CHECKPOINT REVISION

CHECK THE NUMBER OF PENDING PROJECTS & QUIZZES OF THE STUDENT IN THE PANEL.

IF THE STUDENT HAS MORE THAN 3 PENDING/EXPIRED PROJECTS OR QUIZZES, THEN GO AHEAD WITH THE REVISION CLASS.

IF THE STUDENT HAS COMPLETED ALL THE PROJECTS OR HAVING NO DOUBTS, THEN CONTINUE WITH THE BELOW LESSON PLAN.

.

Note to teachers [Only Applicable for C19]:

This is a CHECKPOINT REVISION CLASS meant for revising concepts learned so far. Teachers should guide students to complete pending/expired projects. If the student has pending projects less than 3, then you can continue to do the class activity.

IF STUDENT HAS > 3 PENDING PROJECT
SKIP CLASS ACTIVITY(INCLUDING VA & QUIZ) & HELP THEM COMPLETE PROJECT

ELSE
CONTINUE CLASS ACTIVITY

Next class will be C20

Topic	**CHECKPOINT REVISION CLASS: INFINITE RUNNER GAME**
Class Description	**Students upload the Ghost Runner game on GitHub and host it via GitHub server to share it online.**
Class	**C19**
Class time	**50 mins**
Goal	• Build the Ghost Runner Game. • Review the concepts covered in the past few classes.
Resources Required	• Teacher Resources: ○ Visual Studio Code ○ Laptop with internet connectivity ○ Earphones with mic ○ Notebook and pen • Student Resources: ○ Visual Studio Code ○ Laptop with internet connectivity ○ Earphones with mic ○ Notebook and pen

Class structure	Warm Up - Slide show option Teacher-Led Activity Student-Led Activity Wrap Up - Slide show option	10 Mins 5 Mins 30 Mins 5 Mins

WARM UP SESSION - 10 mins

Teacher starts slideshow  **from slides 1 to 17**
Refer to speaker notes and follow the instructions on each slide.

