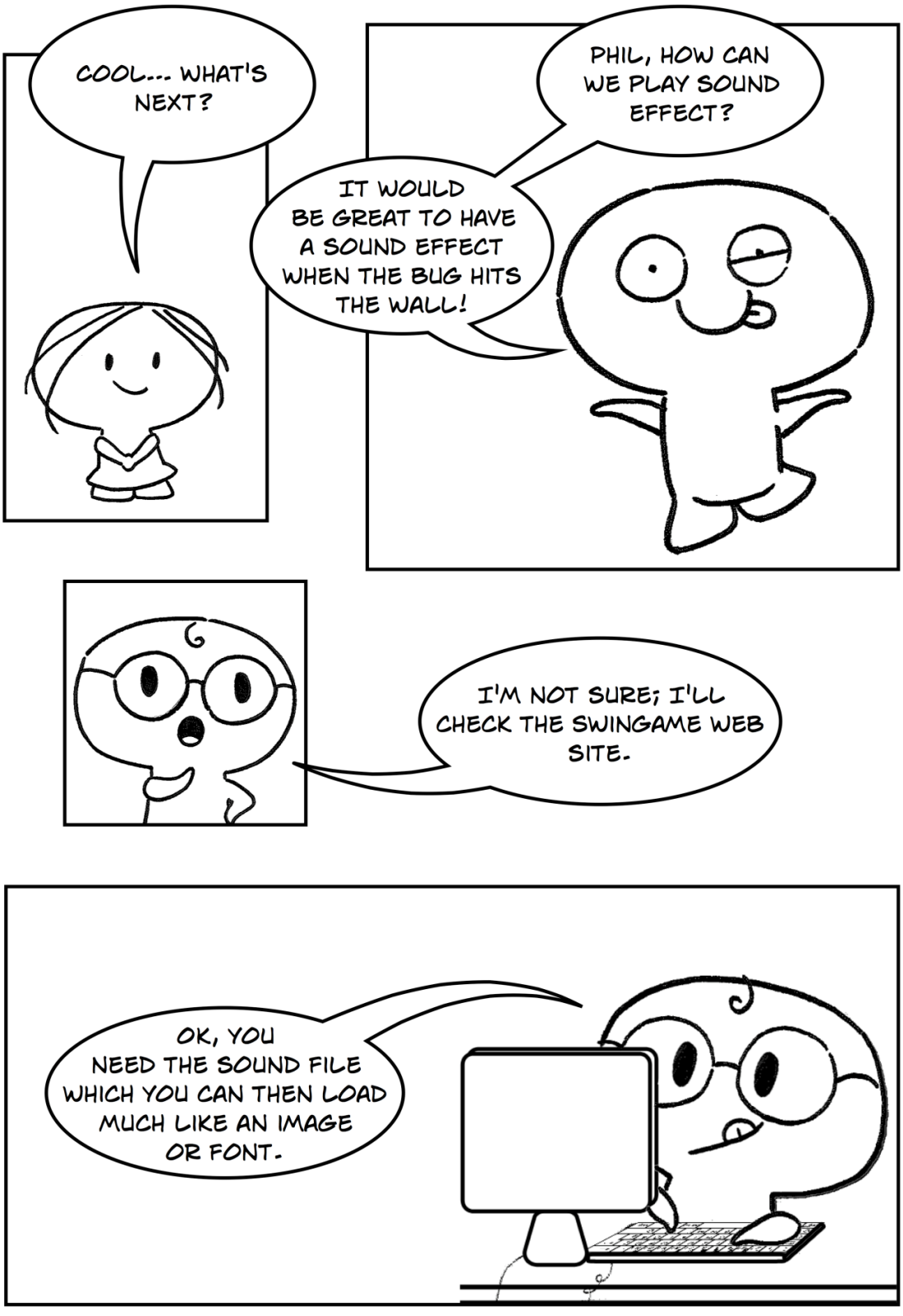
