

girls who
code

Girls Who Code At Home

Python Poetry

Activity Overview



National Poetry Month is a celebration across the world. During the month of April, millions of people celebrate the importance of poetry and recognize the contributions that poets have made to our world. Poetry, whether you are writing it, reading it or listening to it can be incredibly powerful and moving. Poems like Maya Angelou's, [*Still I Rise*](#), and Amanda Gorman's, [*The Hill We Climb*](#), have inspired countless people with messages of resilience and hope.

In this Code at Home activity, you will bring awareness to the art of poetry and code. You'll start this project by choosing a poem that is meaningful to you. Then, you will learn how to format and display your poem using the coding language, Python. Finally, you will create a poem generator that will allow others to personalize the poem you are spotlighting. Before you start, be sure to check out the featured Women in Tech Spotlight, Joy Buolamwini, who many people call, "The Poet of Code."

Learning Goals

By the end of this activity you will be able to...

- ❑ note the differences between different types of poems, such as rhymed poems, free verse, Haikus, and Sonnets.
- ❑ use `print` statements in Python to format a poem.
- ❑ store `input` from a user into a variable to generate different versions of your poem.

Materials

- [Replit Editor](#)
- [Python Poetry Sample Project](#) (Using Multiple Print Statements)
- [Python Poetry Sample Project](#) (Using Newline)
- [Python Poetry Reference Guide](#)
- [Extension #1](#): Add More Variables
- [Extension #2](#): Add in Tab Command
- [Extension #3](#): Add in Wait Command

Prior Knowledge

- No prior knowledge necessary!

Women in Tech Spotlight: Joy Buolamwini



Image Source: [Vital Voices](#)

Joy is the founder of the [Algorithmic Justice League](#), a movement with a mission to create a world with more inclusive technology. During graduate school Joy worked with a facial recognition application. While working with this app, she found that in order for her own face to be recognized, she had to wear a white mask. Since this upsetting experience, she has spent her time researching the ways that stereotypes of gender and race are perpetuated in algorithms. She calls herself the “poet of code” since she combines engineering with language to fight for social change in technology. She believes that, “Poets give voice to the often unseen, unarticulated, or intentionally dismissed.”

Joy is a Rhodes Scholar and Fulbright Fellow, who has been named to many lists including, the Bloomberg 50, BBC 100 Women, Forbes Top 50 Women in Tech, and Forbes 30 under 30. To learn more about Joy’s work in creating a more inclusive technology space, watch her [TED Featured Talk](#) on algorithmic bias. This talk has over 1 million views! You can also learn more about her story in the documentary [Coded Bias](#).

Reflect

Being a computer scientist is more than just being great at coding. Take some time to reflect on how Joy and her work relates to the strengths that great computer scientists focus on building - bravery, resilience, creativity, and purpose.



BRAVERY

Joy has spent her life educating others about how technology can show bias towards certain people. She has used her voice to call for change from many large technology companies. If you had your own platform, what issues would you educate others about?

Share your responses with a family member or friend. Encourage others to read more about Joy and to join in the discussion!

Step 1: What is Poetry? (15 mins)

Did you know that the youngest inaugural poet laureate in the United States is female? Amanda Gorman entered many of our homes this past January, as she recited her poem, “The Hill We Climb” for the inauguration of president, Joe Biden. She was only twenty-two years old when she performed at this national event! A line in her poem that resonated with many was, “Our people diverse and beautiful will emerge, battered and beautiful.” Her powerful words brought hope to many people as her speech went viral. You can read more about Amanda Gorman [here](#).

Poets often speak of a brighter future as they share their art forms with the world. Poetry is a type of literature that is written to share ideas and express emotions. Poets elicit different emotions by carefully selecting words based on their meaning and rhythm. There are many different types of poems, all with unique characteristics. Some of the characteristics we will explore below are rhyme schemes, syllables, and structure.



Image Source: [Amanda Gorman](#)

Get Inspired (10 mins)



There are over 50 different types of poetry! To start this project, let's take some time to explore some of the most popular poetry types. Learn more about each type of poetry by exploring the resources in the table below. While reading, consider the following questions:

- What do you already know about poems?
- What are some other types of poems not included in this list?
- What type of poem do you like the most?
- Remember, poems are written to make you feel a certain emotion. How do each of these poems make you feel?