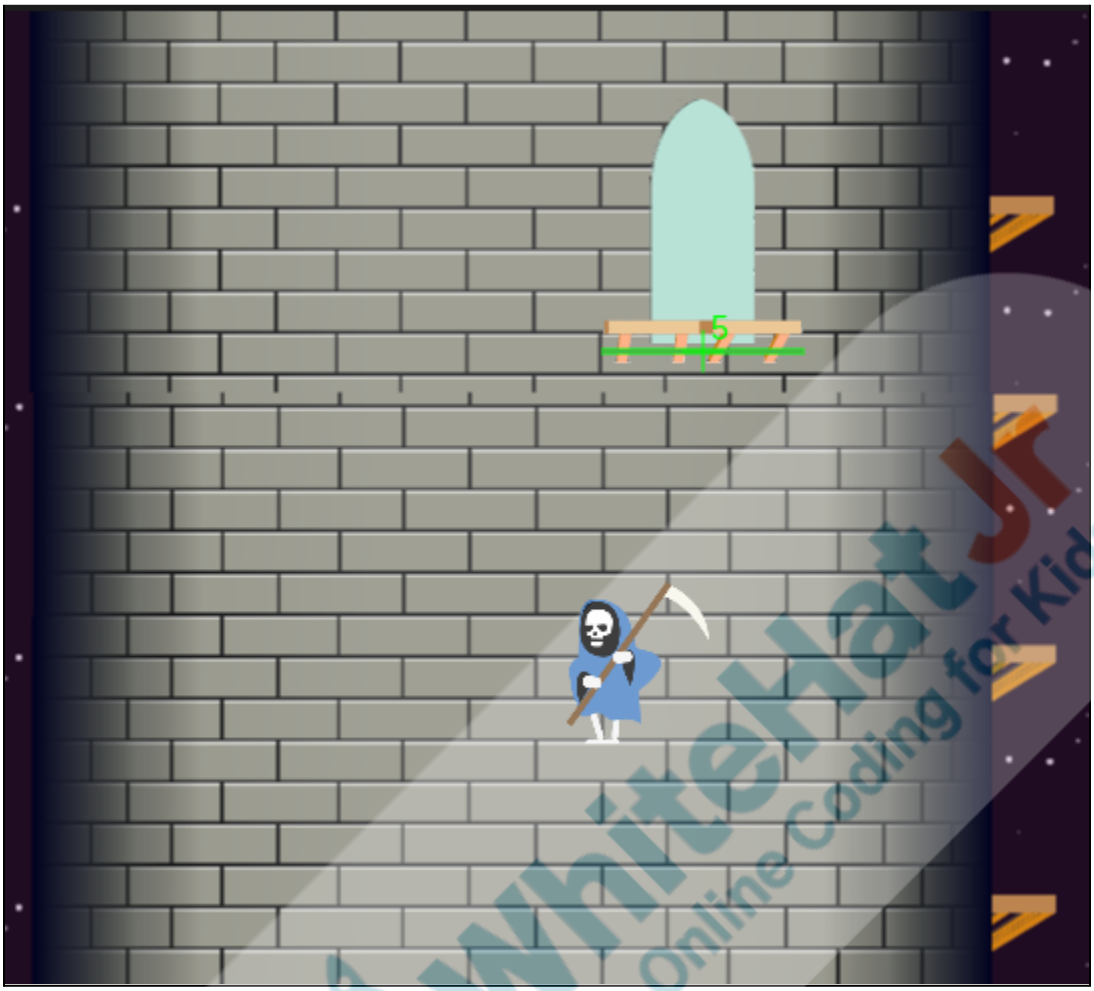
Activity details	Solution/Guidelines
Hey <student name>. How are you? It's great to see you! Are you excited to learn something new today? **Run the presentation from slide 1 to slide 5.** **Following are the warm up session deliverables:** ● Connecting students to the previous class. ● Recall critical points for capstone class.	**ESR**: Hi, thanks, yes I am excited about it! Click on the slide show tab and present the slides.

QnA Session

Question	Answer
Which of the following options can create a canvas that fits the size of a computer window? A. `createCanvas(window,window);` B. `createCanvas(windowWidth,windowHeight);` C. `createCanvas(width,height);` D. `createCanvas(200,200);`	**B**

Identify the correct option for making the path scroll infinitely on the Y axis as shown below.  A. ``` if(path.x > height){ path.x = height/2; } ``` B. ``` if(path.y > height){ path.x = height/2; } ``` C. ``` if(path.x > height){ path.y = height; } ``` D. ``` if(path.y > height){ path.y = height/2; } ```	**D**
Continue the warm up session	
Activity details	**Solution/Guidelines**
Run the presentation from slide 6 to slide 17 to set the problem statement. **Following are the warm up session deliverables:**	Narrate the story by using hand gestures and voice modulation methods to bring

• Introduce the Ghost Runner game as another infinite game designed in a finite space. • Compare similarities and differences to help the student brainstorm through the probable code. • Clearly discuss the game problems so that it is easy for the student to relate and develop the game independently.	in more interest in students.
Teacher ends slideshow 	
TEACHER-LED ACTIVITY - 8mins	
Teacher Initiates Screen Share	
CHALLENGE • **Decompose the game.** • **Ask the student to observe similarities and differences between this game and the Trex runner game.**	
Activity details	**Solution/Guidelines**
Step 2: Teacher-led Activity (5 min)	*Teacher opens the game to talk about it* ***Teacher Activity 1*** What did you observe in the game? What are the similarities between this game and the Trex Runner game? What are the differences?
	The student thinks about it **ESR:** In Trex, we had a horizontally scrolling ground. Here, we have a vertically scrolling tower.