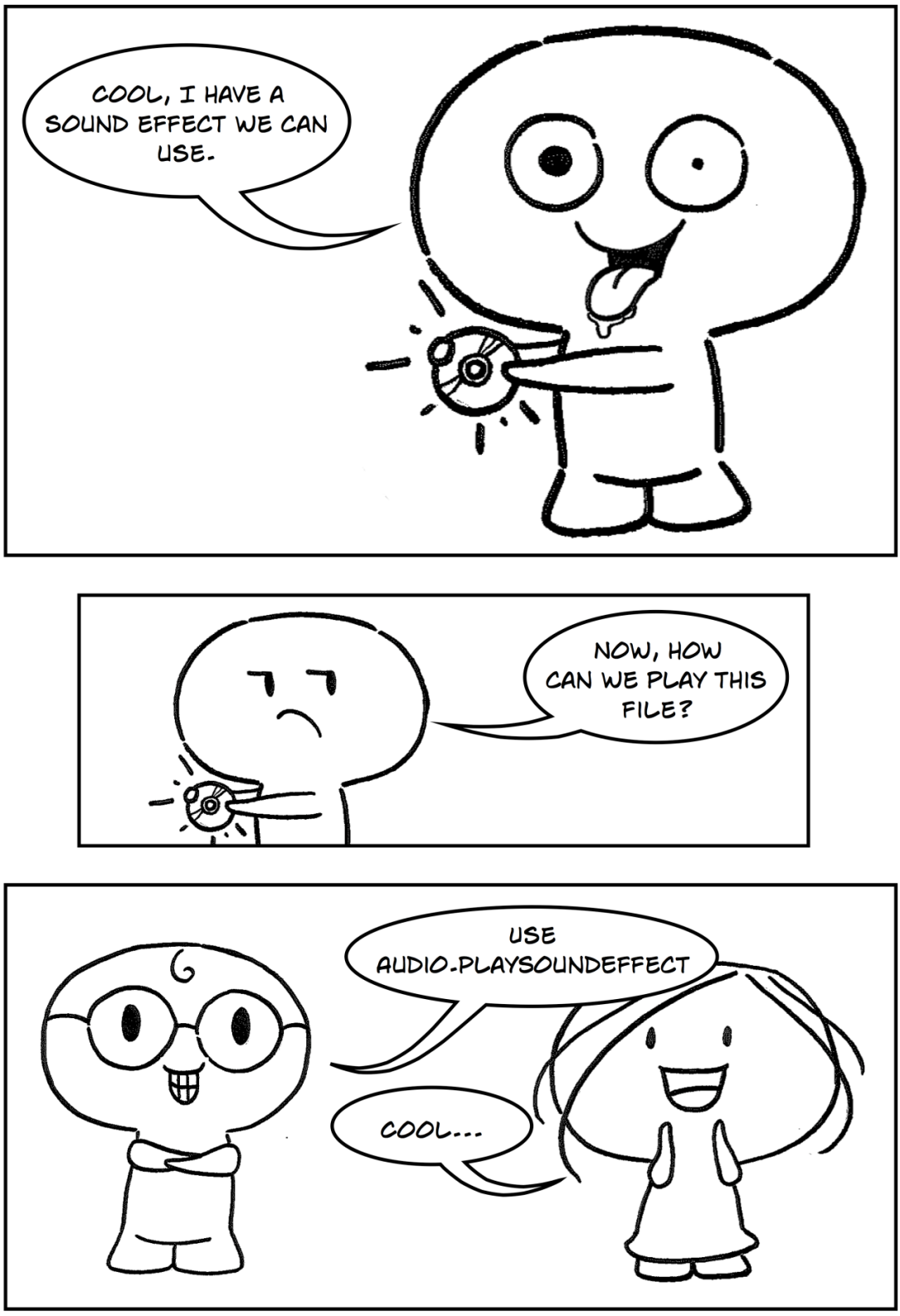
Chapter 4