Step 1: What is Poetry? (cont.)

Type of Poem	Characteristics	Example
<u>Rhymed Poetry</u>	A poem that contains rhymed vowels within the poem.	Sometimes I dream that I can fly. I lift and flap my arms just so, And soon I'm soaring to the sky. Graceful like a bird I go. An Excerpt of a Poem Written By: Patricia A. Fleming
<u>Free Verse</u>	Many refer to this type of poem, as a poem without rules. There is no set meter, no rhyming, and no particular structure.	I was not happy with the rain until my eye caught five wet leaves --- a mysterious autumn watercolor gift here on this broken road. Poem Written By: Amy Ludwig VanDerwater
<u>Haiku</u>	A Japanese poem that has three lines and 17 syllables. The first line has five syllables, the second line has seven syllables, and the third line has five syllables.	An old silent pond A frog jumps into the pond --- Splash! Silence again. Poem Written By: Matsuo Bashō
<u>Sonnet</u>	A poem with 14 lines that has a fixed rhyme scheme. Most sonnets use iambic pentameter, which stresses certain syllables as you read the line of 10 syllables.	Shall I compare thee to a summer's day? Thou art more lovely and more temperate: Rough winds do shake the darling buds of May, And summer's lease hath all too short a date: Sometime too hot the eye of heaven shines, An Excerpt of a Poem Written By: William Shakespeare

Choose a Poem (5 min)



After taking some time to explore some of the different types of poetry, now it is time for you to choose a poem that is meaningful to you. Choosing just one poem can be difficult, but don't worry you can always swap out your poem later or create another poetry generator! As you are narrowing down the poem you will use for the project, we encourage you to start out with a short poem. If you choose a longer poem, start with 1-2 verses of your poem for this project. If you are having difficulty choosing a poem, explore the list of poems below.

- Poets.org
- PoetryFoundation.org
- ButtonPoetry.com
- FamousPoetsAndPoems.com

Write your Poem or Excerpt Below



Check your ideas with the example in the Reference Guide on pg 2.

Step 2: Get Started with Replit (15-20 mins)



For this project we will be using the [Replit](#) web editor. Replit is a free, collaborative, browser-based editor that supports multiple programming languages. This powerful tool can even allow you to code and talk with a group of friends all at the same time!

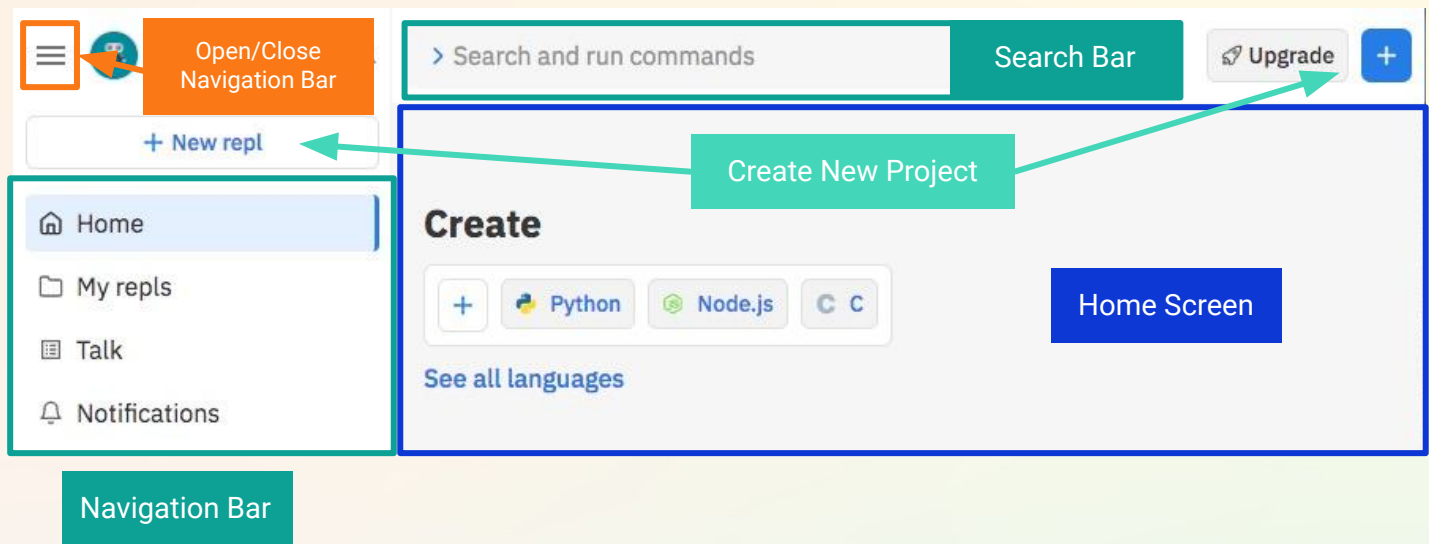
Create an Account (5 mins)