What do you think would be the canvas size in the game?	**ESR:** Width: around 600 Height: around 600
How do you think are the doors and the railings appearing in the tower at different positions?	**ESR:** Doors are being spawned every few frames, similar to how clouds and obstacles were being spawned in the Trex game.
What are the game mechanics here?	**ESR:** • Player presses the “space” key to make the ghost jump up the tower.

	• The ghost can rest on the top of the railings. • If the ghost touches the bottom of the railing or if the ghost falls off the bottom, the game is over.
And of course there is the sound to make the game sound spooky. Do you think you can create the game on your own?	**ESR:** Yes! I can give it a try.
Awesome. I will help you through this. The images and other assets are already uploaded for you on the activity link.	
Teacher starts slideshow :Slide 18 & 19	
Run the presentation for slides 18 & 19 to set the student activity context.	
After you design the ghost runner game we're going to host your game online too, so let's quickly get started. Go ahead and share your screen with me. *Code to define variables,preload() function and tower sprite creation is already given as boilerplate to save time.*	*The student shares his/her screen, opens the Student Activity, and adds code to it.*
Teacher ends slideshow	
Teacher Stops Screen Share	
STUDENT-LED ACTIVITY - 30 mins	

• **Ask the student to press the ESC key to come back to the panel.** • **Guide the student to start Screen Share.** • **The teacher gets into Fullscreen.**	
ACTIVITY • **Create an infinite Scrolling tower.** • **Create the ghost in the tower game.** • **Add game mechanics and game states.** • **Detects collision between the ghost and doors.**	
Step 3: **Student-Led Activity** **(30 mins)** Let's first create an infinite scrolling tower. *Code to define variables,preload() function and tower sprite creation is already given as boilerplate.*	*Student opens **Student Activity 1.*** *The student writes to create an infinite vertically scrolling tower.*

```
var towerImg, tower;

function preload(){
  towerImg = loadImage("tower.png");

}

function setup(){
  createCanvas(600,600);
  tower = createSprite(300,300);
  tower.addImage("tower",towerImg);
  tower.velocityY = 1;
}


function draw(){
  background(0);

  if(tower.y > 400){
    tower.y = 300
  }
  drawSprites();
}
```

Let us spawn the doors in the tower.	*Student writes code to:* ● *load the door image* ● *create a door group* ● *write a function to spawn doors*

```
var towerImg, tower;
var doorImg, door, doorsGroup;

function preload(){
  towerImg = loadImage("tower.png");
  doorImg = loadImage("door.png");

  doorsGroup = new Group();
}

function setup(){
  createCanvas(600,600);
  tower = createSprite(300,300);
  tower.addImage("tower",towerImg);
  tower.velocityY = 1;
}
```

```
  }
  spawnDoors();
  drawSprites();
}

function spawnDoors() {
  //write code here to spawn the doors in the tower
  if (frameCount % 240 === 0) {
    var door = createSprite(200, -50);
    door.addImage(doorImg);

    door.x = Math.round(random(120,400));
    door.velocityY = 1;

    //assign lifetime to the variable
    door.lifetime = 800;

    //add each door to the group
    doorsGroup.add(door);
  }
}
```

	We can similarly spawn the railings.	*The student adds code to spawn the railings below the doors.*

```
var towerImg, tower;
var doorImg, door, doorsGroup;
var climberImg, climber, climbersGroup;

function preload(){
  towerImg = loadImage("tower.png");
  doorImg = loadImage("door.png");
  climberImg = loadImage("climber.png");

  doorsGroup = new Group();
  climbersGroup = new Group();
}


function setup(){
  createCanvas(600,600);
  tower = createSprite(300,300);
  tower.addImage("tower",towerImg);
  tower.velocityY = 1;
}
```

```
function spawnDoors() {
  //write code here to spawn the doors in the tower
  if (frameCount % 240 === 0) {
    var door = createSprite(200, -50);
    door.addImage(doorImg);

    var climber = createSprite(200,10);
    climber.addImage(climberImg);

    door.x = Math.round(random(120,400));
    door.velocityY = 1;

    climber.x = door.x;
    climber.velocityY = 1;

    //assign lifetime to the variable
    door.lifetime = 800;
    climber.lifetime = 800;

    //add each door to the group
    doorsGroup.add(door);
    climbersGroup.add(climber);
  }
}
```

