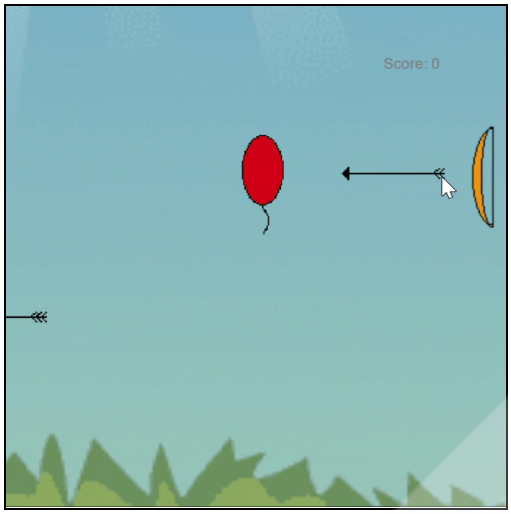
Topic	BUGS -The Curious Case of Disappearing Game Objects	
Class Description	**Students solve the problem of disappearing game objects inside the game. Students also set the collider radius so that the game reaches the END state when the Trex collides with the obstacle. Students add animation to display the game's end state and write the function to reset the game.**	
Class	**C15**	
Class time	**50 mins**	
Goal	• Set the collider radius so that the game ends when Trex touches the obstacle. • Diagnose and design a solution to the problem of disappearing obstacles and clouds. • Add animation and reset function when the game ends.	
Resources Required	• Teacher Resources: ○ VS Code Editor ○ Laptop with internet connectivity ○ Earphones with mic ○ Notebook and pen • Student Resources: ○ VS Code Editor ○ Laptop with internet connectivity ○ Earphones with mic ○ Notebook and pen	
Class structure	**Warm-Up Slides** **Teacher - led Activity 1** **Student - led Activity 1** **Teacher - led Activity 2** **Student - led Activity 2** **Wrap-Up Slides**	**10 mins** **10 mins** **5 mins** **10 mins** **10 mins** **5 mins**

WhiteHat Jr
{ Live Online Coding for Kids }

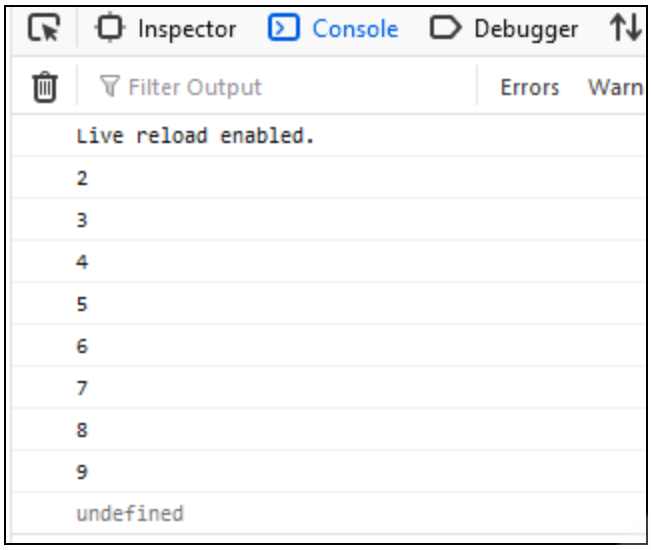
WARM UP SESSION - 10mins	
Teacher starts slideshow from slides 1 to 8 Refer to speaker notes and follow the instructions on each slide.	
Teacher Action	**Student Action**
Hi, so good to see you again! How have you been? So this is going to be the first class of the first module. Are you excited to learn something new? **Run the presentation from slide 1 to slide 3.** **Following are the warm up session deliverables:** ● Recall the progress in the game from the last class. ● Recall the bugs identified in the last class.	**ESR**: Thanks, Yes I am excited about it. Click on the slide show tab and present the slides Narrate the story by using hand gestures and voice modulation methods to bring in more interest in students.
QnA Session	
Question	**Answer**
Select the correct block of code to add the arrow object to arrowGroup. A. `arrowGroup.add(arrow);` B. `arrow.add(arrowGroup);` C. `arrowGroup.add();` D. `arrowGroup.addGroup(arrow);`	**A**

