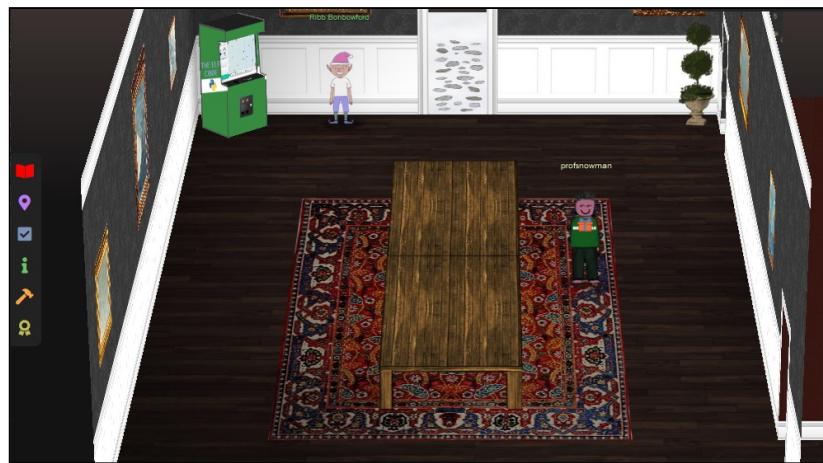


KringleCon 4: Calling Birds!

Dining Room



1. Click to talk to Ribb Bonbowford

Hello, I'm Ribb Bonbowford. Nice to meet you!

Are you new to programming? It's a handy skill for anyone in cyber security.

This here machine lets you control an Elf using Python 3. It's pretty fun, but I'm having trouble getting beyond Level 8.

Tell you what... if you help me get past Level 8, I'll share some of my SQLi tips with you. You may find them handy sometime around the North Pole this season.

Most of the information you'll need is provided during the game, but I'll give you a few more pointers, if you want them.

Not sure what a lever requires? Click it in the Current Level Objectives panel.



New [Hint] Unlocked: Lever Requirements!
[Click here to see this item in your badge.](#)

2. Click to talk to Ribb Bonbowford

You can move the elf with commands like `elf.moveLeft(5)`, `elf.moveTo({"x":2,"y":2})`, or `elf.moveTo(lever0.position)`.



New [Hint] Unlocked: Moving the Elf
[Click here to see this item in your badge.](#)

3. Click to talk to Ribb Bonbowford

Looping through long movements? Don't be afraid to moveUp(99) or whatever. You elf will stop at any obstacle.



New [Hint] Unlocked: Bumping into Walls!
[Click here to see this item in your badge.](#)

4. Click to talk to Ribb Bonbowford

You can call functions like myFunction(). If you ever need to pass a function to a munchkin, you can use myFunction without the () .



New [Hint] Unlocked: Function Calls!
[Click here to see this item in your badge.](#)

5. Click the **i** (Hints) icon

6. Click **Function Calls**

Function Calls

From: Ribb Bonbowford
Terminal: Elf Code Python

You can call functions like `myFunction()`. If you ever need to pass a function to a munchkin, you can use `myFunction` without the `()`.

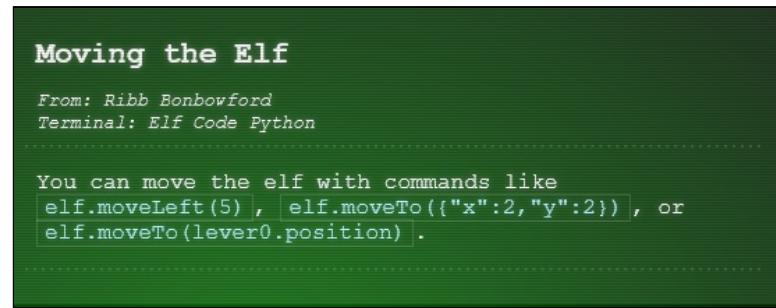
7. Click **Bumping into Walls**

Bumping into Walls

From: Ribb Bonbowford
Terminal: Elf Code Python

Looping through long movements? Don't be afraid to `moveUp(99)` or whatever. You elf will stop at any obstacle.

8. Click Moving the Elf

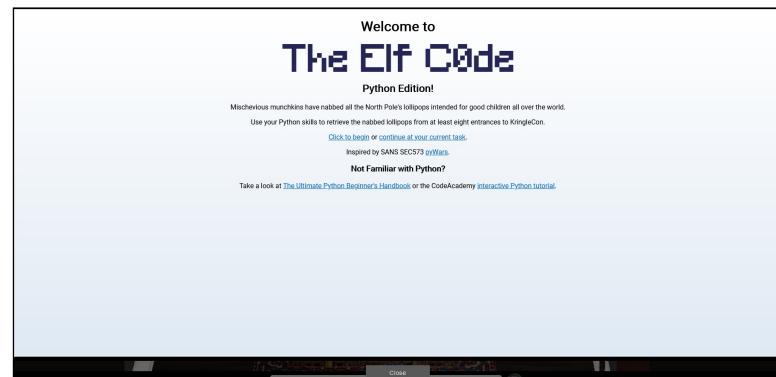


9. Click Lever Requirements



11. Click [Exit]

12. Click The Elf Code machine



13. Click Click to begin



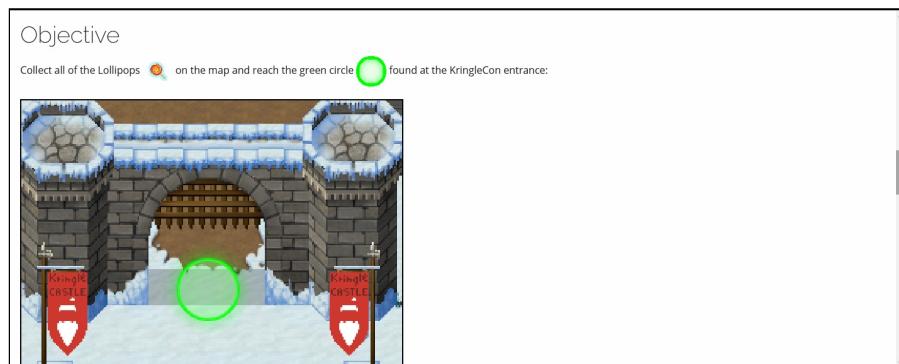
14. Close the Level 0 - Elf Code Demo window



