Mission 8:
Answer Bot

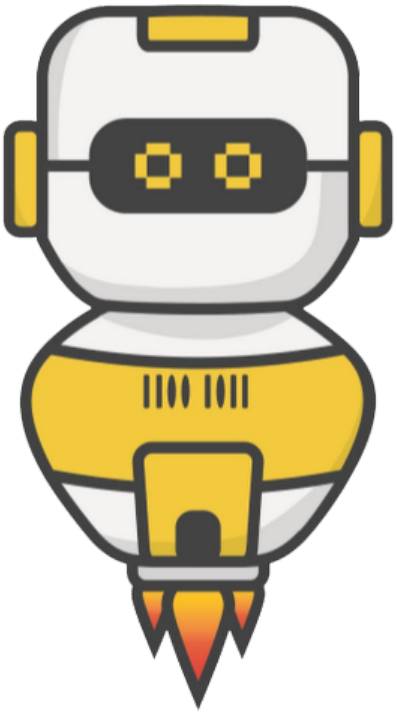
Student Workbook

FIRIA LABS

Mission 8: Answer Bot

Use a list to create a random answer generator.

Let's get random

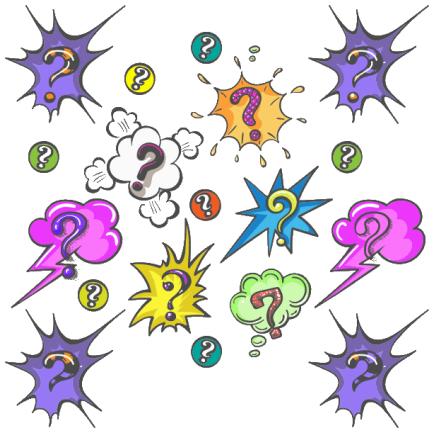


In the last mission, the person running the program had control over what happened. In real-life, sometimes values need to be random.

Go to the Mission 8 Log and fill out the Pre-Mission preparation.

- What are some examples of when you might need something random?

Mission 8: Answer Bot



In this project you will create a random answer generator.

- Instead of selecting messages yourself, like in the previous project, you will have the computer decide for you!
  → Just press a button and let your **Answer Bot** decide :-)

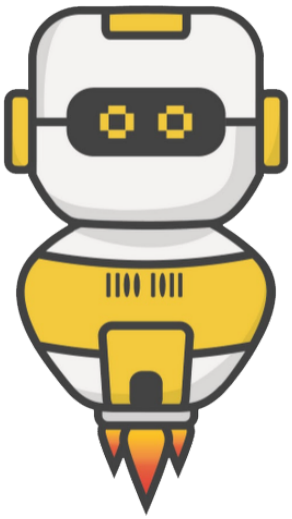
Mission 8: Get started

- Go to https://make.firialabs.com/ and log in.
- Go to Mission 8 
- Click NEXT and start Mission 8.

Objective #1: Display a number

Review displaying an integer from Mission 4

- Start this project by writing code that will:
  - Assign a variable an integer value
  - Display the variable



DO THIS:

- Start a new file named **Answer_Bot**
- Import the codex module
- Assign **number** the value 1
- Use **display.show()** with number
  - Use CodeTrek if you need help
- This will cause an error – do you recognize the error before you run the code?

Answer_Bot ×

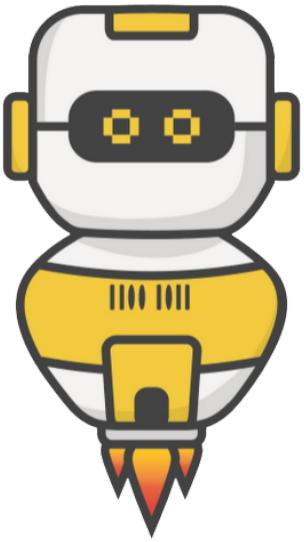
```
1  from codex import *
2
3  number = 1
4  display.show(number)
```

Objective #2: Fix it up

In Mission 4, you learned two ways to fix the error:

- Convert (change) the integer to a string
  - str(number)
- Use display.print(number)
  - Automatically converts the integer to a string

Objective #2: Select more images



DO THIS:

- Change **display.show(number)** to **display.print(number)**

```
1   from codex import *
2
3   number = 1
4   display.print(number)
```

Objective #3: Randomize!