What will the following code block do? `redB.destroyEach();`  A. It will destroy only one balloon in the redB group. B. It will destroy only two balloons in the redB group. C. It will destroy all the balloons in the redB group. D. It will not destroy any balloon in the redB group.	**C**
Continue the warm-up session	
Activity details	**Solution/Guidelines**
Run the presentation from slide 4 to slide 8 to set the problem statement. ● Introduce concepts of Bugs, Errors & Debugging	Narrate the story by using hand gestures and voice modulation methods to bring in more interest in the student.
Teacher ends slideshow	

TEACHER-LED ACTIVITY 1 - 8mins	
Initiates Screen ShareTeacher	
Teacher Action	**Student Action**
Step-1 **Teacher-led Activity** Bugs, what is the first thing that comes to your mind when you hear this word?	**ESR** Student may reply with insects, crickets etc.
The Bugs we are talking about here are not the insects but the bugs in the computer program. What do you think, are bugs and errors different? The error means there is some mistake in the code due which the code is not running at all. It can be due to multiple reasons like you forgot to define a variable, you forgot a bracket, you misspelled a function name etc. There are countless reasons due to which there can be errors in the code. Now bugs are very interesting, if you have bugs in the code, your code may run, but it will have some different results than expected.	**ESR:** Varied. *Teacher opens the code for the Teacher Activity 1 in VS code Editor*

```
function setup() {
  createCanvas(400, 400);
  count();
}

function draw() {
  background(220);
}

function count() {
  let numbers = [1,2,3,4,5,6,7,8,9];
  let len = numbers.length;

  for (let i = 1; i <= len; i++) {
    console.log(numbers[i]);
  }
}
```

Can you guess the output of this code? *Run the code and show the output.* *Encourage the student to think and come up with the answer.*	*The student thinks about the answer.*
Output:	

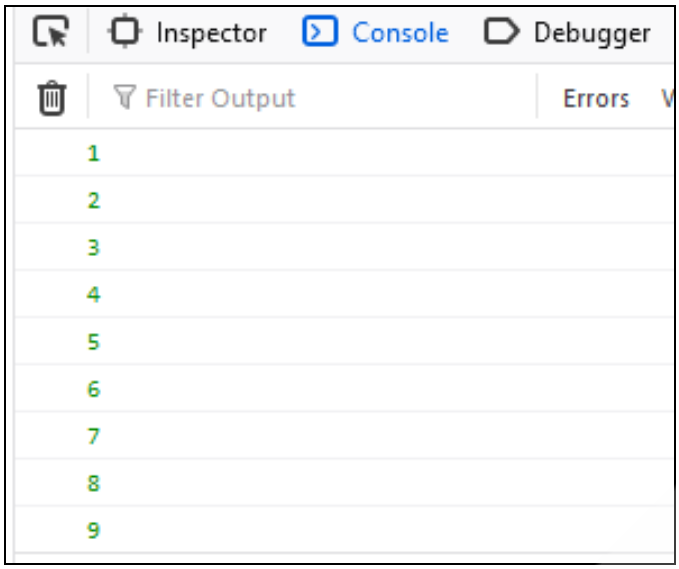


Here we can see the numbers are only getting displayed from 2 to 9 and at the end we are having some undefined value. Can you find out the reason for this bug?	*Encourage the student to find the solution.*
As we know, the index of the array starts from 0, here we want to display all the elements of the array, but if you notice in the for loop we are stating our variable from the 1, that is why it displays the values from the 2nd element because the index of the first element is 0. But what about the last undefined value? Why is that happening? If you notice the stopping condition of the for loop we say that stop when **i<= len**, which means stop the loop when i reaches the value which is equal or less than the length of the array. Length of the array means how many elements are there in the array. Here we have 9 elements in the array. But can you find the index of the last element?	**ESR:** Varied. **ESR**

