CIRCUITS AND ELECTRONICS : ANALOG AND DIGITAL

Names: _____

Instructions: Copy and complete this document and post a link to it from your class site. (One form per group)
If you are an experienced coder and your partner also has some experience, you can skip to the end of the document and solve any one of the challenges.

Value: 5 points

- 3 pts : Exercises are correct or at least attempted for full credit.
- 2 pt : Relatively equal participation from both partners

Learning Goals:

- What is Python
- How to collaborate on Python code
- Printing
- Commenting
- Some common variables
- If else statements
- Comparison operators

---

1. What is Python? In a couple of sentences, describe why the Python language was created and and what is it used for?

Python is a coding language where you can program apps, games, websites etc. It is kind of like Scratch which I used to use a lot but it has more features and relies on text rather than blocks

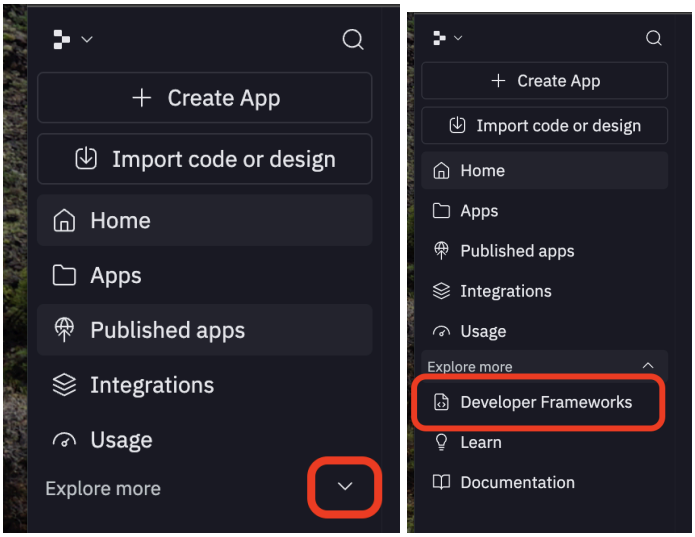
---

2. Get setup on a collaborative cloud tool for writing Python:

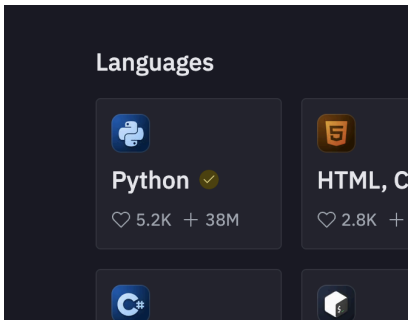
REPL = Read-Eval-Print-Loop

- Start by creating an account https://replit.com

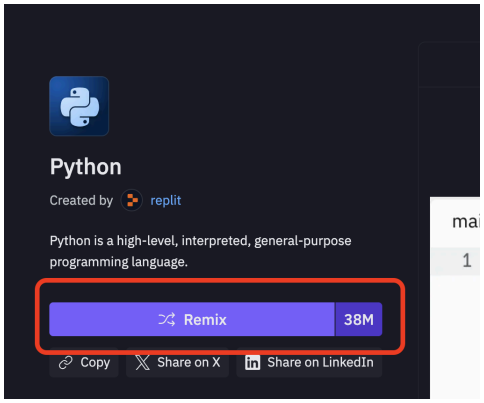
- Once you're logged in, from the menu on the left, click on the "Explore More" arrow and choose Developer Frameworks