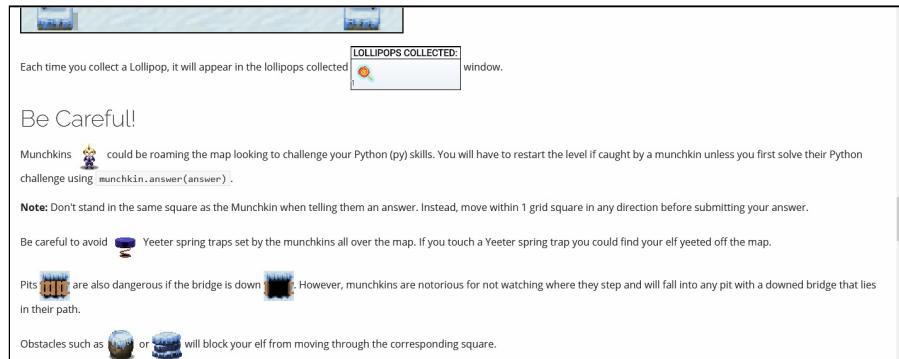
15. Scroll Down the window



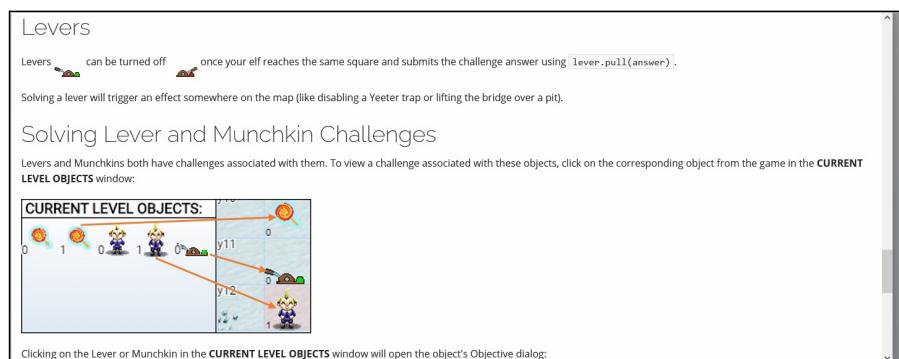
16. Scroll Down the window



17. Scroll Down the window



18. Scroll Down the window



19. Scroll Down the window

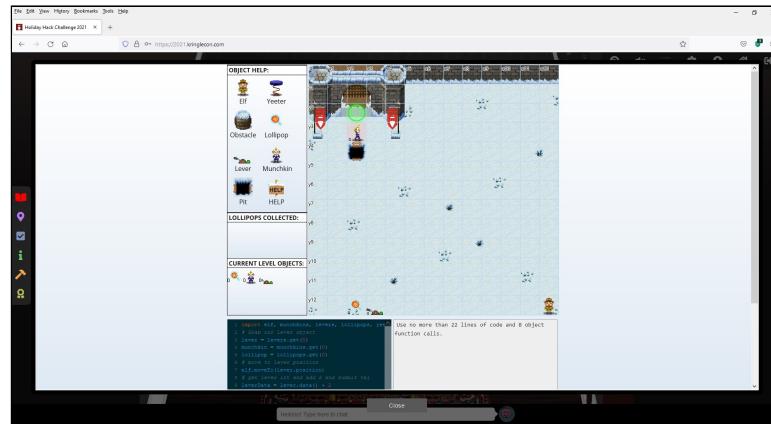


20. Scroll Down the window

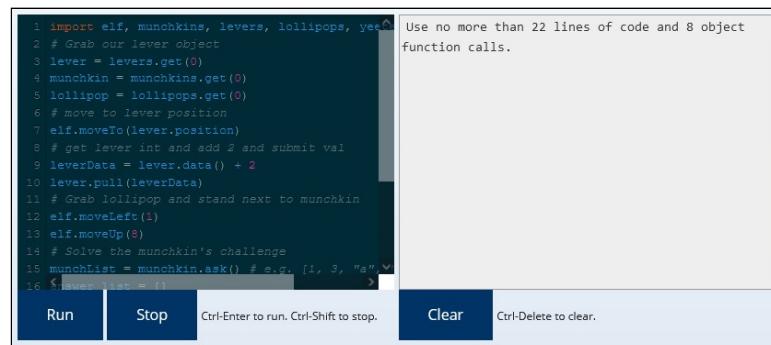


21. Scroll to the Top of the window

22. Close the Elf Code window



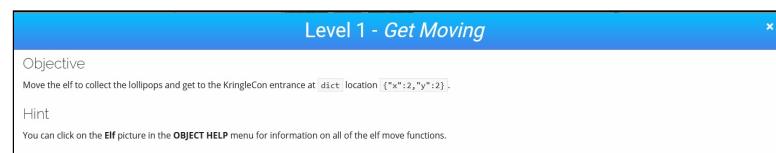
23. Scroll Down the window



24. Scroll to the Top of the window and then press **CTRL + ENTER** to execute the code



25. Click the **Sign**



26. Close the **Level 1 - Get Moving** window



27. Scroll Down the window

The screenshot shows a code editor window with a dark background. A vertical scroll bar is visible on the right side of the code area. At the bottom of the window, there are three buttons: 'Run', 'Stop', and 'Clear'. Below the buttons, there is a status message: 'Ctrl-Enter to run. Ctrl-Shift to stop.' and 'Ctrl-Delete to clear.'

```
1 import elf, munchkins, levers, lollipops, yeeters
2 elf.moveLeft(9)
3
```

Use no more than 8 lines of code and 6 object function calls.

28. Modify the contents of the code window to

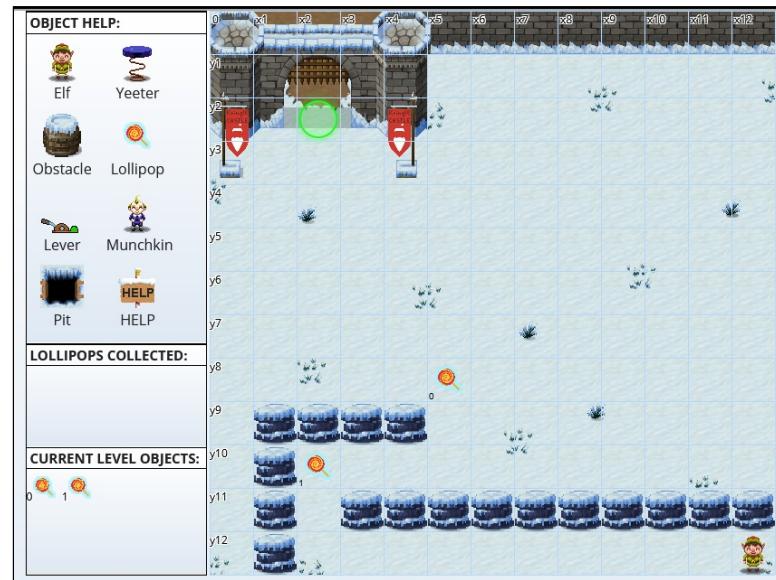
```
import elf, munchkins, levers, lollipops, yeeters, pits
elf.moveLeft(10)
elf.moveTo({"x":2,"y":2})
```

29. Scroll to the Top of the window and then press CTRL + ENTER to execute the code**30. Click the Sign**

The screenshot shows a game objective window with a blue header bar. The title is "Level 2 - Get moveTo 'ing". The window contains the following sections:

- Objective:** Move the elf to collect the lollipops and get to the KringleCon entrance.
- Hint:** The `elf.moveTo` function should help reduce the number of elf move function calls, if used properly.

31. Close the Level 2 - Get moveTo ' ing window



32. Scroll Down the window

```

1 import elf, munchkins, levers, lollipops, yeets
2 # Gets all lollipops as a list
3 all_lollipops = lollipops.get()
4 # Can see lollipop0 using:
5 lollipop1 = all_lollipops[1]
6 # Can also set lollipop0 using:
7 lollipop0 = lollipops.get(0)
8 # or:
9 lollipop0 = all_lollipops[0]
10 elf.moveTo(lollipop1.position)

```

Use no more than 10 lines of code and 6 object function calls.

Run
Stop
Ctrl-Enter to run. Ctrl-Shift to stop.
Clear
Ctrl-Delete to clear.