Index of the first element is 0 so we move in the manner of 0,1,2, 3, 4,5,6,7,8 which means the index of the last element is 8. But in the loop we are going till the number 9 which we are getting from the length of the array, but there is no element at 9th index in the array, that is why we are getting a value as undefined. So how do we resolve these Bugs?	*The student calculates the index of the last element.* **ESR:** Change the starting condition in for loop to 0.
Correct! But that will only resolve the first bug which is to display the numbers from the first element, but what about the undefined variable? To solve this, we just need to change the stopping condition. Instead of saying i<=len, we will write i<len, then our loop will only run till the last element of the array.	**ESR:** Varied.

```
function setup() {
  createCanvas(400, 400);
  count();
}

function draw() {
  background(220);
}

function count() {
  let numbers = [1,2,3,4,5,6,7,8,9];
  let len = numbers.length;

  for (let i = 0; i < len; i++) {
    console.log(numbers[i]);
  }
}
```

Output:



Teacher starts slideshow from slides 9 to 10
Refer to speaker notes and follow the instructions on each slide.

As we can see, we have resolved these bugs from this code. We have one more bug challenge for you. Can you solve it? Let's try. I will guide you through it.	**ESR:** Yes!

Teacher ends slideshow

Teacher Stops Screen Share

Now it's your turn. Please share your screen with me.	

- **Ask Student to press ESC key to come back to panel**
- **Guide Student to start Screen Share**
- **Teacher gets into Fullscreen**

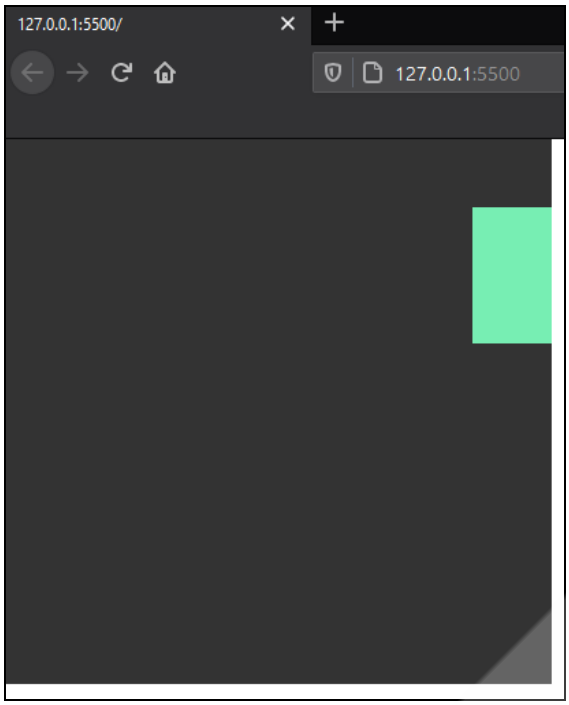
ACTIVITY

- **Solve the bug in the code.**

Teacher Action	Student Action
Step-2 **Student-led Activity 1** **10 mins** Here we have a bouncing cube on the screen, but the problem is when the cube is bouncing, it's going outside the wall and then coming back. Can you figure out what is the issue here and make it bounce on the walls? The cube should be bouncing on the walls which means the right side of the cube will touch the right wall and the cube will start moving in the opposite direction and the same will happen with the left side. *Guide the student to understand and resolve the bug.*	The student opens the Coding Activity link and runs it in the VS code editor. **ESR:** Varied.

Code with BUG:

```
var ball;
var vx = 2;  //velocity in  x direction

function setup() {
  createCanvas(400, 400);
  ball = createSprite(100,100, 100,100);

}
function draw() {
  background(51);

  if(ball.position.x<=0 || ball.position.x>=width)
  {
    //mave the velovoty negative to change the direction
    vx = -vx;
  }

  //set the x velocit to vx
  ball.velocity.x = vx;

  drawSprites();
}
```



Encourage the student to try different solutions.

First let's understand the leading cause behind this bug.

The x and y positions of the sprite are at the center, so when we wrote in our code that bounce the ball when x position is less than 0 or greater than the width of the screen, the computer is bouncing the cube from the center, because that is the center.