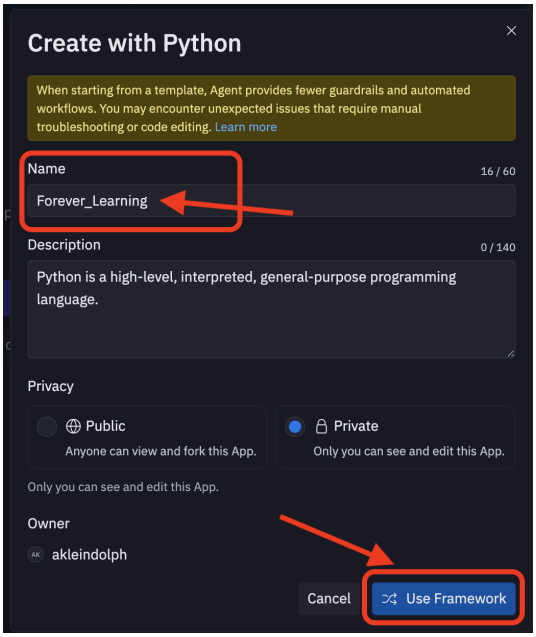
- Choose Python under languages



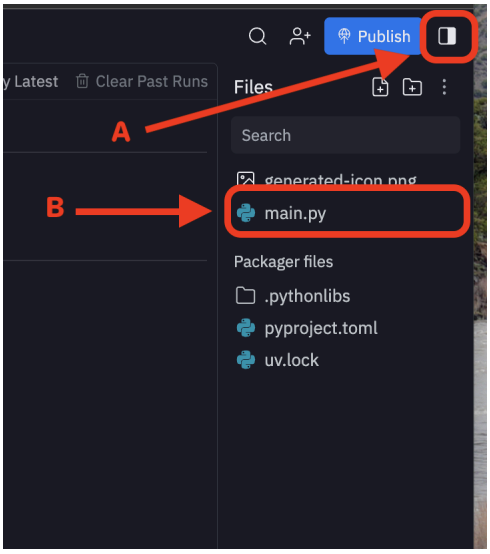
- Click the Remix button



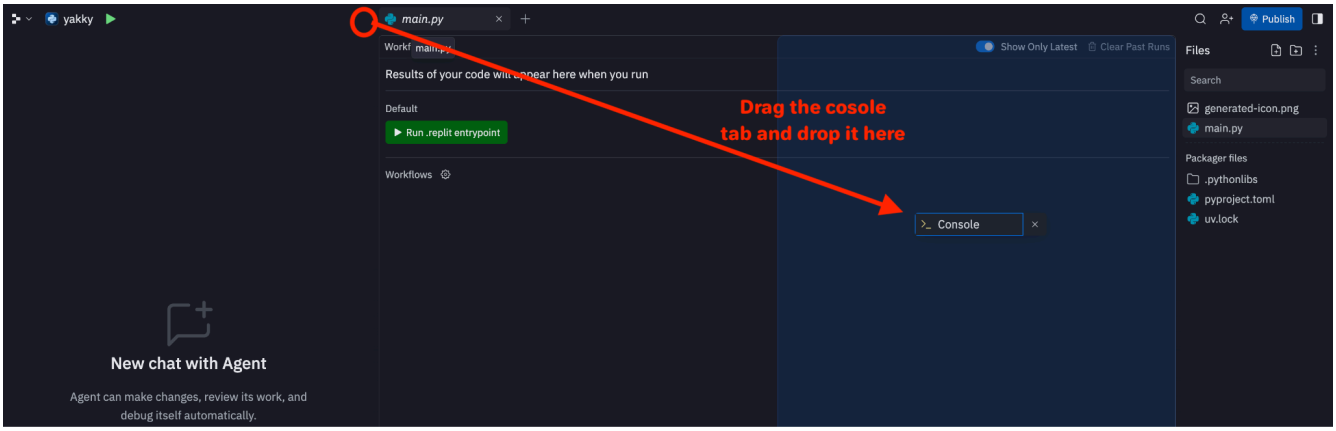
- Give your project a name and press Use Framework