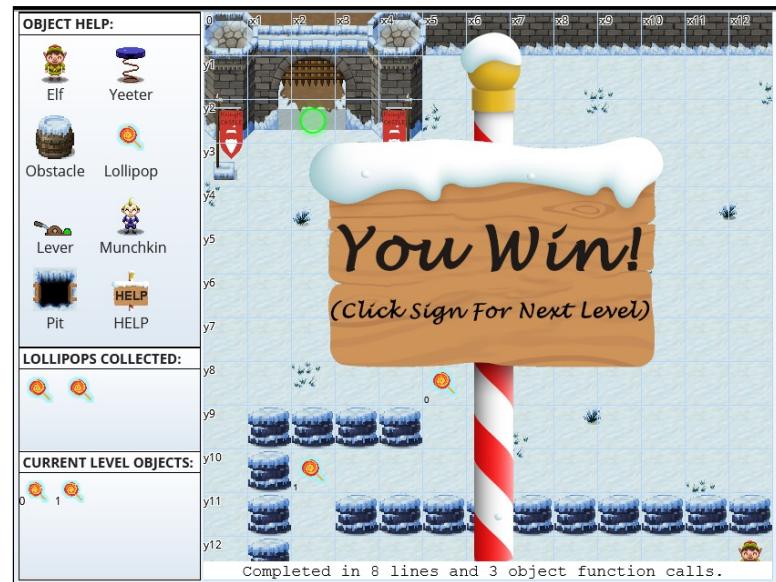
33. Modify the contents of the code window to

```

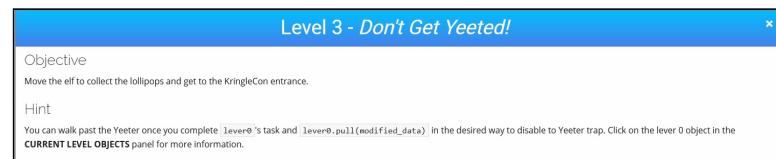
import elf, munchkins, levers, lollipops, yeeters, pits
lollipop1 = lollipops.get(1)
lollipop0 = lollipops.get(0)
elf.moveTo(lollipop1.position)
elf.moveTo(lollipop0.position)
elf.moveTo({"x":2,"y":2})

```

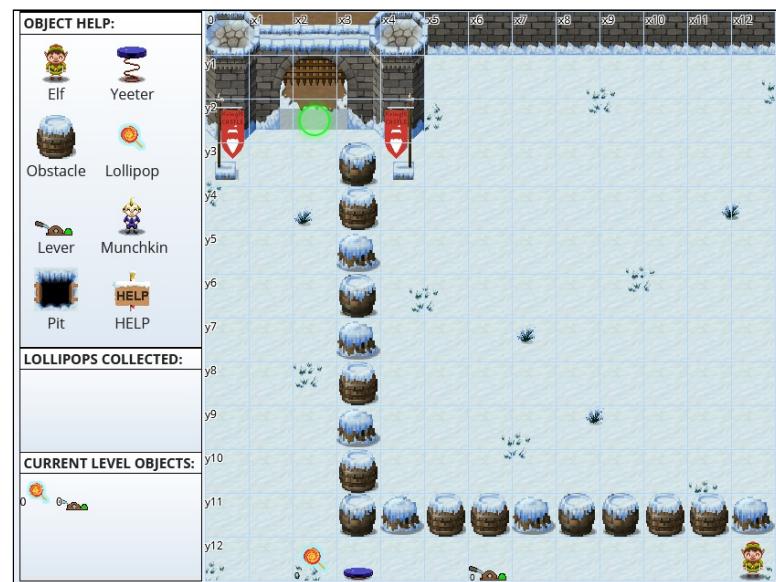
34. Scroll to the Top of the window and then press **CTRL + ENTER** to execute the code



35. Click the **Sign**



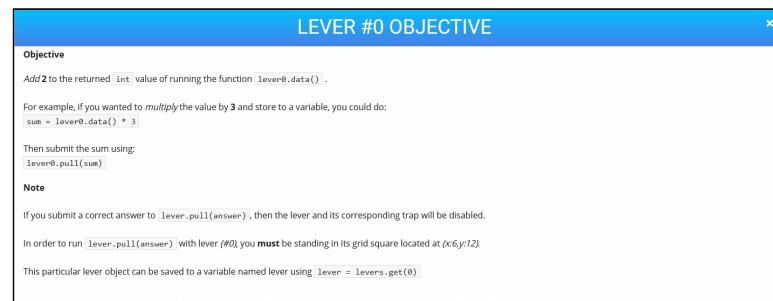
36. Close the **Level 3 - Don't Get Yeeted!** window



37. Scroll Down the window

The screenshot shows a code editor window with a dark background. A horizontal scroll bar is visible at the bottom of the code area. Below the scroll bar are three buttons: "Run", "Stop", and "Clear". To the right of the scroll bar, there is a note: "Ctrl-Enter to run. Ctrl-Shift to stop. Ctrl-Delete to clear." On the far right of the window is a close button (X).

```
1 import elf, munchkins, levers, lollipops, yeeters
2 lever0 = levers.get(0)
3 lollipop0 = lollipops.get(0)
4
```

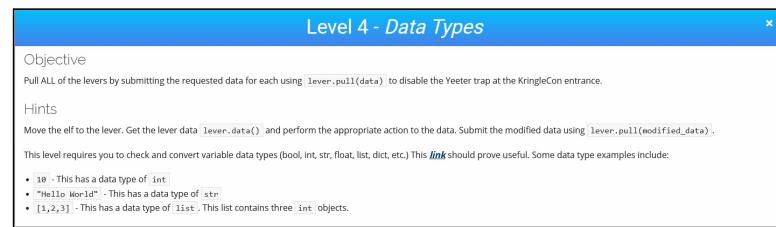
38. Click lever0**39. Close the LEVER #0 OBJECTIVE window****40. Modify the contents of the code window to**

```
import elf, munchkins, levers, lollipops, yeeters, pits
lever0 = levers.get(0)
lollipop0 = lollipops.get(0)
elf.moveTo(lever0.position)
leverData = lever0.data() + 2
lever0.pull(leverData)
elf.moveTo(lollipop0.position)
elf.moveTo({"x":2,"y":2})
```

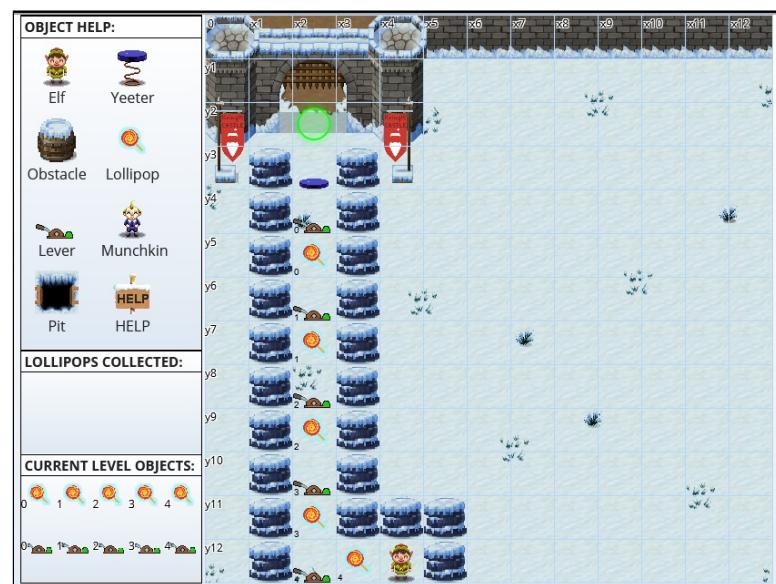
41. Scroll to the Top of the window and then press **CTRL + ENTER** to execute the code