If we want to bounce the cube on the edges we need to modify the condition.

We need to take the distance from the center of the cube to the left and right sides.
Width of the cube is 100 which means from center it will be 50 both sides.

In the condition instead of saying **ball.position.x<=0**, we will add 50 to the 0 and subtract 50 from the width because we don't want the center to come at the 0 position and then go all the way to complete the width which is 400 in our case. Hence, our cube will be bouncing in the 50 to 350 range. Hence the bug is resolved.	

```
var ball;
var vx = 2;  //velocity in  x direction

function setup() {
  createCanvas(400, 400);
  ball = createSprite(100,100, 100,100);

}
function draw() {
  background(51);

  //updated condition
  if(ball.position.x<=50 || ball.position.x>=width-50)
  {
    //mave the velovoty negative to change the direction
    vx = -vx;
  }

  ball.velocity.x = vx;

  drawSprites();
}
```

Teacher Guides Student to Stop Screen Share

TEACHER-LED ACTIVITY 2- 10 mins

Initiates Screen ShareTeacher

CHALLENGE

- **Set the animation for the Trex and collider radius so that the Trex touches the obstacle to set the game state to END.**
- **Think about the problem and the solution behind disappearing game objects in the end state.**

Teacher starts slideshow from slides 11 to 28

Teacher Action	Student Action
Step 2: **Teacher-led Activity** **(10 mins)** *Teacher opens* ***Teacher Activity 2*** Let us review the code from the previous class before we start working on any new code. Can you help review the code?	*The student goes over each block of code and thinks out loud the blocks of code and what they do.*
Teacher runs the code. When we run the code, we see that the game's END state is reached even before the Trex touches the obstacle. This is because every sprite has a collision radius around it. It is like the skin of the sprite. It can experience touch. When two objects collide, their collision radius comes in contact. You can turn on the collision radius by using: **trex.debug = true;** We can make the collision radius small or large. A large collision radius will invoke the effect of collision even when there is no actual collision. The collision radius of our Trex is very large. We can set the collision radius of the Trex to what we desire. sprite.setCollider() function is used to set the collider shape and size. *The teacher adds the* ***trex.setCollider("circle",0,0,40); trex.debug = true;***	*The student observes and learns.*

This will turn a green circle around the sprite on representing the collider radius. ● *The teacher explains each argument in the code:* ● The first argument is to set the shape of the collider. ● The second argument is for x-offset. x-offset is how far we want the center of the collider shape on x-axis from the center of the Trex animation. ● The third argument is for y-offset. y-offset is how far we want the centre of the collider in y-axis from the center of the Trex animation. ● The fourth argument is to set the radius of the Trex. *The teacher should experiment and change these parameters in the game to show how it changes the game behavior.*	*The student experiments with the setCollider arguments with the teacher.*
	

```
console.log("Hello" + 5);

trex.setCollider("circle",0,0,40);
trex.debug = true
```

Let us run the code and see what happens. What do you see now?	**ESR:** The ground stops moving when the Trex touches the obstacles.
What does this mean? What is the game state when the collision happens?	**ESR:** The game state changes to END.
Good. We can also print the game state in the console and see it change when the collision happens. Let's do it. *The teacher writes the code to print the game state on the console.* *The teacher runs the code.*	*The student sees the output.* *Game state is 1(PLAY) when the game runs but as soon as the collision with the obstacle happens, it changes to 0(END).*

```
JS sketch.js ×
JS sketch.js > ...
}

function draw() {

  background(180);
  //displaying score
  text("Score: "+ score, 500,50);

  console.log("this is ",gameState)

```

Alright. We still have the curious case of disappearing obstacles and clouds. We will think about it in a minute. But we also have this trex in running animation even when it has collided with the obstacles.

In the actual game, the Trex animation after END state changes to a different image where its eyes pop out after the collision.

Can you do this? There is an image already uploaded for this.

