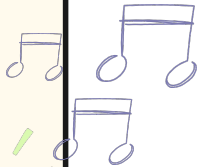


CHOOSE YOUR OWN ADVENTURE GAME

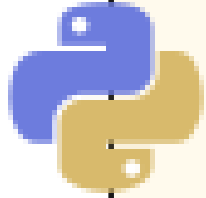
# CYOA BUILDING

STORY

GUI



MUSIC

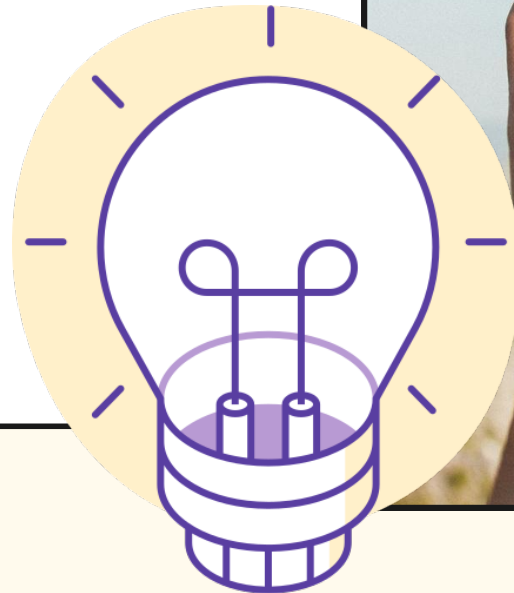


PYTHON



CHOOSE YOUR OWN ADVENTURE GAME

**A CLOSER  
LOOK:  
WHY ARE THEY  
SO POPULAR?**



# WHAT WE'LL USE



## REPLIT

- Make or log into an account on [replit.com](https://replit.com) or open the app
  - It is an IDE
  - Available on EVERY platform!
- Open a new Repl
  - Name it
  - Choose Python as your programming language



## PYTHON

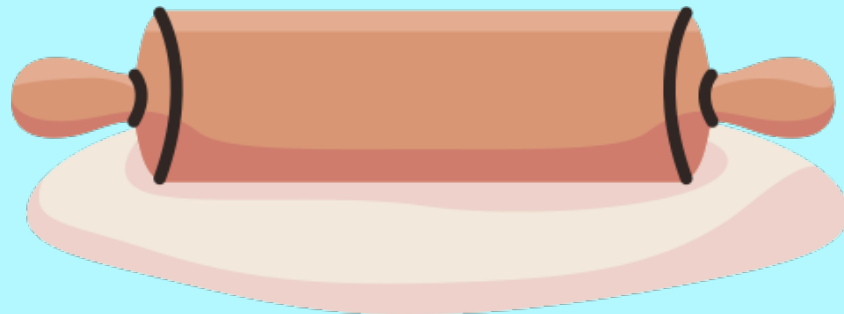
### Why Python?