Now, let us create the ghost character in our game. We want it to jump with the “space” key and we want gravity to act on the ghost.	*The student writes code to create the ghost character, make it jump with the “space” key and make it move “right” or “left” using arrow keys.*

We also want the ghost to move right or left with the arrow keys.	*Student runs the code and checks the output.*

```
var ghost, ghostImg;

function preload(){
  towerImg = loadImage("tower.png");
  doorImg = loadImage("door.png");
  climberImg = loadImage("climber.png");
  ghostImg = loadImage("ghost-standing.png");
}


function setup(){
  createCanvas(600,600);
  tower = createSprite(300,300);
  tower.addImage("tower",towerImg);
  tower.velocityY = 1;

  doorsGroup = new Group();
  climbersGroup = new Group();

  ghost = createSprite(200,200,50,50);
  ghost.scale = 0.3;
  ghost.addImage("ghost", ghostImg);
}
```

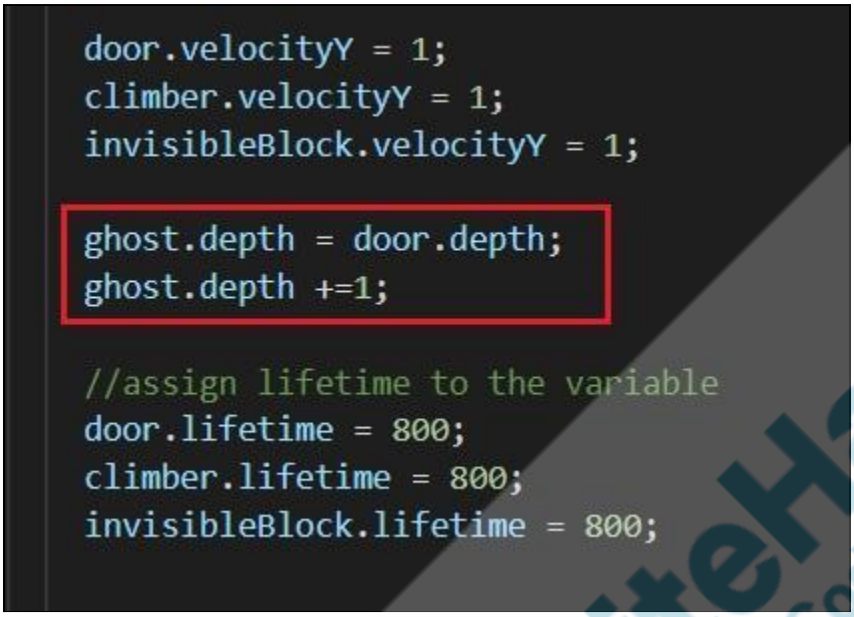
```
function draw(){
  background(0);
  if(tower.y > 400){
    tower.y = 300
  }

  if(keyDown("left_arrow")){
      ghost.x = ghost.x - 3;
  }

  if(keyDown("right_arrow")){
      ghost.x = ghost.x + 3;
  }

  if(keyDown("space")){
      ghost.velocityY = -5;
  }

  ghost.velocityY = ghost.velocityY + 0.8

  spawnDoors();
  drawSprites();
}
```

Let us add depth to our ghost character so that it appears on “top” of the door.

When we create sprites on the canvas they are shown in the order they were created. For example if we create a ghost sprite first and door sprite later, then the door sprite will be visible on top of the ghost.

But we want to display the ghost on top of the door. To overcome this issue we can set the depth of the sprite.

The student writes code to give more depth to the ghost than the door.

First we will set the depth of the ghost and door as equal to keep them at the same level, then we will add 1 to the depth of the ghost so that it appears over the door.		
		