The student opens up Student Activity 2 *to modify the code.*

Teacher Stops Screen Share	
Teacher starts slideshow :Slide 29 to 30	
Run the presentation for slides 29 to 30 to set the student activity context.	
Teacher ends slideshow	
Now it's your turn. Please share your screen with me.	
STUDENT-LED ACTIVITY 2 - 10 mins	

- **Ask Student to press ESC key to come back to panel**
- **Guide Student to start Screen Share**
- **Teacher gets into Fullscreen**

ACTIVITY

- **Diagnose and identify the solution for the case of disappearing game objects.**
- **Add animation to show END state and write the reset function to reset the game.**

Teacher Action	Student Action
Step 3: **Student-Led Activity** **(20 mins)** *Guide the student to change the trex animation when the gameState becomes END.* Do you remember how to change the animation of the sprite?	Student downloads the code from Student Activity 2 and opens in the VS code Editor. *The student writes code to change the trex animation in the end state.* **ESR:** Using sprite.setAnimation().

```
JS sketch.js ×
JS sketch.js > ...
119         restart.visible = true;
120
121         ground.velocityX = 0;
122         trex.velocityY = 0
123
124         //change the trex animation
125         trex.changeAnimation("collided", trex_collided);
126
127         //set lifetime of the game objects so that they are never destroyed
128       obstaclesGroup.setLifetimeEach(-1);
129       cloudsGroup.setLifetimeEach(-1);
130
131       obstaclesGroup.setVelocityXEach(0);
132       cloudsGroup.setVelocityXEach(0);
133     }
```

Okay, now let's think about the curious case of disappearing game objects. The clouds and the obstacles start disappearing after some time. Why do you think that's happening? Have we instructed the computer to remove the objects anywhere in our code?	**ESR:** We have to set Lifetime on these objects so they disappear after a few frames. ESR: **ESR:** Varied.
Awesome. Can you recollect how lifetime works?	**ESR:** We assign lifetime to any game object by assigning it a specific number. After each frame, the lifetime reduces by 1. When lifetime becomes 0, the game object/sprite disappears.
Amazing that you remember. Why were we instructing the computer to destroy/remove the sprite objects after sometime?	**ESR:** New objects - clouds and obstacles - are continuously getting spawned. They continue to occupy space in the memory even when they are outside the screen. This causes a 'memory leak', if we don't destroy the objects we don't need to store.

In the PLAY state, we gave each obstacle a lifetime. Why did we do that? What happened because we gave lifetime to each of the obstacles? But, in the END state, the obstacles don't move - do we want the obstacles to disappear after some time?	**ESR:** Because we did not want any memory leak. **ESR:** They disappeared after some time when they moved out of the screen. **ESR:** No!!
What number should we set as the lifetime of these game objects then, so that they never disappear? Remember how lifetime works. Every frame reduces the lifetime by 1 and the object disappears when lifetime is 0. What number for lifetime will never reach 0 even when the lifetime is reduced/subtracted by 1 in every frame?	*The student tries to make various guesses.* **ESR:** A negative number.
Exactly, if you set the lifetime of the game object to -1, with every frame, it will move away from 0 and never reach 0. We will need to assign the lifetime to a negative number to all the sprites in the ObstacleGroup and the CloudGroup when the gameState becomes END. We can set the lifetime of the sprites in the group using setLifetimeEach instruction at one go.	*The student observes and learns.*

Let us write the code for setting the lifetime of all the spawned objects in the groups to be -1 in the END condition of the game.	*The student writes code to set the lifetime of all the spawned objects in the group to -1.* *The student runs the code to check the output.*

```
JS sketch.js ×
JS sketch.js > ...
    trex.changeAnimation("collided", trex_collided);

    //set lifetime of the game objects so that they are never destroyed
    obstaclesGroup.setLifetimeEach(-1);
    cloudsGroup.setLifetimeEach(-1);

    obstaclesGroup.setVelocityXEach(0);
    cloudsGroup.setVelocityXEach(0);
  }
```