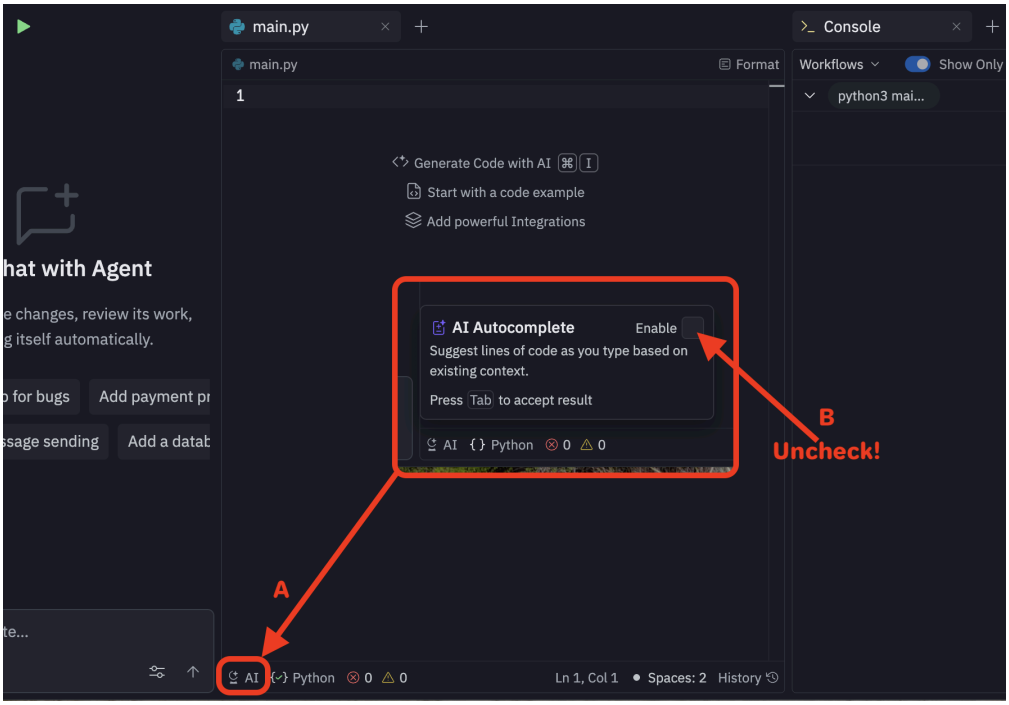
- You will now see a console for your project, but you need to open a main.py file to start editing. Click on the black and white square icon way in the upper right corner to open your files, then click on main.py to start editing that file.



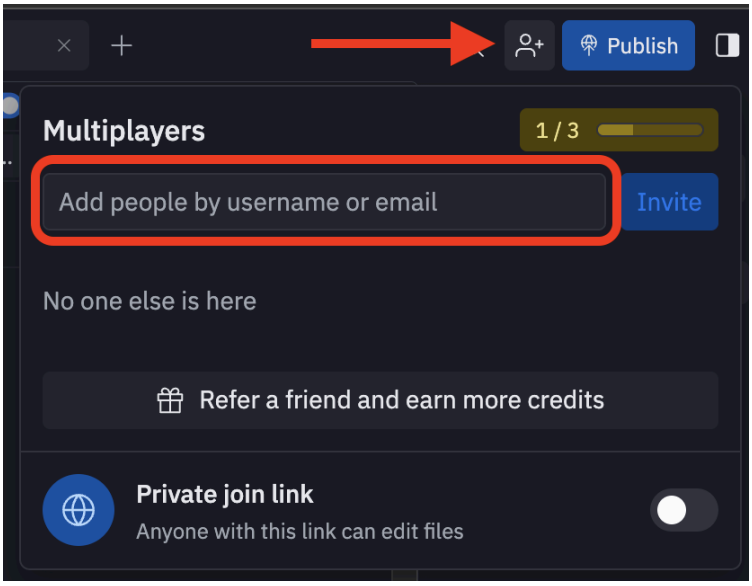
- You want to see your code editor and console at the same time. Drag and drop the console tab to the left of the editor.



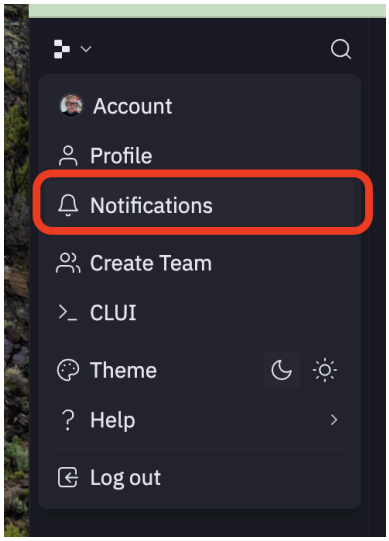
- Turn off the AI autocomplete, which is at the bottom of the editing window.