42. Click the **Sign**



43. Close the **Level 4 - Data Types** window



44. Scroll Down the window

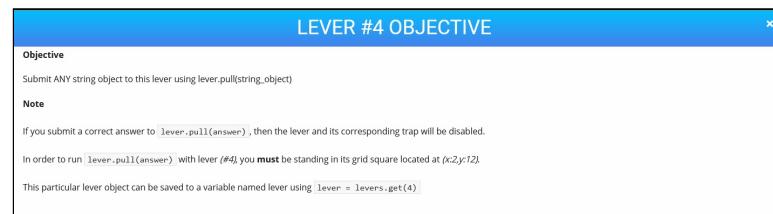
```

1 import elf, munchkins, levers, lollipops, yeets
2 # Complete the code below:
3 lever0, lever1, lever2, lever3, lever4 = levers
4 # Move onto lever4
5 elf.moveTo(2)
6 # This lever wants a str object:
7 lever4.pull("A String")
8 # Need more code below:
9

```

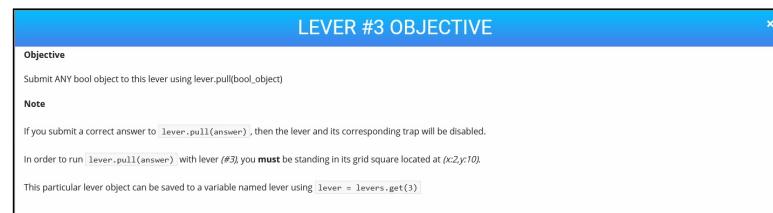
Run | Stop | Ctrl-Enter to run. Ctrl-Shift to stop. | Clear | Ctrl-Delete to clear.

45. Click lever4



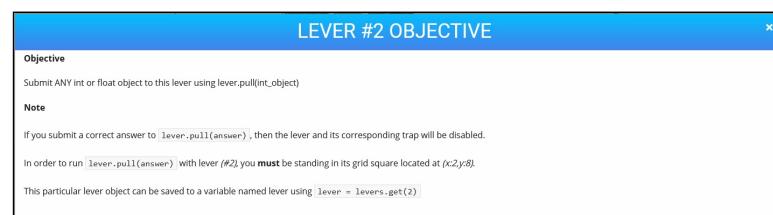
46. Close the LEVER #4 OBJECTIVE window

47. Click lever3



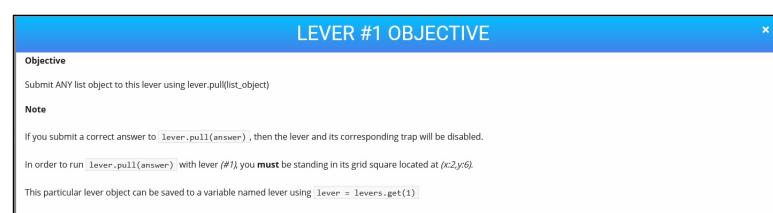
48. Close the LEVER #3 OBJECTIVE window

49. Click lever2



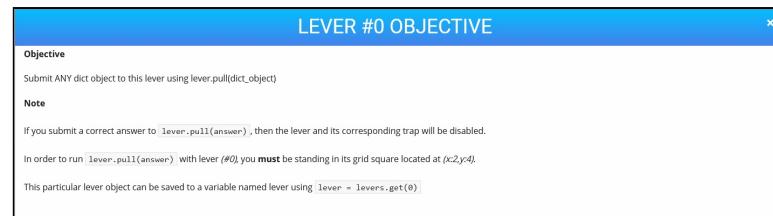
50. Close the LEVER #2 OBJECTIVE window

51. Click lever1



52. Close the LEVER #1 OBJECTIVE window

53. Click **lever0**



54. Close the **LEVER #0 OBJECTIVE** window

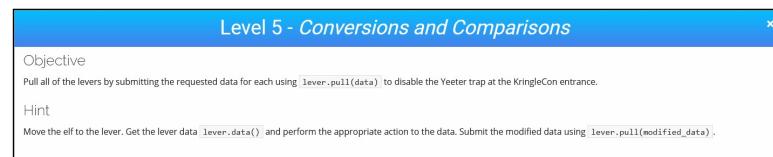
55. Modify the contents of the code window to

```
import elf, munchkins, levers, lollipops, yeeters, pits
lever0, lever1, lever2, lever3, lever4 = levers.get()
elf.moveLeft(2)
lever4.pull("A String")
elf.moveUp(2)
lever3.pull(True)
elf.moveUp(2)
lever2.pull(10.5)
elf.moveUp(2)
lever1.pull([1, 3, "a", "b", 4])
elf.moveUp(2)
lever0.pull({"x":2,"y":2})
elf.moveUp(2)
```

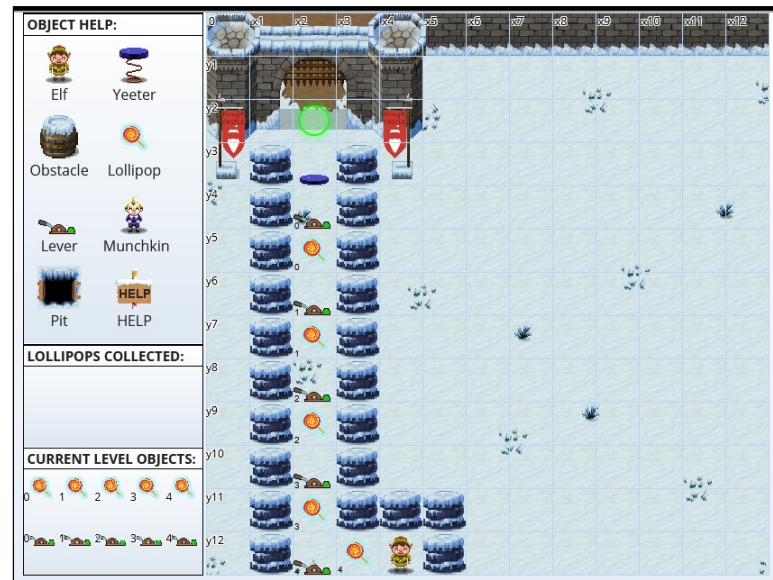
56. Scroll to the **Top** of the window and then press **CTRL + ENTER** to execute the code



57. Click the **Sign**



58. Close the Level 5 - Conversions and Comparisons window



59. Scroll Down the window

```

1 import elf, munchkins, levers, lollipops, yeets
2 # Fix/Complete Code below
3 lever0, lever1, lever2, lever3, lever4 = levers
4 # Solve for each lever, moving to the space
5 # on the lever before calling leverN.pull()
6
7 # Don't forget to move to the KringleCon entrance
8

```

Use no more than 23 lines of code and 18 object function calls.

Run
Stop
Ctrl-Enter to run. Ctrl-Shift to stop.
Clear
Ctrl-Delete to clear.

60. Click lever4

LEVER #4 OBJECTIVE

Objective

Calling `lever.data()` will return a string to you. Take this string and concatenate it with a string of " concatenate". Submit these two concatenated strings using `lever.pull(concatenated_strings)`.

Note

If you submit a correct answer to `lever.pull(answer)`, then the lever and its corresponding trap will be disabled.

In order to run `lever.pull(answer)` with lever (#4), you **must** be standing in its grid square located at (x:2,y:12).

This particular lever object can be saved to a variable named lever using `lever = levers.get(4)`

61. Close the LEVER #4 OBJECTIVE window

62. Click lever3

LEVER #3 OBJECTIVE

Objective

Submit the inverse bool object returned by pulling `lever.data()` to this lever using `lever.pull(inversed_bool_object)`. For example, if `True` is returned, then you must submit `False`.

Note

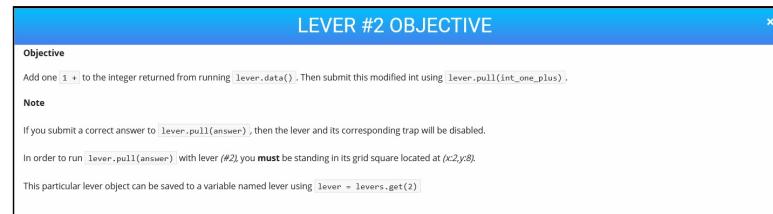
If you submit a correct answer to `lever.pull(answer)`, then the lever and its corresponding trap will be disabled.