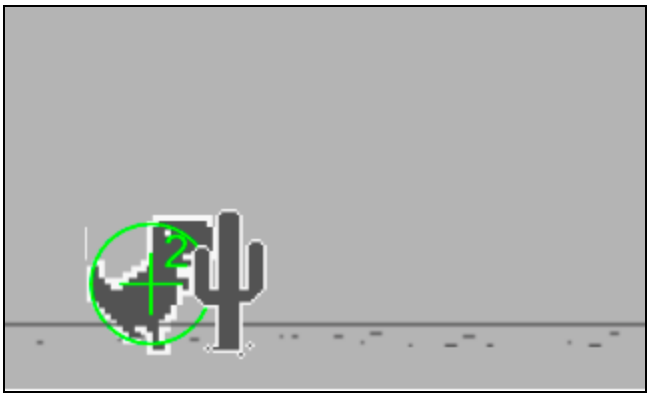
Output:

The objects do not disappear. There is a small bug when we press space just at the time of collision. Can you do that and see what happens? Can you make a guess why this happens? Excellent. Any idea how we can solve this bug?	*The student presses space just at the time of collision and sees the trex flying upwards without gravity.* **ESR:** There is no gravity in the END state and we have just instructed the Trex to jump. **ESR:** We can set the trex y velocity to 0.
Yes! Let's do that.	*Student writes code to set the trex.velocityY = 0 in the END state.* *The student runs the code to see if the bug is resolved.*

sketch.js

```
    }
  }
  else if (gameState === END) {
    console.log("hey")
     gameOver.visible = true;
     restart.visible = true;

     ground.velocityX = 0;
     trex.velocityY = 0

     //change the trex animation
     trex.changeAnimation("collided", trex_collided);

     //set lifetime of the game objects so that they are never destroyed
    obstaclesGroup.setLifetimeEach(-1);
    cloudsGroup.setLifetimeEach(-1);
```

output:



Amazing! Now one final thing. In the original game, we have the "Game Over" text and restart icon displayed on the screen when the game ends. Can you do that? *The teacher guides the student to place the "game over" icon and image on the screen.*	**ESR:** Varied. *The student writes the code and runs it to check the output.*

```
trex.scale = 0.5;

ground = createSprite(200,180,400,20);
ground.addImage("ground",groundImage);
ground.x = ground.width /2;

 gameOver = createSprite(300,100);
gameOver.addImage(gameOverImg);

restart = createSprite(300,140);
restart.addImage(restartImg);

gameOver.scale = 0.5;
restart.scale = 0.5;

invisibleGround = createSprite(200,190,400,10);
invisibleGround.visible = false;

//create Obstacle and Cloud Groups
obstaclesGroup = createGroup();
cloudsGroup = createGroup();
```

```
JS sketch.js ×
JS sketch.js > ...
117        console.log("hey")
118          gameOver.visible = true;
119          restart.visible = true;
120
121          ground.velocityX = 0;
122          trex.velocityY = 0
123
124          //change the trex animation
125          trex.changeAnimation("collided", trex_collided);
126
127          //set lifetime of the game objects so that they are never destroyed
128        obstaclesGroup.setLifetimeEach(-1);
129        cloudsGroup.setLifetimeEach(-1);
130
131         obstaclesGroup.setVelocityXEach(0);
132         cloudsGroup.setVelocityXEach(0);
133       }
134
135
136      //stop trex from falling down
137      trex.collide(invisibleGround);
138
```

Why don't we comment our code and indent it neatly so that it is readable for everyone? This is a very important practice for any programmer and takes regular practice and discipline to build. It should become a habit to comment on your code.	*The student spends time commenting on the code added.*
Optional So, now it's your friend's chance to look at the game and observe some hard-to-find bugs. Great job!	*The student and his/her friend look at the game and discuss some could-be bugs in the game.*