- ☐ **[Sign up](#) or [login](#) to Replit.** In order to save your work you will need to create an account. Follow the instructions on the sign up form to create an account. If you are under 13 you'll need your parent's email address to sign up.
- ☐ **Follow the instructions to create an account.** You may choose to sign up with your Google, GitHub, or Facebook account for faster login.

Explore the Replit Platform (3-5 mins)

Welcome to the Replit platform! Let's explore some of the key features available to get you started.

Step 2: Get Started with Replit (cont.)



- **Navigation Bar:** This column to the left of your screen allows you to access common actions that you might want to engage in, like create or view your Repls.
- **Search Bar:** Instead of using the navigation bar, you can also use the search bar to search and run commands.
- **New Repl:** There are two ways to create a new project. The first is through the navigation bar and clicking on the **+New repl** button or clicking the **+** button on the top right of the window.
- **Main Area:** When you first log in to Replit you will be defaulted to the home screen. Here you will see some quick actions/suggestions for what you might do.

You can learn more about the Replit platform and Repls using this [resource](#).

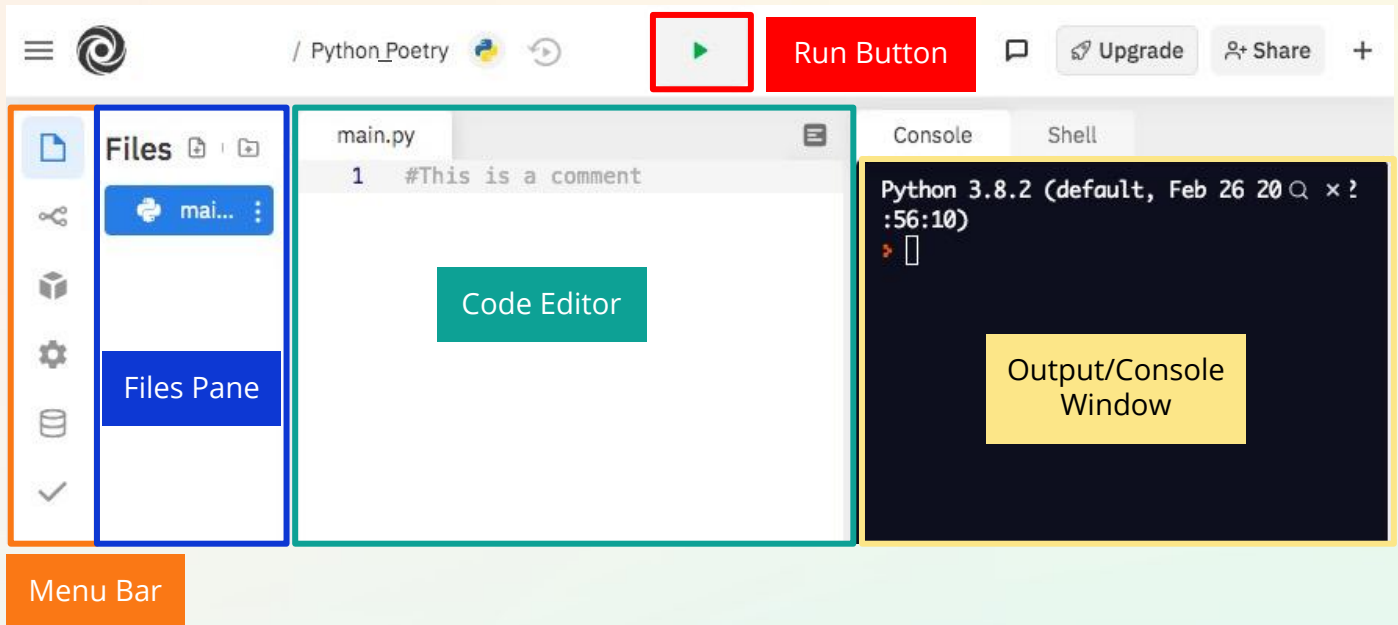
Create a New Replit Project (2 mins)