- Simplicity + Readability
- Beginner Friendly
- Quick development and iteration

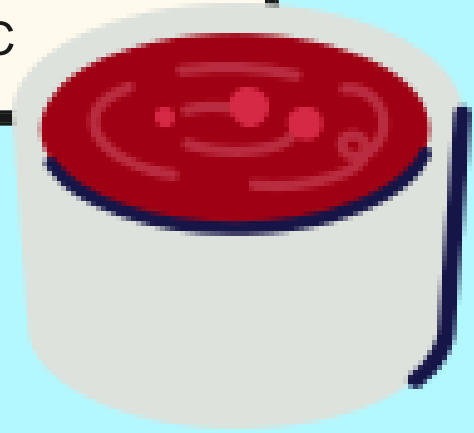


# GAME PLAN

1 STORYLINE + MAIN LOGIC



2 STYLE + MUSIC



3 CREATING A GUI





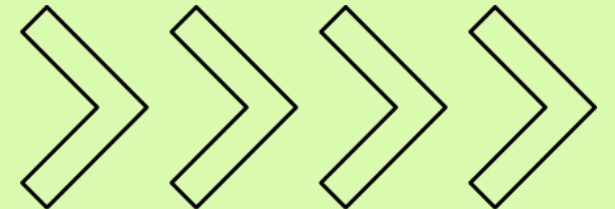


CHOOSE YOUR OWN ADVENTURE GAME

# LEVEL 1:

STORYTELLER'S  
FORGE

Create your Text-Based  
Adventure Game

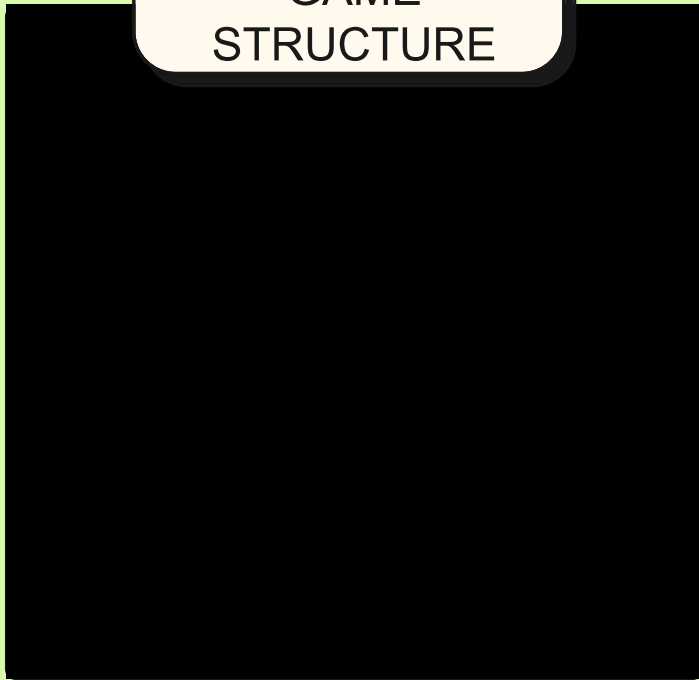


# GAME DESIGN

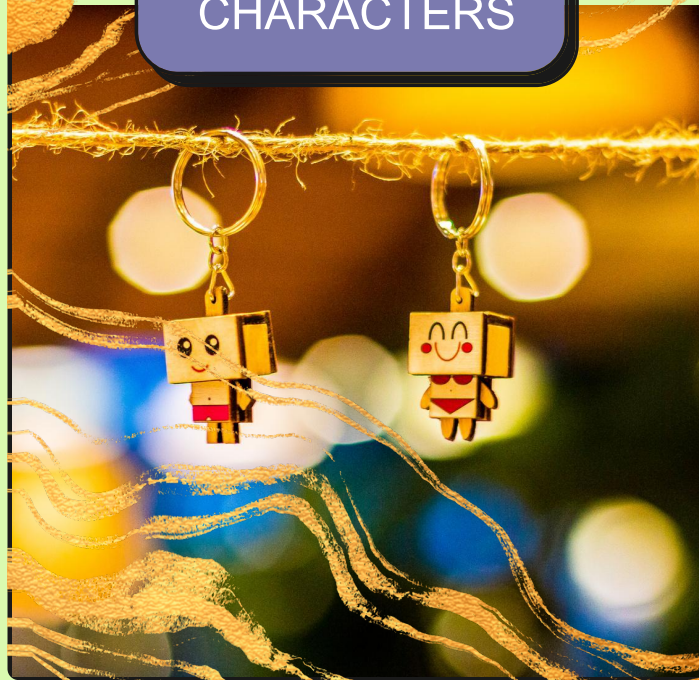
THEME/  
SETTING



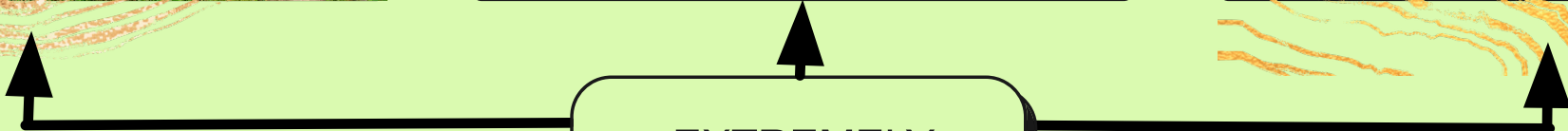
GAME  
STRUCTURE



CHARACTERS



EXTREMELY  
IMPORTANT!