So now that you have seen how to fix bugs, it will be your responsibility to fix these bugs again and show it again to you. There still might be more bugs. The process of finding bugs and fixing them is an important part of the development of a game or an application.	
Teacher Guides Student to Stop Screen Share	
WRAP UP SESSION - 5 Mins	
Teacher starts slideshow Slide 31 to 41	
Teacher Action	**Student Action**
Run the presentation from slide 31 to slide 41 **Following are the warm up session deliverables:** • **Explain the facts and trivias** • **Next class challenge** • **Project for the day** • **Additional Activity**	Guide the student to develop the project and share with us
Quiz time - Click on in-class quiz	
Question	**Answer**
To reset the game which of these is correct? A. clouds group should be destroyed B. obstacle group should be destroyed C. score should be made 0 D. all of the above	**D**
What are the different parameters used in the setCollider() function? A. sprite.setCollider('collider shape',	**A**

xOffset,yOffset,width/radius, height,rotation) B. sprite.setCollider(xOffset,yOffset,width/radius, height,rotation); C. sprite.setCollider(xOffset,yOffset,'collider shape', width/radius, height,rotation) D. sprite.setCollider('collider shape', xOffset,yOffset,radius, height,rotation)	
What is the difference between an error and a bug? A. Error leads to unwanted behavior in the output whereas bugs lead to not displaying the output at all. B. Both bugs and errors are the same. C. Bugs lead to unwanted behavior in the output whereas errors lead to not displaying the output at all. D. None of the above.	**C**
End the quiz panel	
Teacher ends slideshow	
You get Hats Off for your excellent work! Next class we will add sounds to the game. We will also explore other ideas to make the game more fun and challenging. Again, eagerly looking forward to the next class.	*Make sure you have given at least 2 Hats Off during the class for:* Creatively Solved Activities +10 Great Question +10 Strong Concentration +10

*** This Project will take only 30 mins to complete. Motivate students to try and finish it immediately after the class.** **Project Overview** **COLLECTING TREASURE** **Goal of the Project:** In Class 15, you have learned how to set the collider radius for Trex so that the game ends when Trex touches the obstacles. You also learned how to diagnose and design a solution for the problem of disappearing objects. In this project, you will apply what you have learned in the class to create a treasure collecting game. **Story:** Sahil loves treasure hunting, and he wants to create a treasure collecting game. He has already created a complete design of the game. Now he wants to add a scoring system to his game. Can you help him in creating a scoring system? I am very excited to see your project solution and I know you will do really well. Bye Bye!	**Note: You can assign the project to the student in class itself by clicking on the Assign Project button which is available under the projects tab.** *Students engage with the teacher over the project.*
Teacher Clicks ✖ End Class	

Additional Activities
 	
Encourage the student to write reflection notes in their reflection journal using markdown.
 	
Use these as guiding questions:
● What happened today?
  ○ Describe what happened.
  ○ The code I wrote.
● How did I feel after the class?
● What have I learned about programming and developing games?
● What aspects of the class helped me? What did I find difficult?
	The student uses the markdown editor to write her/his reflections in the reflection journal.

Activity	Activity Name	Links
Teacher Activity 1	Reference Code	https://github.com/pro-whitehatjr/Pro-c15_TA1
Teacher Activity 2	Trex Stage 5	https://github.com/pro-whitehatjr/PRO-C14-stage5
Teacher Activity 3	Group Reference	https://molleindustria.github.io/p5.play/docs/classes/Group.html
Teacher Activity 4	Trex Stage 6 (Full Reference Code)	https://github.com/pro-whitehatjr/C-15_trex-6
Student Activity 1	Reference Code	https://github.com/pro-whitehatjr/Pro-c15-SA
Student Activity 2	Trex Stage 5.5	https://github.com/pro-whitehatjr/PRO-c15-trex5.5
Student Activity 3	Group Reference	https://molleindustria.github.io/p5.play/docs/classes/Group.html
Teacher Reference visual aid link	Visual aid link	https://s3-whjr-curriculum-uploads.whjr.online/d6d090e1-032a-4982-903a-eb8590dd520f.html
Teacher Reference	In-class quiz	https://s3-whjr-curriculum-uploads.whjr.online/a50ad5b2-5446-4134-8c4e-0d2180e0ae15.pdf
Project Solution	Collecting Treasure	https://github.com/pro-whitehatjr/Project_C15_Collecting_Treasure