- ❑ **Create a new Repl.** Click the **+** button on the top right corner of your screen or the **+New repl** button in your navigation bar.
- ❑ **Select Python under the language option.** For this project we will be working in Python, but remember that for any future projects you can program in over 50 languages on Replit.
- ❑ **Name your project.** Give your project a descriptive name like **<yourName>_Python_Poetry**. Typically projects should have no spaces and use [camelCase](#) or underscores to separate words.
- ❑ **Click Create Repl.**

Step 2: Get Started with Replit (cont.)

Explore the Editor View (5-8 mins)

Let's take a look into the editor view for Replit. Now that we created a new project, we need to understand where to code, navigate between files and assets, and how to save and run your code!



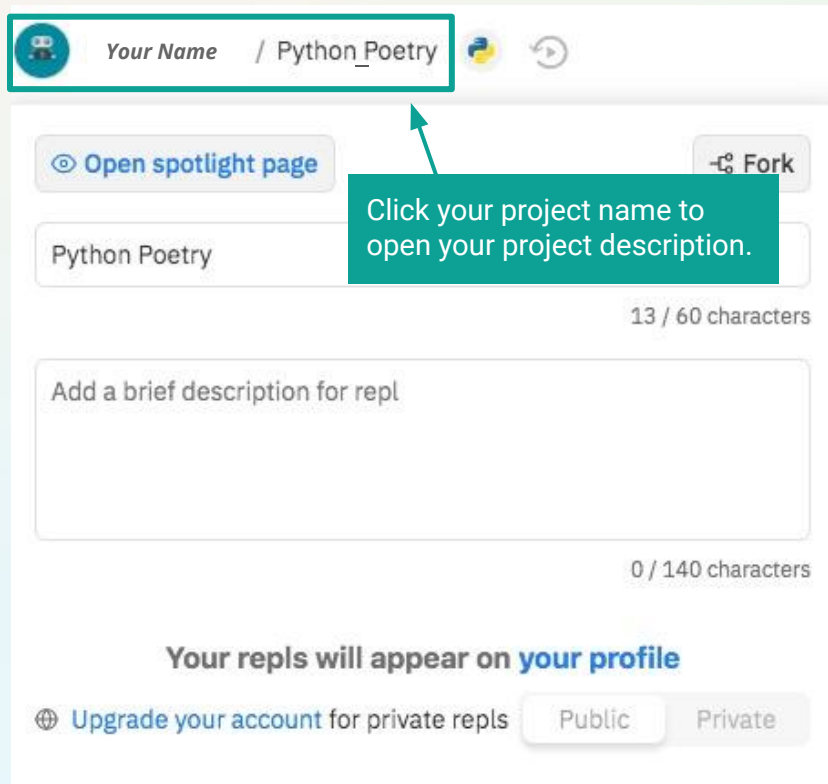
- **Files Pane:** By default, this area will show you all of the files associated with the project. In this activity we will only be working in one file, `main.py`. You may find in other projects you may want to include data files, images, or other Python files. To close this pane, click on the files icon  at the top of the menu bar.
- **Menu Bar:** This bar will allow you to change the view in the file pane. Some options include changing version control, adding packages, debugging, adding a database, and updating settings. In the settings you can customize the layout, theme, font size, and other text settings.
- **Code Editor:** This is where you will write your code!
- **Run Button:** After making changes to your code, click the run button at the top of the editor. You should see your result in the output/console window on the right.
- **Output/Console Window:** This displays any output of your code. All outputs will be visible, so if you click the run button multiple times, each result will be displayed here.