"Sound and Keyboard"

Summary:

In this chapter you need to modify the solution from the previous chapter. You will add sound effects and music, and handle a keyboard input. All resources such as music or sound effects will be given to you.





Part 1

The first step for playing sound effect whenever bug hits the wall is to load sound effect to your program. In order to do so, follow the same steps as the previous tutorials for loading images and fonts. The audio file provided needs to be placed within the Resources > Sounds folder.

To play sound when the bug hits the wall you need to follow the logic below: (Figure 1)

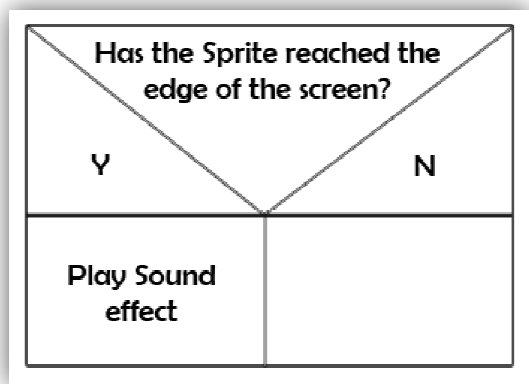


Figure 1

Open the project from the previous exercise (Bugs) and continue to work with this project.

Exercise 1: *Loading the sound effect*

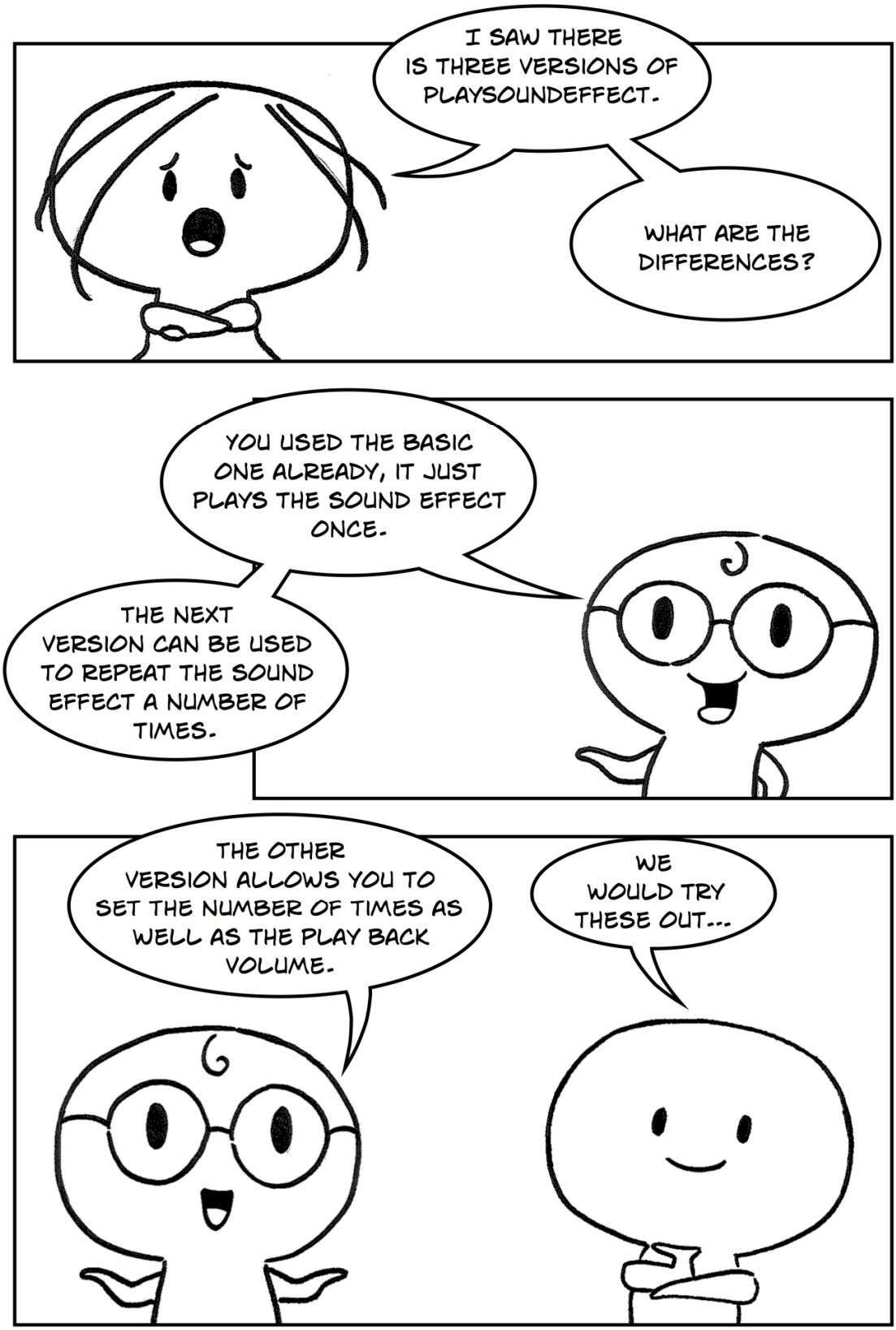
Make the following changes in your program and write your solutions to the worksheet:

1. Load "hit20.wav" file into your program. Use NewSound() function.

Exercise 2: *Playing sound effect*

Make the following changes in your program and write your solutions to the worksheet:

1. Play sound effect each time bug hits the wall. Use
   `Audio.PlaySoundEffect(GameSound("soundName"))`



Part 2

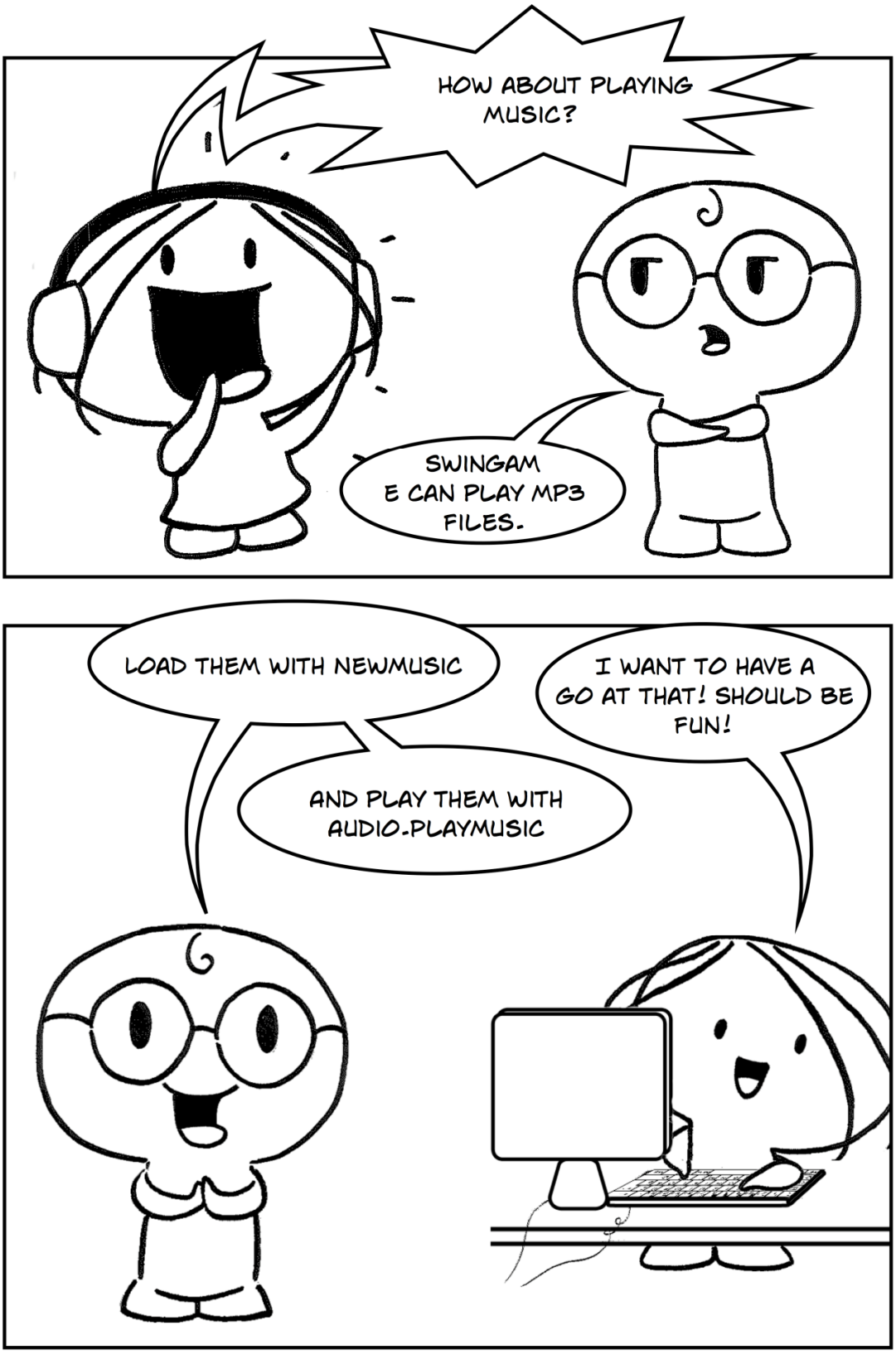
The sound effect can be played in three possible ways; so far we have used the most basic version which simply plays the sound effect once. The second version enables us to play the sound effect for a number of times and by using the third version we can tell the program to play the sound effect for a number of times and also we can set the play back volume.