Python has a **random** module that has built-in functions

- Import the **random** module to access the functions
- One of the built-in functions is **randrange**
- The function call looks like this:
  - **random.randrange(end_value)**
- The **randrange** function returns a random integer between 0 and one less than the end value
- Examples:
  - **number = random.randrange(10)**
  - Gives a random number from 0 to 9
  - **number = random.randrange(5)**
  - Gives a random number from 0 to 4

The **randrange** function generating a random integer between 0 and one less than the end value is really handy!

- The **list index** starts at 0 and goes to one less than the number of items
- If you have a list with 6 items, you can get a random index by using

```
index = random.randrange(6)
```

- Or, avoid the magic number:

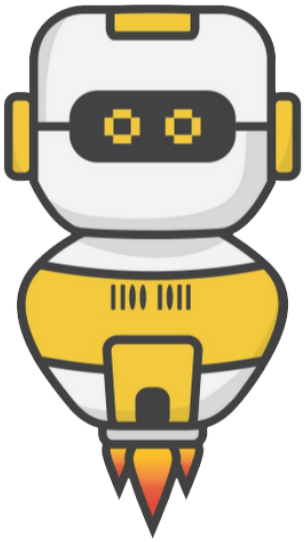
```
index = random.randrange(len(my_list))
```

Objective #3: Randomize!

One more thing: You can make your print bigger!

- **display.print()** has an argument for scale
  display.print(number, scale=1)
- scale=1 is the normal size
- scale=2 makes the text bigger
- scale=3 makes the text even bigger
- If you make the scale too big, the text won't fit on the screen
- Instead it will look like weird shapes
    - If this happens, you know the scale is too big

Objective #3: Randomize!



DO THIS:

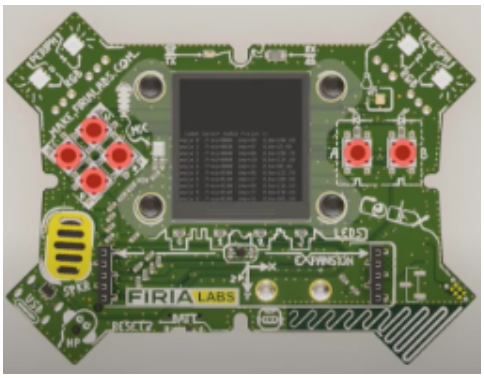
- Change the value of **number** to a random number between 0 and 9
- Add a scale argument to the display.print() statement: start with **scale=1**
- Run the code, then change the scale: **scale=3**
- Run the code, then change the scale to a different number
- Run the code several times. You should get a different random number each time.
- NOTE: *Sometimes you may see the same number repeat, but that's all part of the randomness!*

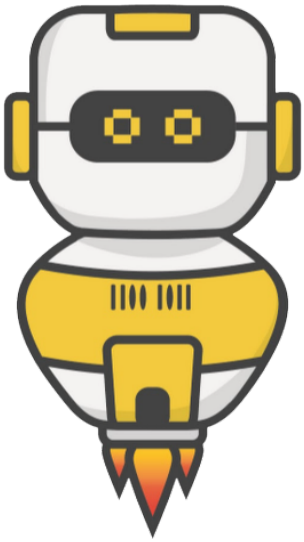
What number goes here?

```
from codex import *
import random

number = random.randrange( ? )
display.print(number, scale=1)
```

Objective #4: Mix things up

- You can improve the code by using a CodeX button and a loop.





DO THIS:

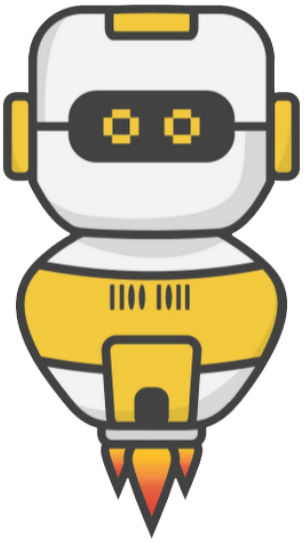
- Go to your Mission Log and write down what you remember about CodeX buttons and loops

Objective #4: Mix things up

Running the program every time you want a random number isn't very fun.