	Time to add some game mechanics in our game. We want the ghost to be able to rest on the top of the railing. How should we do it? Perfect! Let's write code for that.	**ESR**: Make the ghost velocity 0 when the ghost touches the railing. *The student writes to get the ghost to rest on the railing.* *Student runs the code and checks the output.*

```
  ghost.x = ghost.x - 3;
  }

  if(keyDown("right_arrow")){
    ghost.x = ghost.x + 3;
  }

  if(keyDown("space")){
    ghost.velocityY = -5;
  }

  ghost.velocityY = ghost.velocityY + 0.8

  if(climbersGroup.isTouching(ghost)){
    ghost.velocityY = 0;
  }

  spawnDoors();
  drawSprites();

}

function spawnDoors() {
  //write code here to spawn the doors in the tower
  if (frameCount % 240 === 0) {
    var door = createSprite(200, -50);
```

Notice an issue? Yes. The velocity of ghost becomes 0 and the railings continue to move down with a velocity of 1.	**ESR:** The ghost behaves weirdly when it touches the railing from the bottom.

What should we do when the ghost touches the railing from the bottom? We should end the game if the ghost touches the railing from the bottom or falls down beyond the canvas. What can we do to detect the ghost hitting the bottom of the railing? ***NOTE:-*** ***Teacher can provide the student with the block of code and then explain what that block would do.*** We need to create invisible sprites which will act as the obstacles. We will first create the invisible sprite and then keep all the sprites in one group. We can then detect the collision of the ghost and the invisible sprite group using the collide function. If the ghost is touching any sprite from the invisible group then the ghost will be destroyed.	**ESR:** We should end the game. **ESR:** We can create invisible blocks which are spawned just below the railing. If the ghost touches these blocks, we can end the game. *Student writes the code and tests it.*

```
var towerImg, tower;
var doorImg, door, doorsGroup;
var climberImg, climber, climbersGroup;
var ghost, ghostImg;
var invisibleBlockGroup, invisibleBlock;


function preload(){
  towerImg = loadImage("tower.png");
  doorImg = loadImage("door.png");
  climberImg = loadImage("climber.png");
  ghostImg = loadImage("ghost-standing.png");
}

function setup(){
  createCanvas(600,600);
  tower = createSprite(300,300);
  tower.addImage("tower",towerImg);
  tower.velocityY = 1;


  doorsGroup = new Group();
```

```
climbersGroup = new Group();
invisibleBlockGroup = new Group();


ghost = createSprite(200,200,50,50);
ghost.scale = 0.3;
ghost.addImage("ghost", ghostImg);

}

function draw(){
background(0);

  if(tower.y > 400){
    tower.y = 300
  }

  if(keyDown("left_arrow")){
    ghost.x = ghost.x - 3;
  }

  if(keyDown("right_arrow")){
    ghost.x = ghost.x + 3;
  }
```

```
  if(keyDown("space")){
    ghost.velocityY = -5;
  }

  ghost.velocityY = ghost.velocityY + 0.8

  if(climbersGroup.isTouching(ghost)){
    ghost.velocityY = 0;
  }

  if(invisibleBlockGroup.isTouching(ghost) || ghost.y > 600){
    ghost.destroy();
  }


  spawnDoors();
  drawSprites();

}
```

```
function spawnDoors() {
  //write code here to spawn the doors in the tower
  if (frameCount % 240 === 0) {
    var door = createSprite(200, -50);
    door.addImage(doorImg);

    var climber = createSprite(200,10);
    climber.addImage(climberImg);

    var invisibleBlock = createSprite(200,15);
    invisibleBlock.width = climber.width;
    invisibleBlock.height = 2;



    door.x = Math.round(random(120,400));
    door.velocityY = 1;

    climber.x = door.x;
    climber.velocityY = 1;

    invisibleBlock.x = door.x;
    invisibleBlock.velocityY = 1;


    //assign lifetime to the variable
    door.lifetime = 800;
    climber.lifetime = 800;
    invisibleBlock.lifetime = 800;


    //add each door to the group
    doorsGroup.add(door);
    climbersGroup.add(climber);

    invisibleBlock.debug = true;
    invisibleBlockGroup.add(invisibleBlock);


    ghost.depth = door.depth;
    ghost.depth +=1;

  }
}
```