Exercise 1: *Using different versions of PlaySoundEffect*


Make the following changes in your program and write your solutions to the worksheet:

1. Use Audio.PlaySoundEffect(GameSound("soundname"), NoOfLoops). Use this function with "hit20.wav" sound effect, which you previously loaded, before the start of the Game Loop.
2. Use Audio.PlaySoundEffect(GameSound("sooundname"), NoOfLoops, Volume). Use ths function with "hit20.wav" before the start of the Game Loop.

Note: NoOfLoops is number of times that the sound will be played. Volume is numeric representation of volume in your program (preferably use between 0 and 10).



Part 3

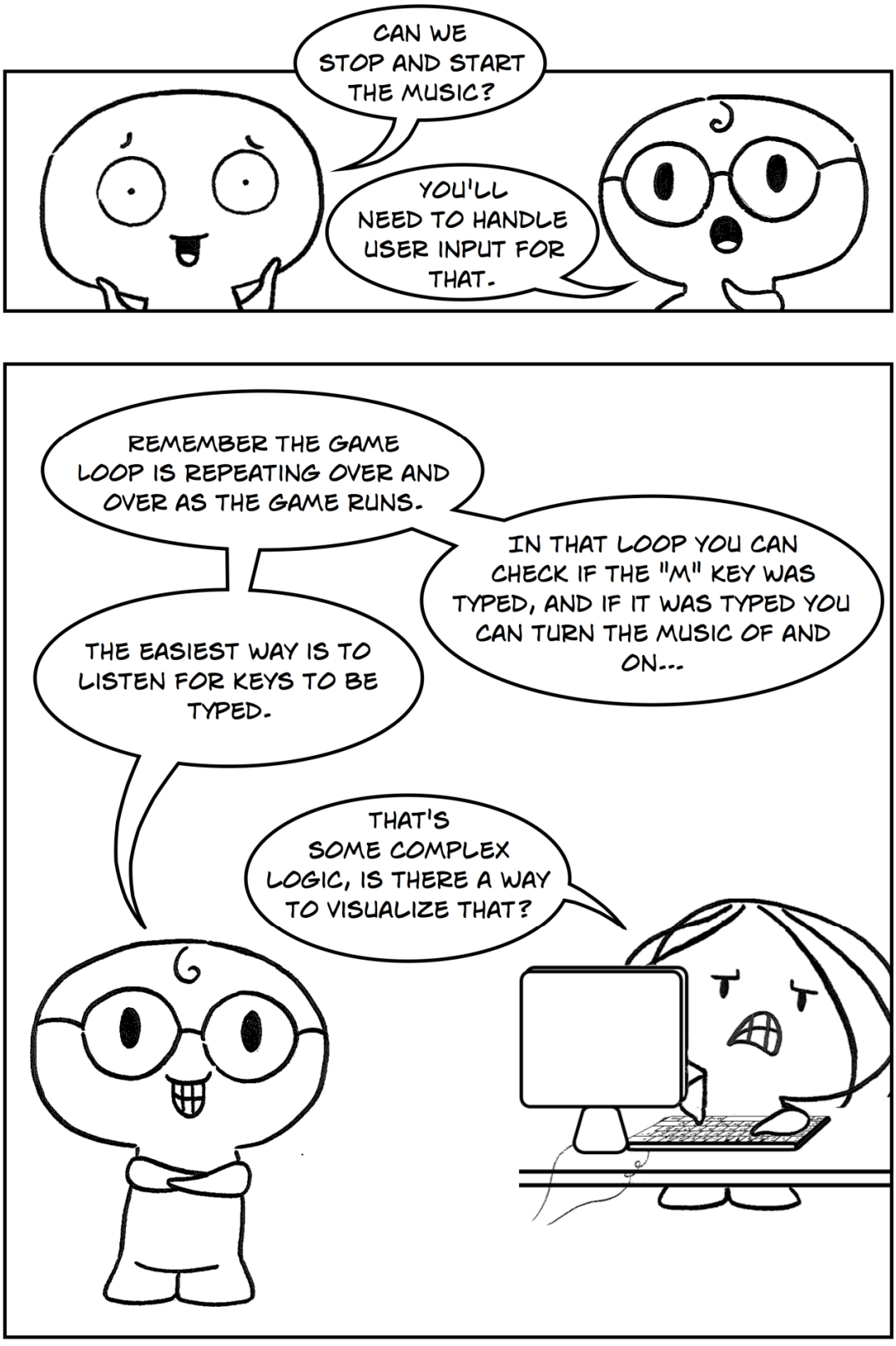
In addition to playing sound effects, we can also play music. We can load a music track into our program and play it once, or play in a certain amount of times, or loop it infinitely. All f these varieties are enclosed in one function – PlayMusic().