In order to run `lever.pull(answer)` with lever (#3), you **must** be standing in its grid square located at (x:2,y:10).

This particular lever object can be saved to a variable named lever using `lever = levers.get(3)`

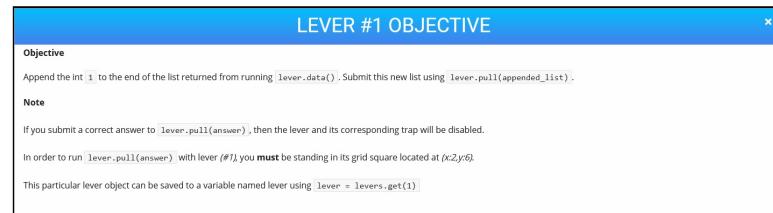
63. Close the LEVER #3 OBJECTIVE window

64. Click lever2



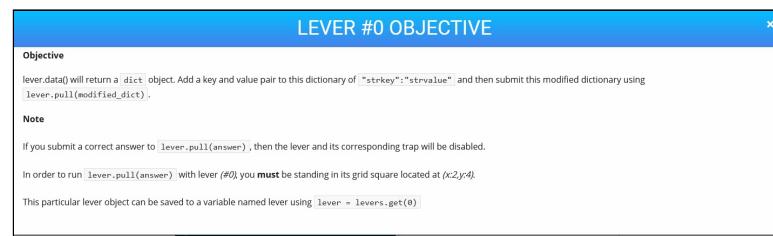
65. Close the LEVER #2 OBJECTIVE window

66. Click lever1



67. Close the LEVER #1 OBJECTIVE window

68. Click lever0



69. Close the LEVER #0 OBJECTIVE window

70. Modify the contents of the code window to

```
import elf, munchkins, levers, lollipops, yeeters, pits
lever0, lever1, lever2, lever3, lever4 = levers.get()
elf.moveLeft(2)
lever4.pull(lever4.data() + " concatenate")
elf.moveUp(2)
lever3.pull(not lever3.data())
elf.moveUp(2)
lever2.pull(lever2.data() + 1)
elf.moveUp(2)
leverData = lever1.data()
leverData.append(1)
lever1.pull(leverData)
elf.moveUp(2)
leverData = lever0.data()
leverData.update({"strkey": "strvalue"})
lever0.pull(leverData)
elf.moveUp(2)
```

71. Scroll to the Top of the window and then press **CTRL + ENTER** to execute the code



72. Click the **Sign**

Level 6 - Types And Conditionals

Objective
Move the elf to the lever. Get the lever data `lever.data()` and perform the appropriate action to the data. Submit the modified data using `lever.pull(modified_data)`.

Hints
This level requires the use of operators to compare and modify data. This [link](#) on operators should help.
Data types will also need to be checked using conditionals in `if`, `elif`, `else` statements. This [link](#) on conditionals should help.
You will also need to use conditionals to check data types. This [link](#) on types should help.
For example, if you want to check the type of a variable, you could use:

```
if type(var) == str:  
    print("It's a string!")
```

73. Close the **Level 6 - Types and Conditionals** window

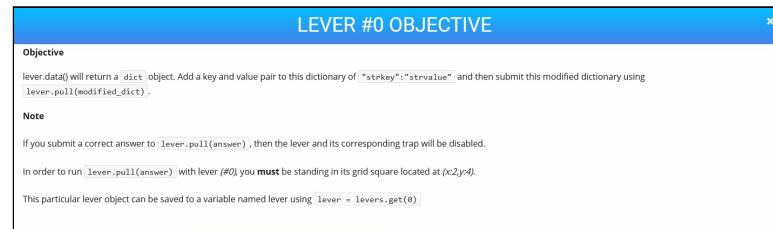


74. Scroll Down the window

```
1 import elf, munchkins, levers, lollipops, yeets
2 # Fix/Complete the below code
3 lever = levers.get(0)
4 data = lever.data()
5 if type(data) == bool:
6     data = not data
7 elif type(data) == int:
8     data = data * 2
9 #elf.move
10 #lever.something
11
```

Run | Stop | Ctrl-Enter to run. Ctrl-Shift to stop. | Clear | Ctrl-Delete to clear.

Use no more than 23 lines of code and 6 object function calls.

75. Click lever0**76. Close the LEVER #0 OBJECTIVE window****77. Modify the contents of the code window to**

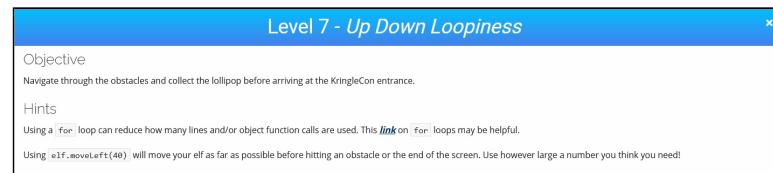
```
import elf, munchkins, levers, lollipops, yeeters, pits
lever = levers.get(0)
data = lever.data()
if type(data) == bool:
    data = not data
elif type(data) == int:
    data = data * 2
elif type(data) == list:
    for i in range(len(data)):
        data[i] = data[i] + 1
elif type(data) == str:
    data = data + data
elif type(data) == dict:
    data['a'] += 1

elf.moveUp(2)
lever.pull(data)
elf.moveUp(2)
```

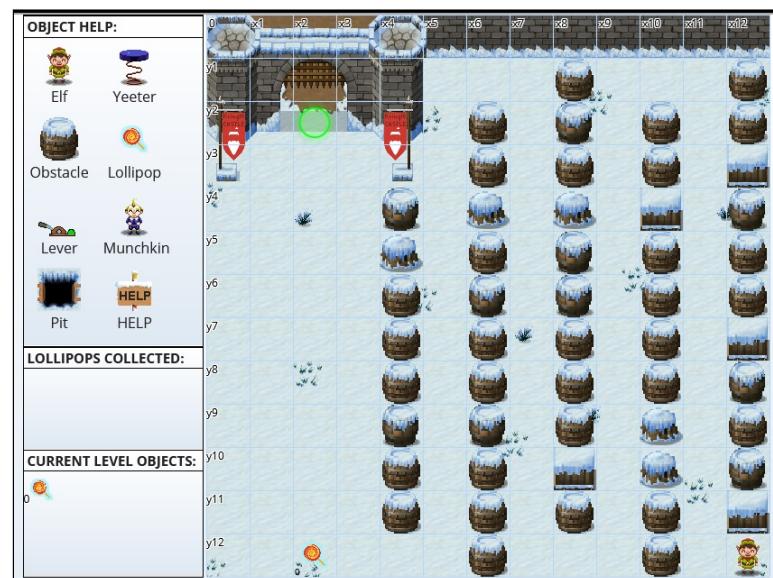
78. Scroll to the Top of the window and then press **CTRL + ENTER** to execute the code



79. Click the **Sign**



80. Close the **Level 7 - Up Down Loopiness** window



81. Scroll Down the window

```

1 import elf, munchkins, levers, lollipops, yeeters, pits
2 for num in range(2): #not sure if number is right
3     elf.moveLeft(3)
4     # needs more code
5

```

Use no more than 12 lines of code and 12 object function calls.

Run | Stop | Ctrl-Enter to run. Ctrl-Shift to stop. | Clear | Ctrl-Delete to clear.

