
12. If feather.Y + feather.Height == Height
    1. Stop game using bool
    2. Display “Press R to play again”
    3. Check for pressed R
       1. Initialize again
13. Draw boy and feather
14. DrawString of score using SpriteFont

30 pts

# Pong

1. Initialize new KeyboardState
2. Initialize Rectangles for both paddles
3. Initialize ints for games and scores on both sides
4. Initialize double random X and Y velocity of ball at the center of screen (Width/2, Height/2)
5. Initialize double for random spin and rotation
6. Initialize double for ball X and Y (intervening)
7. Load Texture2D of background
8. Load Texture2D of white squares
9. Load Texture2D of two paddles
10. Load SpriteFont for scores
11. Detect paddle movement (W-S or Up-Down)
    1. paddle Rectangles.Y-- or Y++
12. If ball Rectangle intersects left or right side
    1. Initialize again
    2. Add to score as needed
    3. If one side has more than 11 with minimum difference of 2, then that side wins a game, otherwise continue
13. When the ball hits top or bottom walls or paddles
    1. Change X and Y velocity as required by spin
    2. Reduce spin by fraction
14. Reverse xVelocity when it hits paddles
15. Regulate ball Rectangle.X and Rectangle.Y with intervening double variables
16. Draw white squares using loop
    1. new Rectangle(Width/2, i \* 50, 25, 25)
17. Draw both paddles and ball
    1. Ball should be rotating if there is spin
18. DrawString of scores and games on all 4 corners