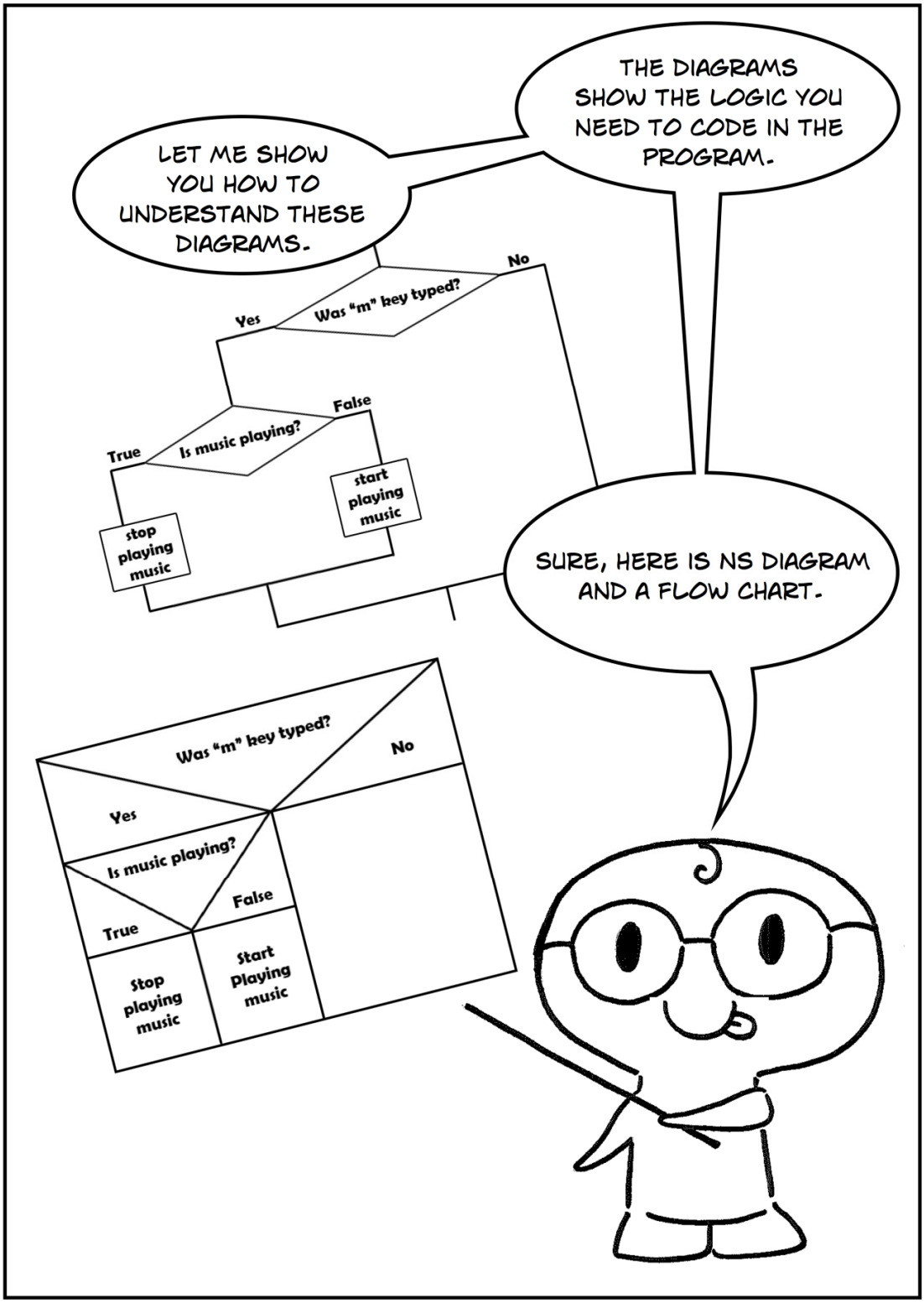
Exercise 1: *Playing music*


Make the following changes in your program and write your solutions to the worksheet:

1. Play a music track which is infinitely repeated. To do so, load lion.mp3 into your program, and use Audo.PlayMusic(GameMusic("trackname"), -1) where -1 is indicator of looping to infinity.