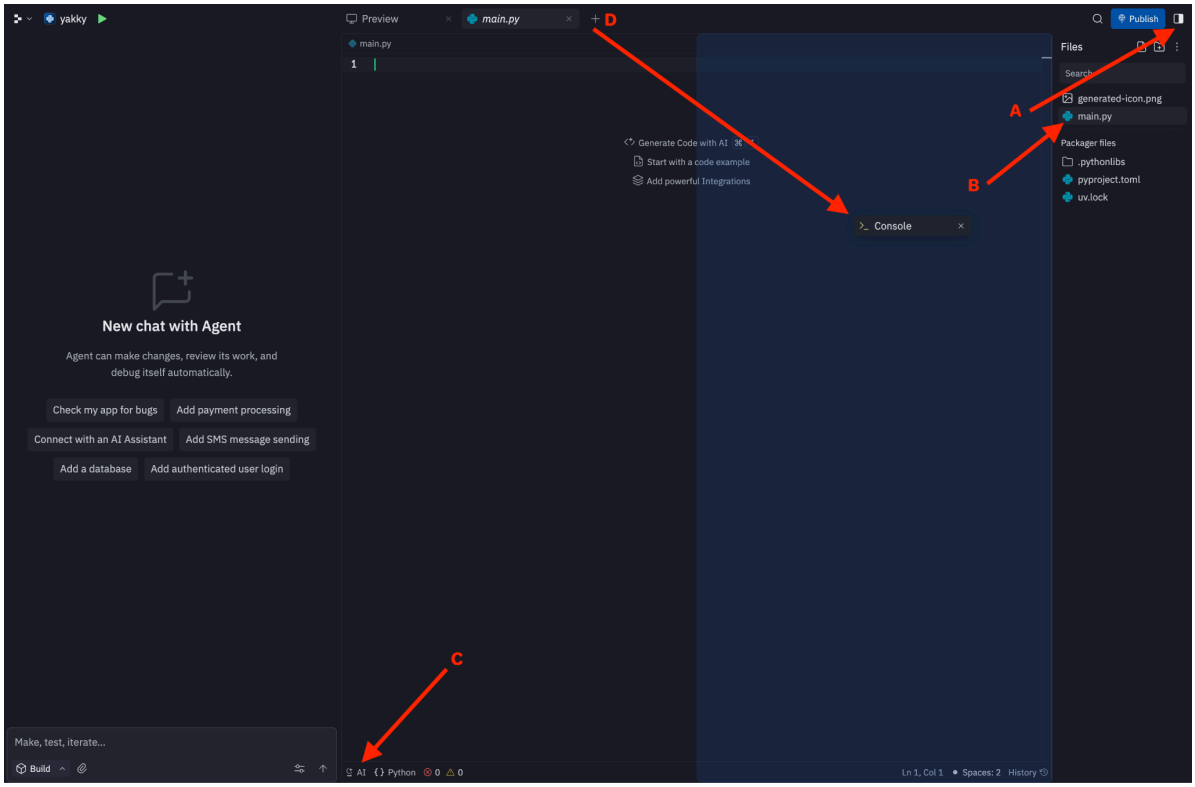
- Using the person icon in the upper left, invite your partner by sending them a link or by adding them via username to the Repl
  (you can only invite one collaborator with the free version)



Once you send the invite, it can take up to 60 seconds for the other person to receive it. They should check their Replit account notifications or their email.



- The person accepting the collaboration request must:(all shown above.)
  - open the file browser
  - click on main.py
  - turn off the AI autocomplete
  - open a console



- FINALLY, Try typing anything in the window until both of you can see the other user.