82. Modify the contents of the code window to

```

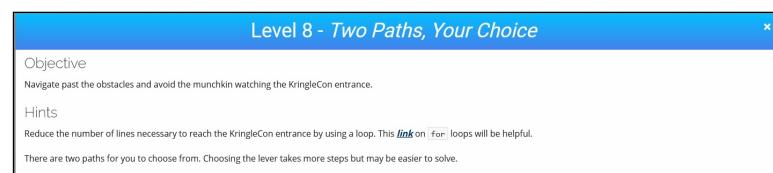
import elf, munchkins, levers, lollipops, yeeters, pits
for i in range(2):
    elf.moveLeft(3)
    elf.moveUp(12)
    elf.moveLeft(3)
    elf.moveDown(12)
elf.moveLeft(3)
elf.moveUp(12)

```

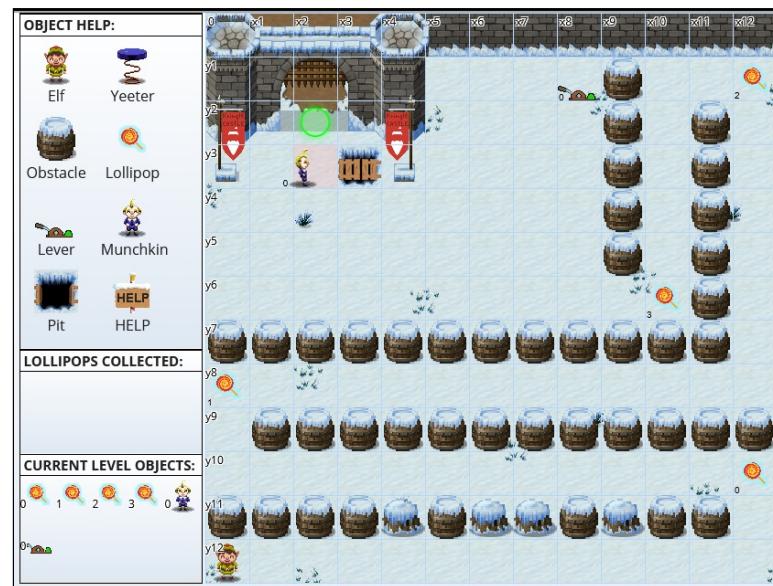
83. Scroll to the Top of the window and then press CTRL + ENTER to execute the code



84. Click the Sign



85. Close the **Level 8 - Two Paths, Your Choice** window



86. Scroll Down the window

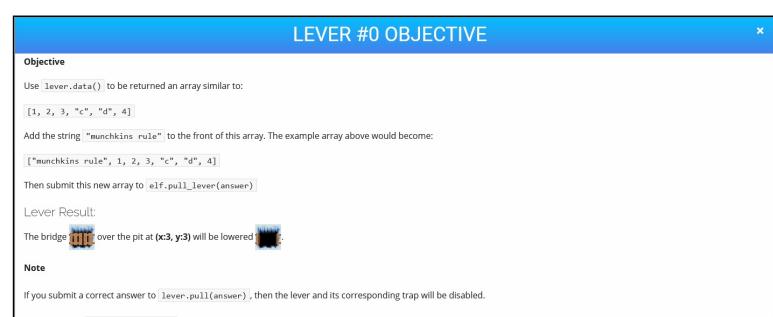
```

1 import elf, munchkins, levers, lollipops, yeeter
2 all_lollipops = lollipops.get()
3 for lollipop in all_lollipops:
4     # needs more code
5 # After loop we solve the lever or munchkin ch
6

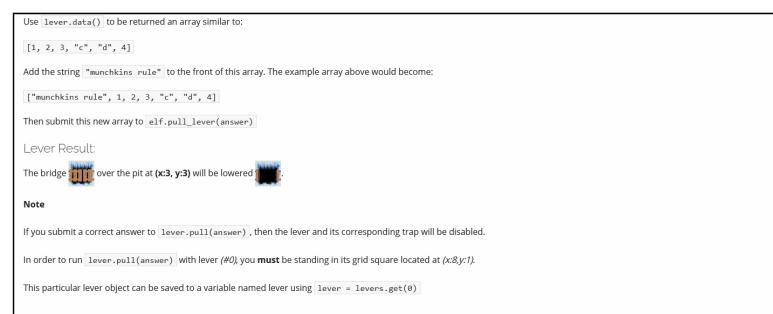
```

Run | Stop | Ctrl-Enter to run. Ctrl-Shift to stop. | Clear | Ctrl-Delete to clear.

87. Click **lever0**

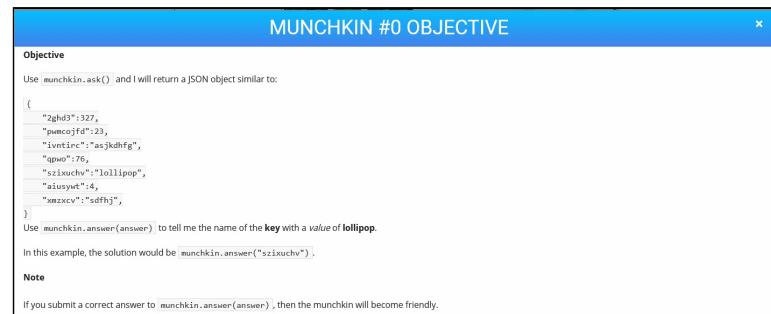


88. Scroll Down the window

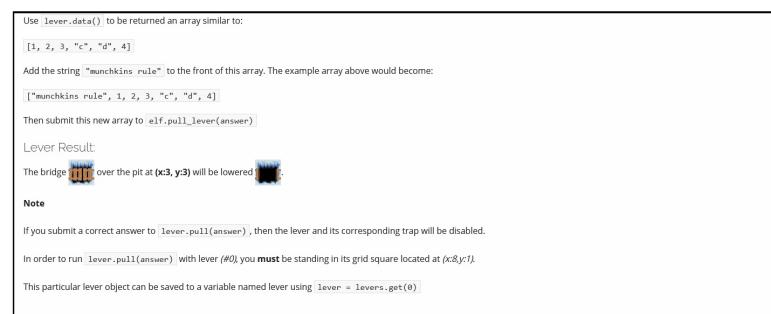


89. Scroll to the **Top** of the window and then close the **LEVER #0 OBJECTIVE** window

90. Click **munchkin0**



91. Scroll **Down** the window

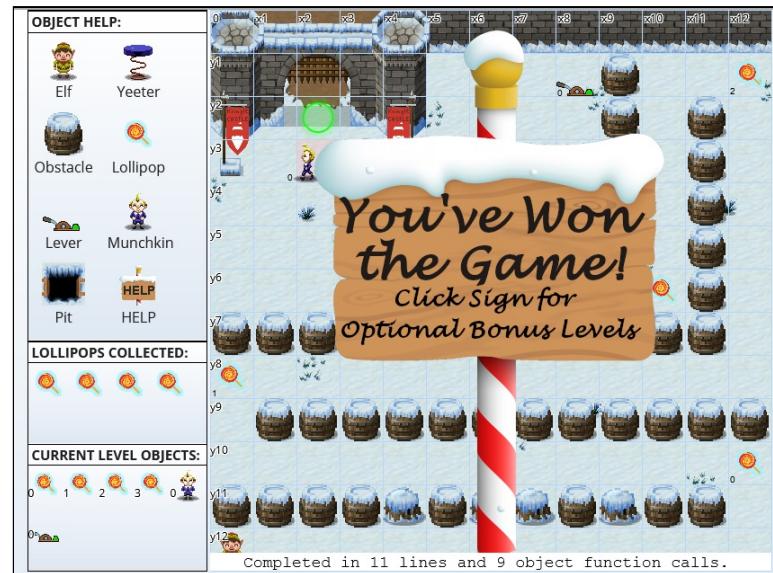


92. Scroll to the **Top** of the window and then close the **MUNCHKIN #0 OBJECTIVE** window

93. Modify the contents of the code window

```
import elf, munchkins, levers, lollipops, yeeters, pits
all_lollipops = lollipops.get()
for lollipop in all_lollipops:
    elf.moveTo(lollipop.position)
lever = levers.get(0)
elf.moveTo(lever.position)
leverData = lever.data()
lever.pull(["munchkins rule"] + leverData)
elf.moveDown(3)
elf.moveTo({"x":2,"y":2})
```