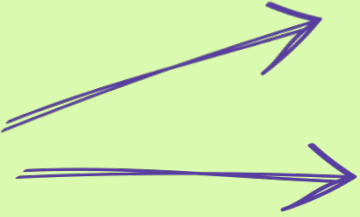
# CONDITIONALS

if, elif, and else statements:

enable your program to perform different actions or execute different blocks of code based on whether a condition is true or false!

## EXAMPLE

```
1  if (insert condition here):  
2      (do something here)  
3  
4  elif (insert condition here):  
5      (do something here)  
6  
7  else:  
8      (do something here)
```



# WHILE LOOPS

a repetitive loop that runs  
until a certain condition  
becomes false



## EXAMPLE

```
1 while (insert condition here):  
2     (do something here)
```

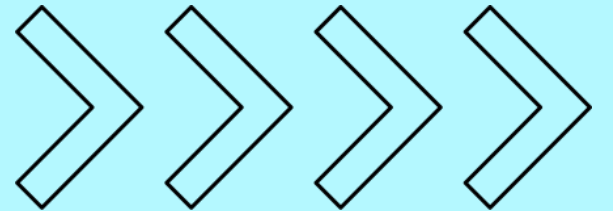


CHOOSE YOUR OWN ADVENTURE GAME

# LEVEL 2:

## SONIC SAGAS

Adding Music and Style to your  
Text Based Adventure Game



# IMPORTING PACKAGES

Packages provide more functions and classes that are not part of the original language.



## EXAMPLE

```
1 import (package here)
```



# DEFINING FUNCTIONS

Functions: named blocks of reusable code that perform a certain task.

Def Functions: define function behavior, name, and parameters

## EXAMPLE

```
1 def (function name here):  
2     (do something here)
```



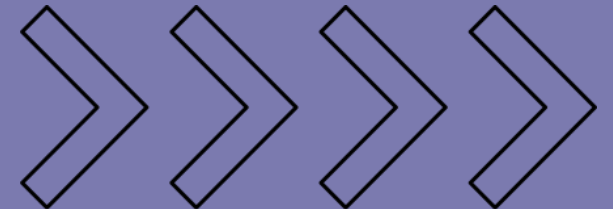


CHOOSE YOUR OWN ADVENTURE GAME

# LEVEL 3:

## GUI: FINAL BOSS

Creating the GUI





# WIDGETS

Displays information or a way for a user to interact with a platform. Great for consistency, is reusable, simple, and efficient!

## EXAMPLE

```
1 variable = tk.goal_output(customize  
2             here)  
3 variable.pack()
```

What can goal output be?   
There are 18 in total, but here's a few!





# LABELS

Like sticky notes! Display fixed text or info in a GUI, giving context for other elements like buttons or input fields.

## EXAMPLE

```
1 variable = tk.Label(customize)
2
3 variable.pack()
```



# INPUTS

Text entry forms that users can type into and make selections with.

## EXAMPLE

```
1 variable = tk.Entry(customize)
2
3 variable.pack()
```



# BUTTONS

Packages provide more functions and classes that are not part of the original language.

## EXAMPLE

```
1 variable = tk.Button(customize)
2
3 variable.pack()
```

Let's move on from widgets

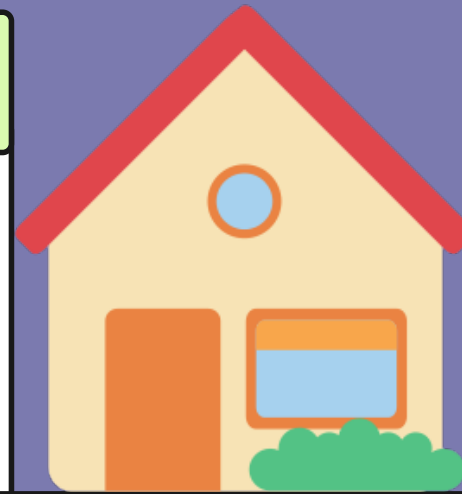




# FRAMES + WINDOWS

Windows: main screens of your application's interface, has many frames to create a user-friendly layout. (house)

Frames: containers for grouping widgets, and organizes elements like buttons and text boxes in a window. (rooms)



VS



## EXAMPLE

```
1 window_var = tk.Tk()  
2 window_var.title("title")  
3 window_var.geometry("# x #")  
4 window_var.resizable(T, or F)  
5  
6 frame_var = tk.Frame(window_var)
```

# GET BUILDING!