Let's introduce game states into our code. We would like the game to be over when the ghost touches the invisibleBlockGroup. We would also like a different screen to be displayed when the game is over. First we will set the **gameState** variable as "play". In the **draw()** function we will check this game state; if **gameState == "play"**, then only the user can move the ghost. If the gameState =="end" then we will display the game over text on the canvas.	The *student writes code to introduce game states and create a game over screen.*

```
var towerImg, tower;
var doorImg, door, doorsGroup;
var climberImg, climber, climbersGroup;
var ghost, ghostImg;
var invisibleBlockGroup, invisibleBlock;
var gameState = "play"

function preload(){
  towerImg = loadImage("tower.png");
  doorImg = loadImage("door.png");
  climberImg = loadImage("climber.png");
  ghostImg = loadImage("ghost-standing.png");
  spookySound = loadSound("spooky.wav");
}

function setup(){
  createCanvas(600,600);
  spookySound.loop();
  tower = createSprite(300,300);
  tower.addImage("tower",towerImg);
  tower.velocityY = 1;

  doorsGroup = new Group();
  climbersGroup = new Group();
  invisibleBlockGroup = new Group();

  ghost = createSprite(200,200,50,50);
  ghost.scale = 0.3;
  ghost.addImage("ghost", ghostImg);
}
```

```
function draw(){
  background(0);
  if (gameState === "play") {
    if(keyDown("left_arrow")){
      ghost.x = ghost.x - 3;
    }

    if(keyDown("right_arrow")){
      ghost.x = ghost.x + 3;
    }

    if(keyDown("space")){
      ghost.velocityY = -10;
    }

    ghost.velocityY = ghost.velocityY + 0.8

    if(tower.y > 400){
      tower.y = 300
    }
    spawnDoors();


    //climbersGroup.collide(ghost);
    if(climbersGroup.isTouching(ghost)){
      ghost.velocityY = 0;
    }
    if(invisibleBlockGroup.isTouching(ghost) || ghost.y > 600){
      ghost.destroy();
      gameState = "end"
    }

    drawSprites();
  }

  if (gameState === "end"){
    stroke("yellow");
    fill("yellow");
    textSize(30);
    text("Game Over", 230,250)
  }

}
```

```
function spawnDoors() {
  //write code here to spawn the doors in the tower
  if (frameCount % 240 === 0) {
   var door = createSprite(200, -50);
   var climber = createSprite(200,10);
   var invisibleBlock = createSprite(200,15);
   invisibleBlock.width = climber.width;
   invisibleBlock.height = 2;

   door.x = Math.round(random(120,400));
   climber.x = door.x;
   invisibleBlock.x = door.x;

   door.addImage(doorImg);
   climber.addImage(climberImg);

   door.velocityY = 1;
   climber.velocityY = 1;
   invisibleBlock.velocityY = 1;

   ghost.depth = door.depth;
   ghost.depth +=1;

   //assign lifetime to the variable
   door.lifetime = 800;
   climber.lifetime = 800;
   invisibleBlock.lifetime = 800;


   //add each door to the group
   doorsGroup.add(door);
   invisibleBlock.debug = true;
   climbersGroup.add(climber);
   invisibleBlockGroup.add(invisibleBlock);
  }
}
```

Amazing work! We now just need to load the spooky sound and play it in a loop while the game is on.	*The student writes code to load and play the sound!*

```
function preload(){
  towerImg = loadImage("tower.png");
  doorImg = loadImage("door.png");
  climberImg = loadImage("climber.png");
  ghostImg = loadImage("ghost-standing.png");
  spookySound = loadSound("spooky.wav");
}

function setup(){
  createCanvas(600,600);
  spookySound.loop();
  tower = createSprite(300,300);
  tower.addImage("tower",towerImg);
  tower.velocityY = 1;

  doorsGroup = new Group();
  climbersGroup = new Group();
  invisibleBlockGroup = new Group();
```

We've got the game ready! You can ask your friends to play the game and give you feedback on what can be better in the game!	
Awesome! In the next class we are going to start learning physics engine.This helps us in creating more interesting games, without writing too much code.	