NOTE: To load music track use NewMusic("trackmane", "track") inside the LoadMusic Sub.





Part 4

In addition, you can stop music and start playing it any time you want. In order to do so you have to handle user input in your program. You can do this by using If..Else statement. The logic for this is shown in Figure 3 and 4.

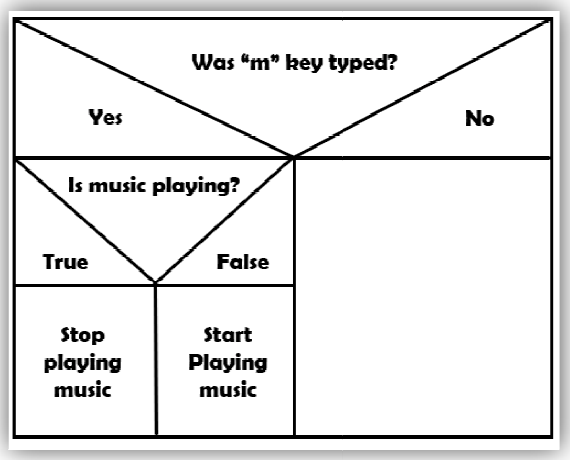


Figure 2

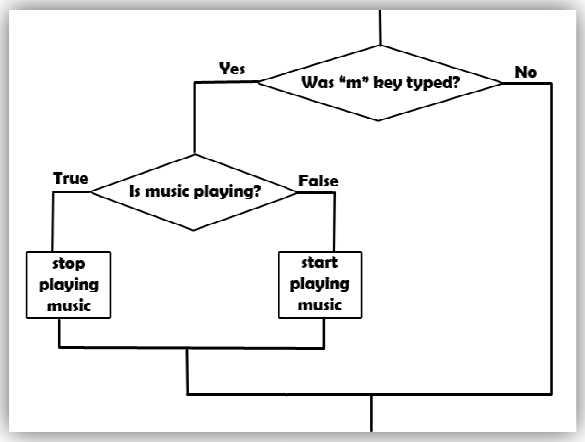


Figure 3

Did you know: Figure 3 is a flowchart. A flowchart is common type of chart, that represents an algorithm or process, showing the steps as boxes of various kinds, and their order by connecting these with arrows. Flowcharts are used in designing or documenting a process or program in various fields.

In order to check whether "m" key was typed you should use `Input.WasKeyTyped(SwinGame.Keys.VK_M)`. This is a built-in function in the SwinGame that checks whether the key was typed and returns true if it was and false if wasn't.