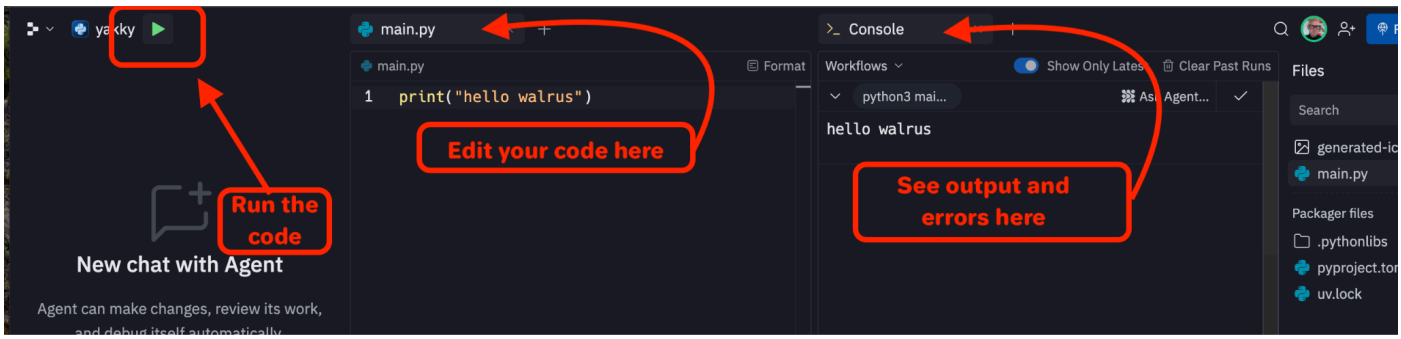
---

3. Printing in Python

Printing in Python is easy! Here's an example.

- Make sure your console is open
- Type what you want to print with this syntax

  print ("anything you want")
- Press the green run button



Try the input() function—this is where things get interactive
Ask for someone's name and print it.

```python
name = input("What is your name? ")
print(name)
```

Add a screenshot of what you and your partner printed:

---

4. Commenting

When coding, what is the purpose of commenting? Examples here

It can be used to make personal notes (the code will not execute the code)

How do you make a single-line comment in python?

#Comment

How do you make a multiline comment in python?

You would have to press enter and then add the second comment

#comment
#comment cont.

5. Some common variables

Explain, in one phrase, each of these variable types:

int stands for an integer

Float stands or a decimal

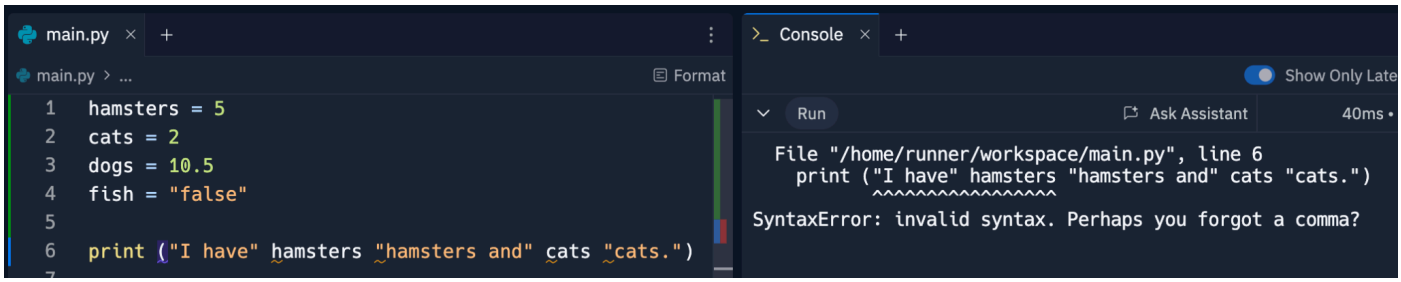
str strings a group into text

bool represents a boolean value which can mean true or false

---

6. Use some variables and data types

Here's an example where we say how many pets we have (too many BTW). However, you will get an error when you run this code.