Modify your code to:

- Use a loop
- Get a random number every time BTN_A is pressed



Modify your code

DO THIS:

- Add a **while True:** loop
- Add code to check for a button press (BTN_A)
- Run the code and press BTN_A several times
- You should see a random number each time you press the button

```
from codex import *
import random

while True:
    if                                :
        number = random.randrange(10)
        display.print(number, scale=3)
```

Add code for checking a button press

Objective #5: Robot opinion

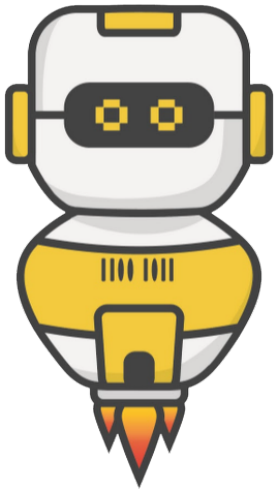
Now that you can display a random number, can you display a random text from a list?

- Time to give CodeX an opinion!
- Think of a question you want CodeX to answer
- Then display a random answer
- This is a perfect place for a list
- Use a random number for the index

This is your Answer Bot, so you can choose the question you want it to answer

- It could be
    - Favorite sports team
    - Best singer / band
    - Favorite food
    - Magic 8 Ball answers
    - Best subject in school
    - ***You*** decide!

Objective #5: Robot opinion



DO THIS:

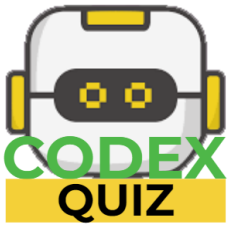
- Go to the Mission Log and plan your question and several possible answers
- Modify the code by creating a list in your code with the possible answers
- Assign a variable the number of items (len of list)

Put your own question and possible answers here

```
from codex import *
import random

# What's for lunch?
answers = ["Pizza", "Burger", "Salad",
           "Burrito", "Nothing", "Pasta"]
count = len(answers)
```

- Modify the if statement to:
  - Get a random number for the index
  - Assign a variable the value from the list
  - Display the list item variable (adjust the scale if needed)

```
while True:
    if buttons.was_pressed(BTN_A):
        index = random.randrange(count)
        my_choice = answers[index]
        display.print(my_choice, scale=3)
```



Mission Quiz: Get some answers

Test your skills by **taking the quiz**.

Objective #6: Wait for answer

- The CodeX module has a built-in list for colors - no need to create one.

```
# Built into the 'colors' module
COLOR_LIST = [
    BLACK, BROWN, RED, ORANGE, YELLOW,
    GREEN, BLUE, PURPLE, GRAY, WHITE,
    CYAN, MAGENTA, PINK, LIGHT_GRAY, DARK_GREEN,
    DARK_BLUE,
]
```

- You do not need to type this in your code
- You can use this list to select a random color for the pixels.
- Let the pixels cycle through random colors while you wait for an answer!

Working with the built-in COLOR_LIST:

- **len(COLOR_LIST)** will give you the number of items
- Use **len(COLOR_LIST)** to get a random number

```
number = random.randrange(len(COLOR_LIST))
```

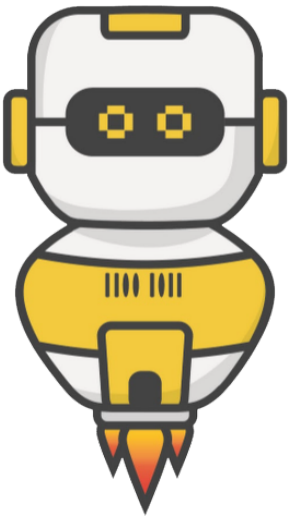
- Use the random number to get a color from the list

```
color = COLOR_LIST[number]
```

- Use the color to turn on the pixels

```
pixels.set(0, color)
pixels.set(1, color)
pixels.set(2, color)
pixels.set(3, color)
```

Objective #6: Wait for answer



Put it all together!

DO THIS:

- Import sleep
- Get a number for a random pixel color
- Use the number to get a color from the COLOR_LIST
- Use the color to turn on all the pixels
- Use a short sleep to make the pixels flash

```
while True:
    # Flashy pixels
    number = random.randrange(len(COLOR_LIST))
    color = COLOR_LIST[number]
    pixels.set(0, color)
    pixels.set(1, color)
    pixels.set(2, color)
    pixels.set(3, color)
    sleep(0.25)

    if buttons.was_pressed(BTN_A):
```

Objective #7: Choices, choices

Four flashing pixels is ... flashy!

