As outlined in Sprite 3 the project required the following tasks:

* + - Task for creating a Mario class.
    - Task for creating a Mario state interface and an Animated Sprite interface.
    - Task for handling Mario's state transitions.
    - Task for implementing animation and drawing of each state that Mario can be in (use descriptive titles and remember to set the effort amount).
    - Task for creating enemy classes.
    - Task for creating item classes.
    - Task for creating a Block state interface and a Block State interface.
    - Tasks for animation and drawing of all types of blocks and or block states (use descriptive titles and remember to set the effort amount).
    - Task for implementing a Command interface and a Controller interface.
    - Task to implement a keyboard mapping to commands

In the Team Cinco DeMaya’s implementation of our Super Mario clone game we implemented 13 tasks as explained below:

1. Collision Detection & Response
2. A command structure for 11 command classes
   1. Down
   2. Fireball
   3. Idle
   4. Left
   5. Null
   6. Pause
   7. Quit
   8. Reset
   9. Right
   10. Run
   11. UP
3. Two Controller Classes with PauseMenue Classes for each
   1. Gamepad
   2. Keyboard
4. Content was added using the Factor Design Pattern with 5 classes
   1. Block
   2. Collectable
   3. Enemy
   4. Pipe
   5. Sprite
5. Game Objects were defined in 10 distenct classes
   1. Blocks
   2. Camera
   3. Enemy
   4. Fireball
   5. Level
   6. Mario
   7. Pipe
   8. Items
      1. Coins
      2. FireFlower
      3. OneUpMushroom
      4. Star
      5. Supper Mushroom
6. Game Object Sprite classes implemented the IAnaimated sprite interface and are grouped into five categories
   1. Background
      1. Overworld
      2. OverworldBGHills
      3. Underground
   2. Block
      1. Question Block
      2. Winged
   3. Enemies
      1. Dino
      2. Shell-Less Koopa
   4. Item
      1. Coin
      2. Fireball
      3. FireFlower
   5. Gate
   6. MarioMoving
   7. Null
   8. Static
7. Game Object States
   1. Block States
      1. Exclamation Block State
      2. Generic Block State
      3. Ground Block State
      4. Invisible Block State
      5. Pipe Block State
      6. Question Block State
      7. Stone Block State
      8. Wing Block State
   2. Enemy States
      1. Banzai Bill States
      2. Dino States
         1. Dead
         2. LeftSmashed
         3. LeftTall
         4. RightSmashed
         5. RightTall
      3. Koopa States
         1. DeadShellless
         2. LeftWalkingShellless
         3. RightWalkingShellless
      4. NullEnemy
   3. Mario States
      1. Big Mario
         1. LeftCrouchingBig
         2. LeftIdleBig
         3. LeftJumpingBig
         4. LeftMovingBig
         5. RightCrouchingBig
         6. RightIdleBig
         7. RightJumpingBig
         8. RightMovingBig
      2. Fire Mario
         1. LeftCrouchingFire
         2. LeftFireballFire
         3. LeftJumpingFire
         4. RightIdleFire
         5. RightJumpingFire
         6. RightMovingFire
      3. Small Mario
         1. Dead
         2. LeftCrouching
         3. LeftIdle
         4. LeftJumping
         5. LeftMoving
         6. RightCrouching
         7. RightIdle
         8. RightJumping
         9. RightMoving
   4. Pipe States
      * 1. DownPipe
        2. LeftPipe
        3. UpPipe
8. Game States
   1. Dead
   2. Game Over
   3. Lives Screne
   4. PauseGame
   5. PipeTransition
   6. PlayGame
   7. TransitionGame
   8. VictoryGame
9. HUD and Background
   1. BackgroundHolder
   2. HUD
10. Interfaces (which enforce methods as described)
    1. IAnamatedSprite
       1. Fields
          1. int UpdateSpeed {get;set;}
       2. Methods
          1. GetBoundingBox(Vector2 location)
          2. Update(GameTime gameTime)
          3. Draw (SpriteBatch spriteBatch, Vector2 location)
    2. IBlockState (which enforce methods as described)
       1. Methods
          1. GetBoundingBox(Vector2 location)
          2. Update(GameTime gameTime)
          3. Draw (SpriteBatch spriteBatch, Vector2 location)
    3. ICollectable
       1. Fields
          1. IAnamatedSprite sprite{get;set;}
          2. Vector2 position {get; set;}
          3. Vector2 velocity {get; set}
          4. ICollectablePhysicsState physicsState{get; set;}
          5. bool isSpawning {get; set;}
       2. Methods
          * 1. Update(GameTime gameTime)
            2. Draw (SpriteBatch spriteBatch, Vector2 location)
            3. GoLeft()
            4. Goright()
            5. Spawn()
    4. ICollectablePhysicsState
       1. Methods
          1. Update(Icollectable item, GameTime gameTime)
    5. ICommands
       1. Methods
          1. Execute()
    6. IController
       1. Methods
          1. Update()
    7. IEnemyPhysicsState
       1. Methods
          1. Update(Enemy enemy, GameTime gameTime)
    8. IEnemyState
       1. Fields
          1. Rectangle GetBounjdingBox(Vector2 location)
       2. Methods
          1. Update(Enemy enemy, GameTime gameTime)
          2. Draw (SpriteBatch spriteBatch, Vector2 location)
          3. TakeDamage(Enemy enemy)
          4. GoLeft(Enemy enemy)
          5. GoRight((Enemy enemy)
    9. IGameState
       1. Methods
          1. Update(Enemy enemy, GameTime gameTime)
          2. Draw (SpriteBatch spriteBatch, Vector2 location)
       2. IMarioPhysicsState Methods
          1. Update(Enemy enemy, GameTime gameTime)
          2. Run ()
    10. IMarioState
        1. Fields
           1. IAnamatedSprite Sprite{get; set}
        2. Methods
           1. GetBoundingBox(Vector2 location)
           2. Update(Enemy enemy, GameTime gameTime)
           3. Draw (SpriteBatch spriteBatch, Vector2 location)
           4. Up()
           5. Down()
           6. GoLeft()
           7. GoRight()
           8. Idle()
           9. Land()
           10. MakeBigMario()
           11. MakeSmallMario()
           12. MakeFireMario()
           13. MakeFireballMario()
           14. MakeDeadMario()
    11. IPipeState
        1. Methods
           1. GetBoundingBox(Vector2 location)
           2. Update(Enemy enemy, GameTime gameTime)
           3. Draw (SpriteBatch spriteBatch, Vector2 location)
           4. Eat(Mario mario, Pipe pipe)
           5. Puke(Mario mario, Pipe pipe)
           6. Chew( (Mario mario)
           7. Gag(Mario mario)
    12. ISpritFactory
        1. Methods
           1. IAnamatedSprite build(SpriteFactory.sprites sprite)
11. Physics States
    1. Enemy Physics
       1. EnemyFallingState
       2. EnemyGroundState
    2. Item Physics
       1. ItemFallingPhysics
       2. ItemGroundPhysics
    3. Mario Physics
       1. FallingState
       2. GroundState
       3. JumpingState
    4. Utility Classes
       1. ValueHolder
12. Other Standard Classes
    1. Game1
    2. LevelBuilder (builds the level from csv data file)
    3. Program (Main)
    4. SoundManger (provides sound effects: game.Content.Load<SoundEffect>(“Sound File”)
13. Content
    1. Sprite .png files
    2. Map .csv files
    3. Music and SFX .mp3 files
    4. ReadMe .txt file