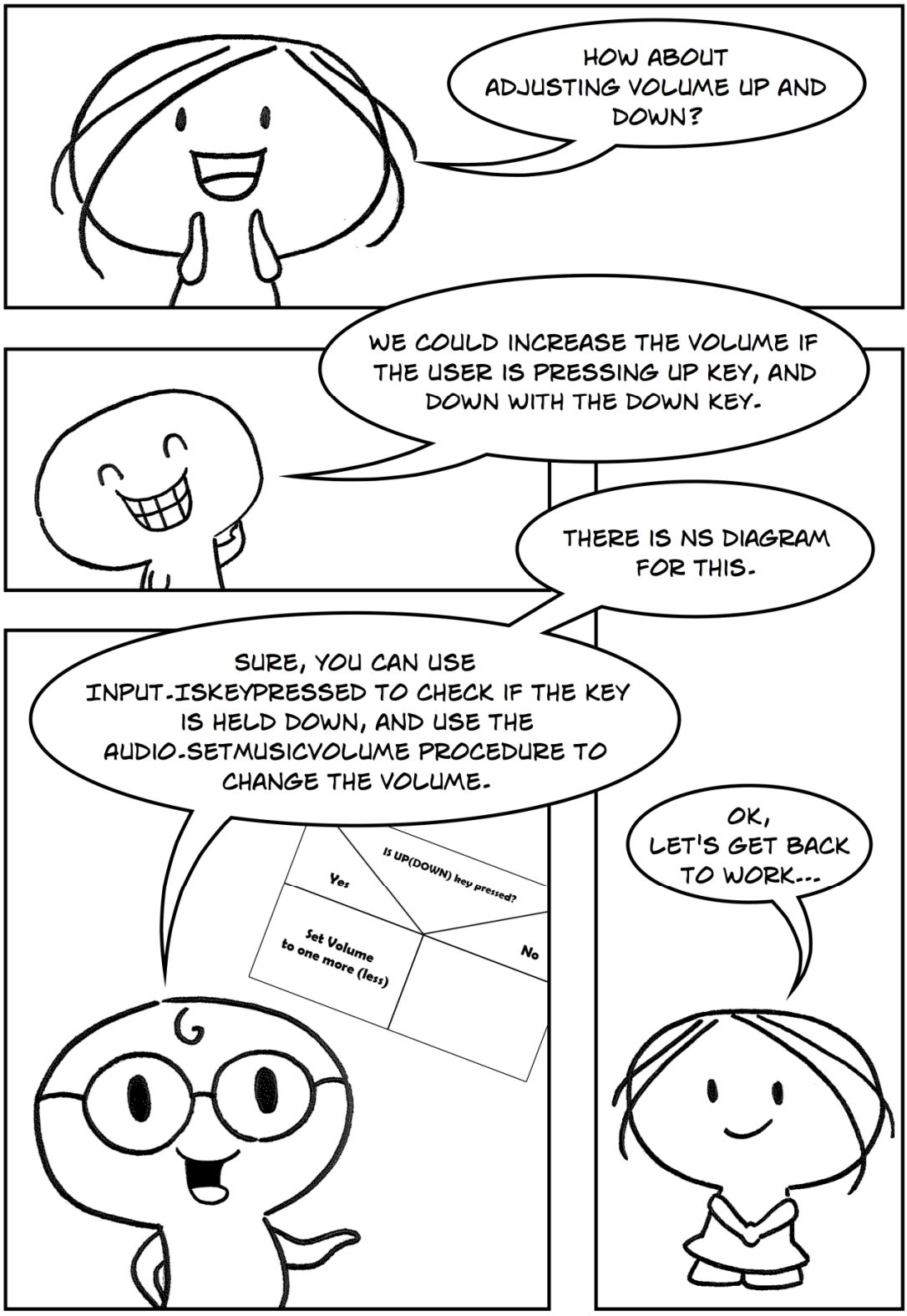
To check whether music is playing use `Audio.IsMusicPlaying`(). This function returns true if the music is playing and false if it is not. You can stop playing music by calling `Audio.StopMusic() function` and start to play it again by calling `Audio.PlayMusic(GameMusic("trackname"), -1)`.

Exercise 1: *Handling user input.*

Make the following changes in your program and write your solutions to the worksheet:

1. Allow to start and stop music in your program when typing the "m" key. Use the logic represented in Figure 1 and 2 and functions introduced above.

NOTE: The code for this exercise should be located inside the game loop.



Part 5

To change the volume you should follow the logic shown in Figure 4.

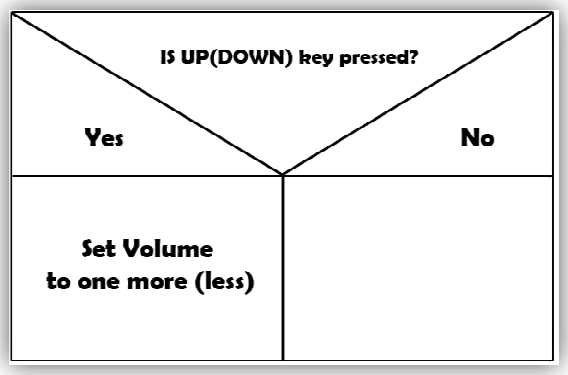


Figure 4

You can check whether the key was pressed by using `Input.IsKeyPressed(SwinGame.Keys.VK_Up/DOWN)`. This is a built-in function in SwinGame which returns True or False. To tell to your program to change the volume when arrows were pressed use `Audio.SetMusicVolume(Audio.MusicVolume +/- 0.01F)`.

Exercise 1: *Changing the volume*

Make the following changes in your program and write your solutions to the worksheet:

1. Write the code that allows you to change the volume up when UP key was pressed and down when DOWN key was pressed.