- What about making the four pixels flash different colors instead of the same color?
- You just have to assign a random color to each pixel
- This could take a few lines of code, so...
- There is a simpler way

Another built-in random function

- You already know about **random.randrange()**
- Another built-in function is **random.choice()**
- It will randomly pick an item from a list –
  - No need to use a number or index variable
- It looks like this:

```
color = random.choice(COLOR_LIST)
```

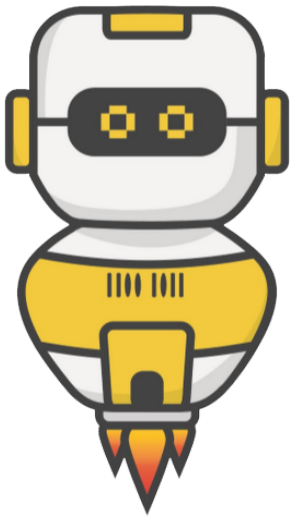
- You can use this command, one for each pixel:

```
# Flashy pixels
color = random.choice(COLOR_LIST)
pixels.set(0, color)
color = random.choice(COLOR_LIST)
pixels.set(1, color)
color = random.choice(COLOR_LIST)
pixels.set(2, color)
color = random.choice(COLOR_LIST)
pixels.set(3, color)
```

- Do you think you can use **random.choice()** for your **answers** list as well?

```
my_choice = random.choice(answers)
display.print(my_choice, scale=3)
```

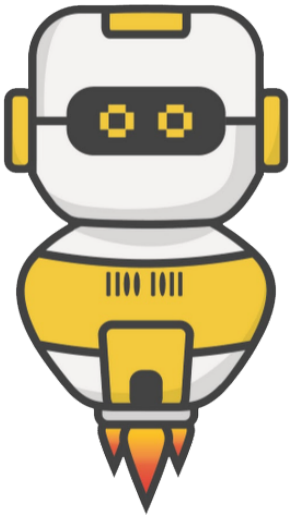
Objective #7: Choices, choices



DO THIS:

- Delete the two lines of code that use **random.randrange()**
  - One for color
  - One for answers
- Delete the **count** variable (not needed anymore)
- Modify the code for **color** and **my_choice** to use **random.choice()**
- Get four random colors and use each one in a different pixel
- Use the code snippets on the previous page if you need help

Objective #7: Choices, choices



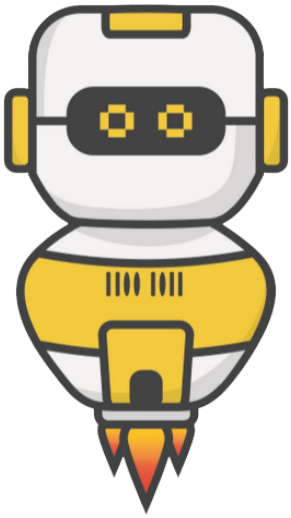
DO THIS:

- Does your code look similar to this?
- Did you pass off your two goals?

```
from codex import *
import random
from time import sleep

# What's for lunch?
answers = ["Pizza", "Burger", "Salad",
           "Burrito", "Nothing", "Pasta"]

while True:
    # Flashy pixels
    color = random.choice(COLOR_LIST)
    pixels.set(0, color)
    color = random.choice(COLOR_LIST)
    pixels.set(1, color)
    color = random.choice(COLOR_LIST)
    pixels.set(2, color)
    color = random.choice(COLOR_LIST)
    pixels.set(3, color)
    sleep(0.25)

    if buttons.was_pressed(BTN_A):
        my_choice = random.choice(answers)
        display.print(my_choice, scale=3)
```

Mission Complete

You have completed the eighth mission.



Do this:

- Read your "Completed Mission" message
- Complete your Mission 8 Log
    - Post-Mission Reflection
- Get ready for your next mission!

Wait! Before you go ... Clear the CodeX

Go to FILE -- BROWSE FILES

Select the "**Clear**" file and open it

Run the program to clear the CodeX

Okay. Now you can go.