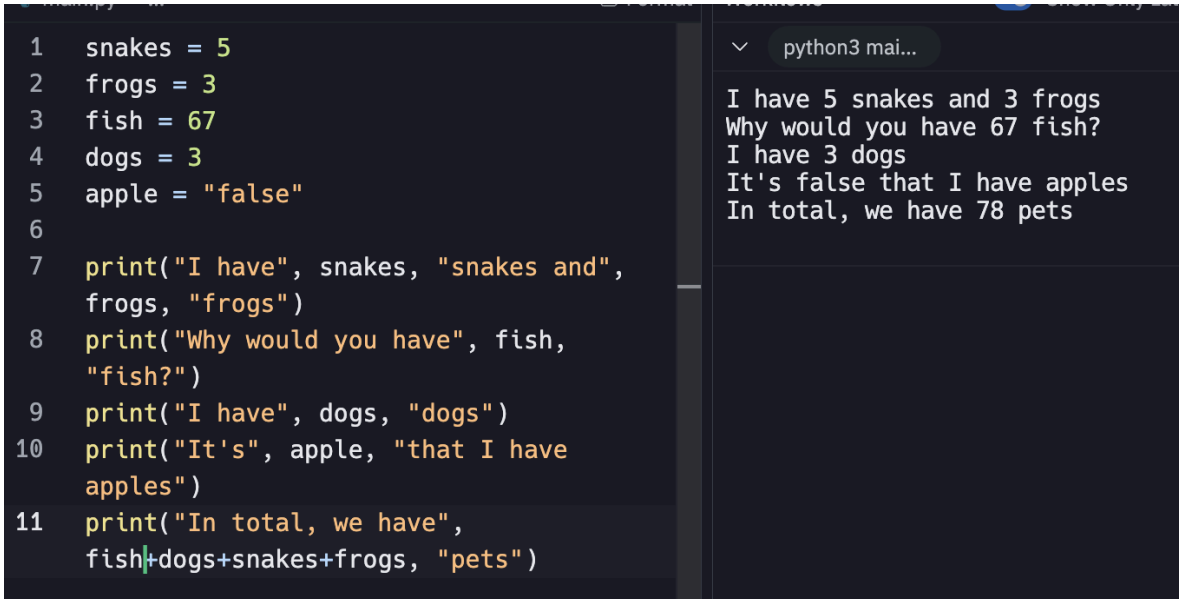
Because we have different types of data, strings which are characters in quotes, and integers (numbers), we need to use a comma when we change data types.



Now, continue this code by printing all the following:

- How many dogs do you have?
- Do you have any fish? Think of a sentence structure that will accommodate the answer by referencing the variable.
- Add one more pet type and number to your collection
- Create a sentence that adds all your pets together.

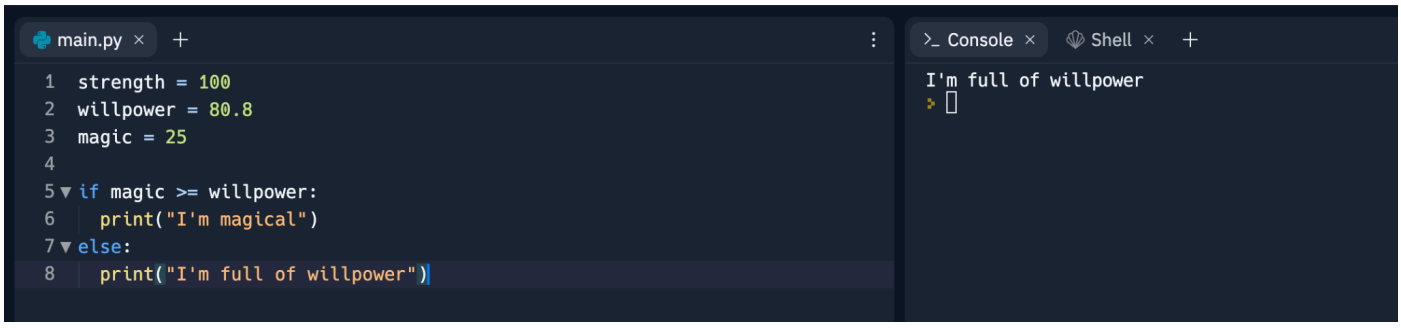
Screenshot your code here:



7A. Comparison operators and if statements

In Python, indentations mean something. We indent to show that a command is part of a loop or a conditional statement. See an example here

Look at this example. Notice how the print commands are indented, and only one is executed based on the True or False nature of the if statement.



Create code that does the following:

- You are a character with: Persuasion, Luck, Influence, and Stealth
- Set those variables to any numeric value you want
- Create two "if else" statements that compare at least two of the characteristics and print something about you.
- Use two different comparison operators. Your choices are (== , >= , <= , >, <)

Screenshot your code here:

```
1  magic = 6
2  strength = 7
3
4  if strength >= magic:
5    print("strength is greater than
   magic")
6  else:
7    print("strength is not greater than
   magic")
8
```

7B: Using “and” and “or” in if statements

Modify the example code from the last exercise to use a “and” and “or” in some if statement. Create code that does the following:

- Check to see if strength OR magic is greater than willpower. Print a statement if True.
- Check to see if both willpower AND strength are greater than magic. Print a statement if True.

Find examples here.