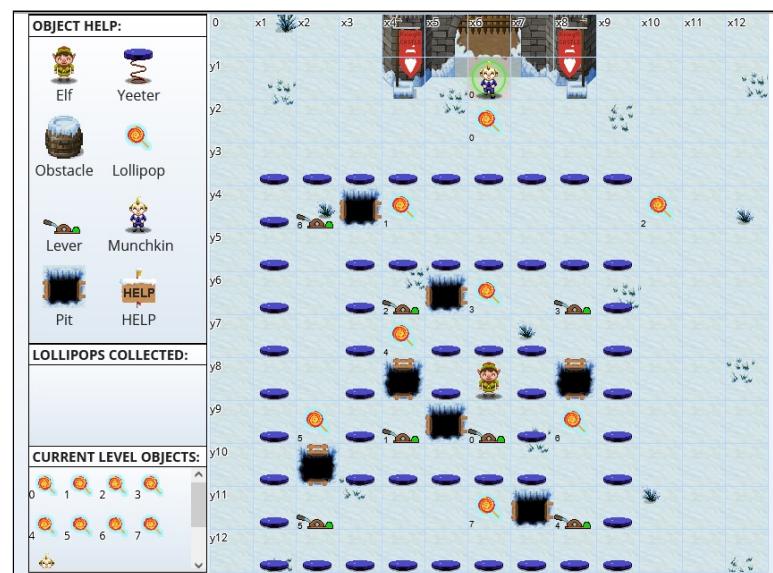
94. Scroll to the Top of the window and then press **CTRL + ENTER** to execute the code



95. Click the **Sign**



96. Close the **Level 9 - Yeeter Swirl** window



97. Scroll Down the window

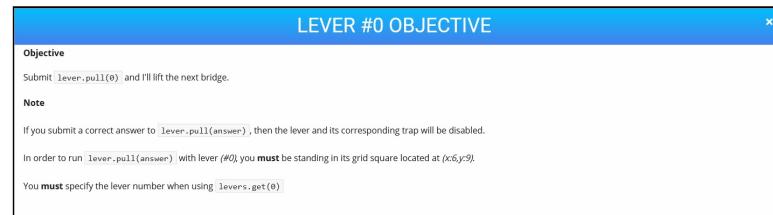
```

1 import elf, munchkins, levers, lollipops, yes
2
3 def func_to_pass_to_munchkin(list_of_lists):
4     sum_of_ints_in_list_of_lists = 0
5     return sum_of_ints_in_list_of_lists
6
7 all_levers = levers.get()
8 # Create Movement pattern:
9 moves = [elf.moveDown, elf.moveLeft, elf.moveUp]
10
11 # We iterate over each move in moves getting ...
12 for i, move in enumerate(moves):
13     # We need to call each move
14     # if i is less than the len(all_levers), we ...
15     <--> # i should be at lever#0. We need to ...

```

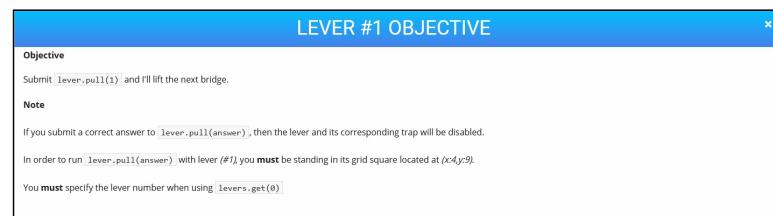
Run | Stop | Ctrl-Enter to run. Ctrl-Shift to stop. Ctrl-Delete to clear.

98. Click lever0



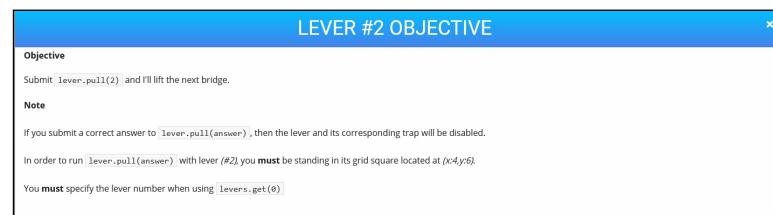
99. Close the LEVER #0 OBJECTIVE window

100. Click lever1



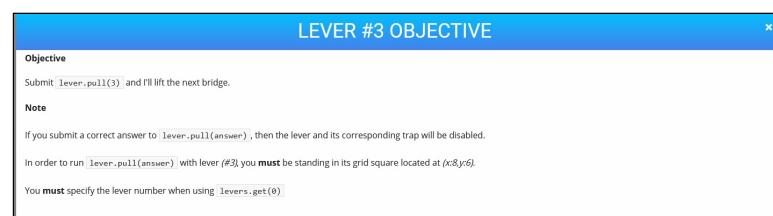
101. Close the LEVER #1 OBJECTIVE window

102. Click lever2



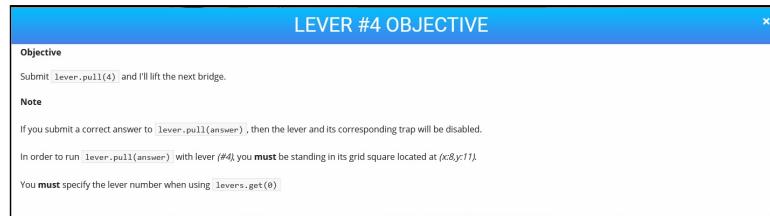
103. Close the LEVER #2 OBJECTIVE window

104. Click lever3



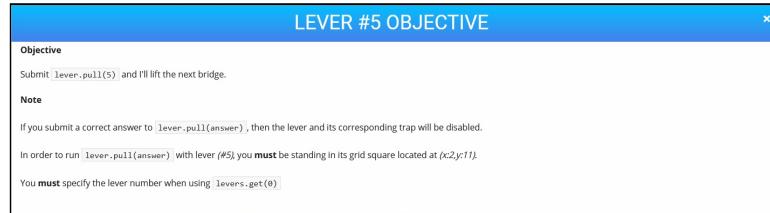
105. Close the LEVER #3 OBJECTIVE window

106. Click lever4



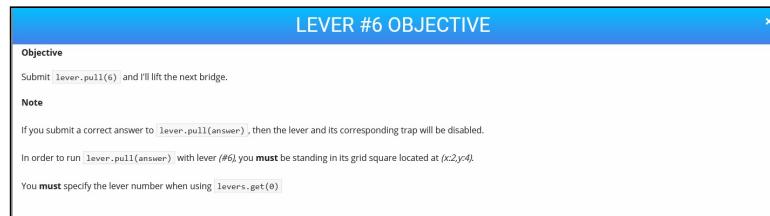
107. Close the LEVER #4 OBJECTIVE window

