Games are fun when we have music and sound effects in them.

First, we need to load all the sound files in the code. function we used to do that is the preload() function.

background music that will play continuously, then we need sound effects when the user:

● Cuts the rope.

● When the bunny eats the fruit or when the bunny could not eat the fruit.

● We will also add the sound of the air blowing when the user presses the blower button.

**Game to be challenging** for the user

we need to place the bunny a little away from the fruit so that if the user cuts the rope, the fruit will not fall directly on the bunny. So to push the fruit in the direction of the bunny we will create an air balloon button, when the user clicks on this button it will push the fruit towards the bunny.

For this task we need two things:

1. Function to push the fruit.

2. Button to execute the function

In the matter.js library, we have a function to apply force to a body. In our case, we will use this function to apply the force on the fruit body. But we need an event to apply force and that event is the button press. We will create a function named **airblow**() in the sketch.js file.

In this function, we will add code to apply force on the fruit body. To apply force we need the point of the body where we want to **apply the force** followed by the direction and amount of the force.

Points and direction to apply. **applyForce(fruit,{x:0,y:0},{x:0.01,y:0});** We will apply force on the center point of the fruit and the direction set as towards the right side. That is why the x value is +ve 0.01(this also sets the amount of force), to reverse the direction of force change the sign to negative, if you want to apply force in the vertical direction then you need to do the same with the y value. Along with this, we also want the air sound to be played when this function is called. To play this sound, we will simply call the play() function on that sound. Air is the sound variable, so we can write air.**play**();

We have our function, but this function won’t do anything until we call this function. To call the function we will attach this with a button and the button will be the image of a balloon. We will declare a variable for that, let’s call it var blower. In the setup() function, we will add code to create the image button, and we will also add the airblow() function with this button. To create the button, we need to specify the position and the size of the button

add this background music to our game. This step is very simple, we have loaded sounds in the preload() function. We will, now, start playing this background song in a setup() function, and this will play continuously. To do that we can say **bk\_song.play(),** we also want to set the volume of the song. For that, we will use the **setVolume**() function, in the brackets of the function you will pass the volume percentage, where 0.5 means 50 % and 1 means 100%. You can choose any value according to your needs. Here we will set it as 0.5

Here you can hear the background song playing. But we have no control over it. It is going to keep on playing.

You must have noticed in games we can turn the music on or off.

We will do a similar procedure as we did for the air blower sound. We will create a function to stop the music, then we will add that function with a button on the canvas.

Let’s define a function named **mute**().

To stop any sound, we can use the **sound.stop()** function, But this would completely stop the sound. We won't be able to play the sound until we restart the game. We need to make sure that we have the ability to stop and play sound with the press of a button.

To implement this, we need to test whether the sound is already playing or not.

If the sound is playing then we will stop the sound else we will play the sound. To check whether the sound is playing or not, we have an inbuilt function in p5js as **isPlaying**(). This will give the output as true if the sound is already playing, else it will give false.

Let's attach this with a button image, just like we did for the air blower. Declare a variable first as var **mute\_btn**; Then in the setup() function we create the image button and attach it with the mute() function.

Now we need to finally add three more sound effects:

1. When we cut the rope.

2. When the bunny eats fruit.

3. When the fruit drops on the ground and the bunny is sad.

First, we add the sound when the user cuts the rope. We already have the function **drop**() for it, we will just add sound.play() in this function, and it will be called when the user presses the cut button.

add the sound when the bunny eats the fruit. This sound needs to be added where we are detecting the collision and changing the animation. This is very easy in the draw() function to go to the collision condition and add the code eating\_sound.play().

add the sound of a sad bunny then you can play the game. We are already checking the collision of the bunny and ground. We will add the sad\_sound in this condition itself

The sad bunny sound is now playing continuously

When we are detecting the collision of fruit and ground we have one condition as **fruit!=null**, which is true, and the other condition of the fruit position being greater than 650 is also true, that is why the sound is playing endlessly.

to overcome this, We can make one condition false, then it will stop acting like this. We will set, fruit=null; due to this, the above condition

if(fruit!=null && fruit.position.y>=650) will be false. Since, we are using an AND operator, in this, both the if conditions should be true. If we set fruit=null, the above condition will be false and the following code will not execute, thus stopping the continuous music being played. So our sound will only play once which we desire.