|  |  |
| --- | --- |
| Class No. | I learnt |
| 01 | Object Oriented Programming – about classes and objects |
| 02 | Sprite objects – rectangular objects that have pre-defined properties & functions |
| 03 | Conditional Programming – IF Statement |
| 04 | Assign AI to the computer paddle – computerPaddle.y = ball.y |
| 05 | Conditional Programming – FOR loop |
| 06 | Game states – serve, play, end |
| 07 | Sounds & Animations- playSound(“name\_of\_the\_file”,true)  playerPaddle.setAnimation(name\_of\_the\_animation) |
| 08 | CAPSTONE CLASS – World’s Hardest Game |
|  |  |
| 09 | Shifted from Code.org to p5 web editor how to create an infinite scrolling bg by giving it velocity and resetting it animations- loadAnimation() & addAnimation()  Trex jump & simulate gravity trex.velocityY = trex.velocityY + 0.5; |
| 10 | Debugged 2 bugs – 1. Trex should listen to the space key only when on the ground  2. Trex should look like it is walking on the ground invisible ground- ground.visibile = false; |
| 11 | function spawnClouds() frameCount, random numbers, depth |
| 12 | function spawnObstacles() string concatenation, switch statement, framerate, memory leak - lifetime |
| 13 | Game states – play and end & groups |
| 14 | Trex was colliding at a distance – trex.debug = true; trex.setCollider(“rectangle”,0,0,100,100) changed the trex’s animation – changeAnimation  Disappearing objects – lifetime was made negative |
| 15 | Sounds & Game Adaptivity & assign AI to the trex |
| 16 | Scope of variables – Global & Local variables  Reset functions |
| 17 | Host the game online – Web Server 200 OK (local computer), Github |
| 18 | Convert website into an application using a web-wrapper app called ‘Thunkable’ |
| 19 | CAPSTONE CLASS – Ghost Runner |
|  |  |
| 20 | Algorithms – isTouching() and bounce() |
| 21 | Parameter driven functions & my own library |
| 22 | Physics Engine Introduction – bouncing ball |
| 23 | Classes & objects – Toppling boxes – push() and pop() |
| 24 | Created the different class of the Angry Birds Game & added images |
| 25 | Inheritance – Base/Parent class & Sub/Child classes – ‘extends’ & ‘super’ |
| 26 | Git |
| 27 | Matter.Constraint – constraint between the bird and a point |
| 28 | Slingshot class and its functions |
| 29 | Made the catapult and rubber band |
| 30 | Arrays – length, index position |
| 31 | Trajectory – smoke image |
| 32 | JSON data structure & API calls – different background according to hour of the day |
| 33 | Debugging techniques |
| 34 | CAPSTONE CLASS – Wrecking Ball |
|  |  |
| 35 | Introduction to database |
| 36 | Structure the code – form, player, game |
| 37 | Properties & functions of above classes – made a game in the database |
| 38 | Created car sprites |
| 39 | Added the images to the cars and the track |
| 40 | Mark the current car according to the player |
| 41 | Player Rank |