108. Click lever5



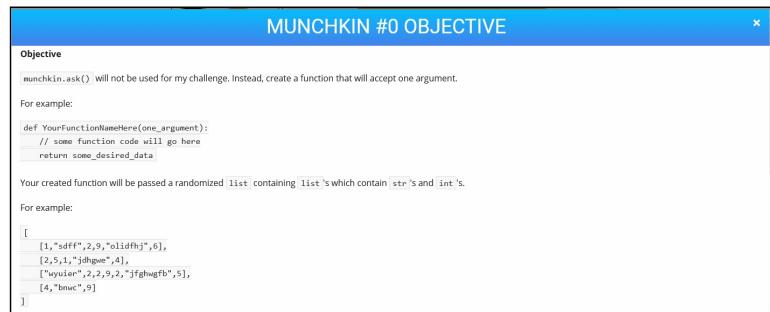
109. Close the LEVER #5 OBJECTIVE window

110. Click lever6

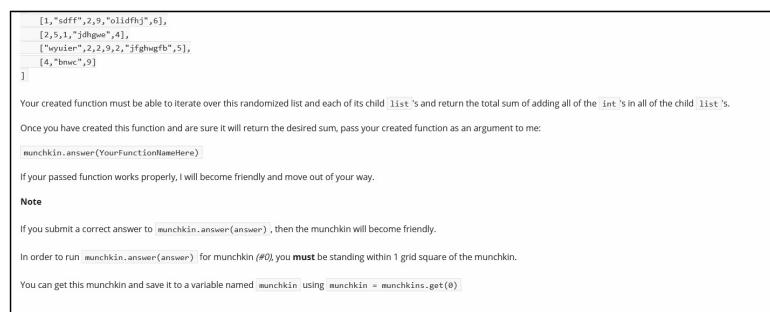


111. Close the LEVER #6 OBJECTIVE window

112. Click munchkin0



113. Scroll Down the window



114. Scroll to the Top of the window and then close the MUNCHKIN #0 OBJECTIVE window

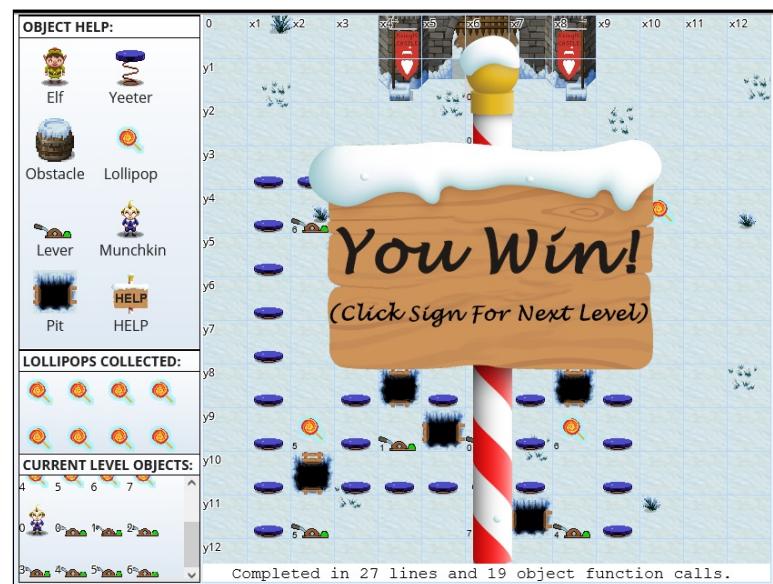
115. Modify the contents of the code window to

```
import elf, munchkins, levers, lollipops, yeeters, pits

def sum(list_of_lists):
    total = 0
    for i in range(len(list_of_lists)):
        for j in range(len(list_of_lists[i])):
            if type(list_of_lists[i][j]) == int:
                total = total + list_of_lists[i][j]
    return total

munchkin = munchkins.get(0)
all_levers = levers.get()
for i in range(7):
    if i == 0 or i == 4:
        elf.moveDown(i + 1)
    elif i == 1 or i == 5:
        elf.moveLeft(i + 1)
    elif i == 2 or i == 6:
        elf.moveUp(i + 1)
    elif i == 3:
        elf.moveRight(i + 1)
        all_levers[i].pull(i)
    elf.moveRight(8)
    elf.moveUp(2)
    elf.moveLeft(4)
    munchkin.answer(sum)
    elf.moveUp(1)
```

116. Scroll to the Top of the window and then press CTRL + ENTER to execute the code



117. Click the Sign

Level 10 - Munchkin Dodging Finale

Objective
Dodge the munchkins to get to the KringleCon entrance.

Hints
You want to move once each munchkin is the furthest grid coordinates away on the x axis (which will be -6 for each munchkin). Use while loops and implement a conditional check using the `elf.position["x"]` and `munchkin.position["x"]` values to check how far away the munchkin is before using `moveTo` to the next lollipop position. When using while loops, use a small delay of `time.sleep(0.05)` to ensure the browser does not lock up.

118. Close the Level 10 - Munchkin Dodging Finale window



119. Scroll Down the window

```

1 import elf, munchkins, levers, lollipops, yeeters
2 import time
3 muns = munchkins.get()
4 lols = lollipops.get()[:-1]
5 for index, mun in enumerate(muns):
6     # need to wait while absolute distance betw
7     # elf.position["x"] and mun.position['x'] i
8     # then we move to next lollipop
9     # We can use time.sleep(0.05) to add a sma
10

```

Use no more than 17 lines of code and 15 object function calls.

 Ctrl-Enter to run. Ctrl-Shift to stop.
 Ctrl-Delete to clear.

120. Modify the contents of the code window to

```

import elf, munchkins, levers, lollipops, yeeters, pits
import time
muns = munchkins.get()
lols = lollipops.get()[:-1]
for index, mun in enumerate(muns):
    while abs(elf.position["x"] - mun.position['x']) < 6:
        time.sleep(0.05)

        elf.moveTo(lols[index].position)

    elf.moveTo({"x":2,"y":2})

```

121. Scroll to the Top of the window and then press **CTRL + ENTER** to execute the code



New [Achievement] Unlocked: Elf Code Python Bonus Levels!!
[Click here to see this item in your badge.](#)

122. Click the **Close** button

123. Click to talk to **Ribb Bonbowford**

Gosh, with skills like that, I'll bet you could help figure out what's really going on next door...

And, as I promised, let me tell you what I know about SQL injection.

I hear that having source code for vulnerability discovery dramatically changes the vulnerability discovery process.

I imagine it changes how you approach an assessment too.

When you have the source code, API documentation becomes tremendously valuable.



New [Hint] Unlocked: SQL Injection with Source!
[Click here to see this item in your badge.](#)

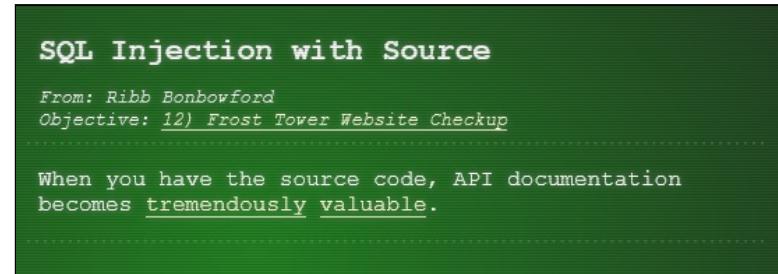
124. Click to talk to **Ribb Bonbowford**

Who knows? Maybe you'll even find more than one vulnerability in the code.

Wow - even the bonus levels! That's amazing!

125. Click the **i** (Hints) icon

126. Click **SQL Injection with Source**



127. Click **[Exit]**

128. Move **Up** and enter the **Courtyard**