Teacher Guides Student to Stop Screen Share	
WRAP UP SESSION - 5 Mins	
Step 4: **Wrap-Up** In today's class we reviewed what we had learned so far to create a game similar to Trex - Ghost on the Tower game! How are you feeling? What are the other kinds of infinite runner games do you think you can create now?	**ESR:** Varied. **ESR:** Varied.
We'll be starting with a new game in our next class.	
You get Hats Off for your excellent work! *Make sure you have given at least 2 Hats Off during the class for:* Next class, we will start a new exciting game, You will also be introduced to concepts of Physics Engine. Stay hooked into this till then.	

Teacher starts slideshow  from slide 20 to slide 29	
Activity details	**Solution/Guidelines**
Run the presentation from slide to slide **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 on Slide 24	
Question	**Answer**
To move the game character to the left, _____ is used. A. character_name.x=character_name.x+3 B. character_name.x=character_name.x-3 C. character_name.y=character_name.y+3 D. character_name.y=character_name.y-3	**B.**
The syntax for assigning the lifetime to the variable is _____. A. object.lifespan=value B. object.lifestart=value C. object.lifeend=value D. object.lifetime=value	**D.**
To add sound, which instruction is used? A. loadSound() B. loadMusic() C. load.Sound() D. load.Music()	**A.**

End the quiz panel	
FEEDBACK • **Encourage the student to make reflection notes in the markdown format.** • **Compliment the student for her/his effort in the class.** • **Review the content of the lesson.**	
Congratulations! Yet another milestone! Yet another achievement! Are you ready to face your fears? It's time to give your best while you apply the programming constructs acquired during the past few classes to create the Ghost Runner game.	**ESR:** Varied.
Project Overview **YOUR OWN INFINITE RUNNER GAME** **Goal of the Project:** In Class 19, you created a "Ghost on the Tower" infinite runner game and revised the concepts learned so far. In this project, you will apply what you have learned in the class to create your own infinite runner game. **Story:** You are a freelance coder who works on coding projects. You just won your first major project! A client wants you to build an infinite runner game. Now, there are so many infinite runner games already in the market. Your task is to create an infinite runner game that is distinct from others and is enjoyable to play. I am very excited to see your project solution and I know you will do really well.	**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.*

Bye Bye!	
Teacher ends slideshow 	
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
Student Activity 1	Ghost tower game	https://github.com/pro-whitehatjr/PRO_C19_LP_SA1
Teacher Activity 1	Final Reference code	https://github.com/pro-whitehatjr/PRO_C19_LP_TA1
Teacher Activity 2	Visual Aid	https://curriculum.whitehatjr.com/Visual+Project+Asset/PRO_VD/BJFC-PRO-V3-C19-withcues.html
Teacher Activity 3	In class quiz	https://s3-whjr-curriculum-uploads.whjr.online/9bed8fb0-5ade-4de5-99a0-d35ef970dba3.pdf
Teacher Reference	Project booster LP	https://s3-whjr-v2-prod-bucket.whjr.online/32dad903-b495-40ea-bfdd-a49ade21168d.pdf
Teacher Reference VA	Booster VA	https://curriculum.whitehatjr.com/Visual+Project+Asset/Boosterclass_withoutcues.html
Teacher Reference Video	Scaffolding Techniques	https://www.youtube.com/watch?v=0kB6pNmLoM8