Screenshot your code here:

```python
magic = 10
strength = 9
willpower = 8

if strength >= magic:
  print("strength is greater than magic")
else:
  print("strength is not greater than magic")

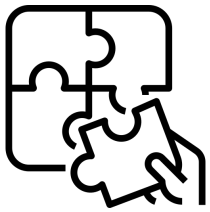
if strength or magic  >= willpower:
  print("strength or magic is greater than willpower")

if willpower and magic >= strength:
   print("willpower and magic are greater than strength")
```

```
strength is not greater than magic
strength or magic is greater than willpower
willpower and magic are greater than strength
```

For those who have some extra time:

Can you figure out any part of this puzzle?



1. Find an online resource that shows you how to randomize values.
2. Randomize a set of characteristics (look up how to randomize in python)
3. Executing the code will trigger a “battle” where two characters with randomly generated values are compared in multiple ways; then, an outcome is printed explaining which character won and why.

Screenshot your code here:

Python exercises for folks with some prior knowledge:

1. Personal Report

Objective:
Create a Python script that generates a fun, personalized report based on user input. Make the report look like it's from a "Secret Agent" mission briefing, with some lighthearted additions!

Sections to Include:

A. Mission Title: A cool, mysterious title provided by the user, displayed in all caps with a dynamic, centered border.
B. Agent Details:
    a. Agent's Code Name (string), e.g., "Agent Thunderbolt."
    b. Today's Date in a "Mission Initiated" format, e.g., "Mission Initiated: YYYY-MM-DD."
C. Mission Summary:
    a. Brief description of the mission (string), like "Retrieve the Golden Artifact."
    b. Total number of "mission objectives" (integer), validated to ensure it's a positive integer.
    c. Success rate percentage (float) to two decimal places, representing your "Mission Success Probability."

Requirements:

- Use f-strings and/or **`format()`** to align and style each line creatively.
- Include error handling to ensure valid data entries for all fields. If a user enters an invalid input (like a string for an integer field), prompt them to "Rethink and enter the correct data."
- Display a fun "Mission Complete" message at the end, with a rating based on the success probability.

2. Potion Shop Inventory Tracker

Objective:
Create a Python program that simulates an inventory system for a fantasy potion shop. Your program should use all four variable types (`int`, `float`, `str`, `bool`) in a fun, meaningful way.

Details:

1. Potion Name (`str`): Store the name of a magical potion, like "Elixir of Swiftness."
2. Stock Quantity (`int`): Track how many of these potions are left in stock.
3. Potion Price (`float`): Store the price per potion in gold coins.
4. Availability (`bool`): Indicate whether the potion is currently available for sale (e.g., `True` for available, `False` for sold out).

Functions to Include:

- `add_stock()`: Add potions to the stock.
- `sell_potion()`: Sell a potion, decrease stock, and print a fun success message if available.
- `check_availability()`: Print a message indicating if the potion is available.
- `display_info()`: Display all potion details in a user-friendly way.

3. Turn-based game

Objective:

Create a simple turn-based battle game using Python classes to model characters and actions. Each character will have randomized stats, and the game will proceed in rounds until one character wins.

Steps:

A. Import Required Modules
   a. Use the **`random`** module to generate random attributes.
B. Create a **`Character`** Class
   Define a **`Character`** class that includes:
   a. Attributes:
      i. **`name`** (**`str`**): The character's name.
      ii. **`health`** (**`int`**): Initial health, randomized between 50 and 100.
      iii. **`strength`** (**`int`**): Physical attack power, randomized between 10 and 20.
      iv. **`magic`** (**`int`**): Magical attack power, randomized between 5 and 15.
      v. **`defense`** (**`int`**): Defense value, reducing damage taken, randomized between 5 and 10.
   b. Methods:
      i. **`attack()`**: Randomly choose between a strength or magic attack.
      ii. **`take_damage(amount)`**: Reduce health by **`amount`**, considering the defense attribute.
C. Game Loop Logic
   a. Use a loop to alternate turns between two characters, each choosing to either attack or defend.
   b. For each round:
      i. Display each character's stats.
      ii. Prompt the player to choose an action (attack or defend).
      iii. Calculate damage, taking into account the attacker's attribute and defender's defense.
      iv. Display the outcome, including health updates.
   c. End the game when one character's health reaches 0, declaring the other character as the winner.