Tip: You may have noticed that there is no **save** button in Replit. As long as you have internet connection, all changes in your code will be saved automatically! Once you click the run button your Repl is saved, so make sure you always run your code before closing your editor!

Step 2: Get Started with Replit (cont.)

Before we start coding our Python Poetry project, let's add a brief project description to our Repl.

- ❑ **Open the project description.** Locate your project name at the top left of your screen. Click the name of your project. This should open a new window containing your project name and an area for a brief description.
- ❑ **Add a brief description.** Something you might want to include in your description may include:
 - ❑ **Overview:** How is it supposed to work?
 - ❑ **Instructions:** Are there any specific instructions needed to run your project?
 - ❑ **Attributions:** Did you get help from others or additional resources? Make sure you shout out these people and resources! This would be a great place to list the poet of the poem you chose.



The screenshot shows the Replit interface for a project named 'Python_Poetry'. At the top, there is a header bar with a user profile icon, the text 'Your Name / Python_Poetry', and icons for Python and a refresh button. Below the header, there is a button labeled 'Open spotlight page' and a 'Fork' button. The main content area has a text input field with the placeholder text 'Add a brief description for repl'. Above this field, there is a green callout box with the text 'Click your project name to open your project description.' and an arrow pointing to the project name in the header. Below the input field, there is a character count '0 / 140 characters'. At the bottom, there is a section titled 'Your repls will appear on your profile' with a link to 'Upgrade your account for private repls' and two buttons: 'Public' and 'Private'.

This section does not need to be final now, you can always go back and edit your description later.

Step 3: Introduction to Python (5 mins)



Python is a text-based programming language, which means that all commands will need to be typed! Many programmers choose to use Python because it is easy to learn and understand since it mimics English-like commands. Python is an **open source** language, meaning that it is freely available for the public to use and modify as necessary. There are strict guidelines on how updates are accepted and implemented to the language, but essentially anyone can contribute to the evolution of the language!

Features of Text-Based Languages

Programmers often use certain features of a programming language to make it easier for users to understand what they were trying to accomplish in their project. Things such as referring to the numbered lines or using comments allow code to be more readable.

1. Numbered Lines: As we type code into Python, each line of code is numbered in the code editor.

```
main.py
1  import time
2
3  print("Welcome to the Poem Generator.")
4  print("I need a few words from you to complete the poem.")
```

When we refer to a certain part of our code, the numbered lines allow us to be specific on where to look. If our program has an error in the code, Python will share which line of code needs to be debugged. In the example below, we can see that there is an error in the code on line four.

```
File "main.py", line 4
    print("I need a few words from you to complete the poem.")
                                     ^
SyntaxError: EOL while scanning string literal
```

Step 3: Introduction to Python (cont.)

2. Code Comments: These are lines of code that are not read by the computer. They are used to describe what a certain section of code will do. To add a code comment in Python, you start each line with the `#` symbol. This lets the computer know that the line of code is a comment, so the computer does not need to perform any of the code directly after the `#` symbol. Let's take a look at an example:

Sample Code:

```
1. # This is a comment
2. print("Hello World.")
3. # Here we can write anything without the computer reading our messages
4. message = "hi"
```

Take a moment to review the code snippet above. Which lines of code are comments? How do you know? Remember, you can refer to the specific lines of code by their line number.



Check your ideas with the example in the Reference Guide on pg 3.

Step 4: Print Statements in Python (15 mins)



To get started in Python, we will learn how to use the `print()` function. In order to have a formatted message appear in the output portion of your screen, we will wrap a string of characters in a `print()` command. When using this function, the string needs to be wrapped in double quotes `" "` or single quotes `' '`.

Printing your Poem (5 mins)

Let's write our first `print()` statement. Start by using the `print()` command to `print()` the string, hello world in our replit screen. Below are a few different ways you could `print()` this statement. Notice, you can use either single or double quotes when printing a message and the same message will appear.

Sample Code:	Output
1. <code>print('Hello World.')</code>	Hello World
2. <code>print("Hello World.")</code>	Hello World

Step 4: Print Statements in Python (cont.)

Creating a New Line (5 mins)

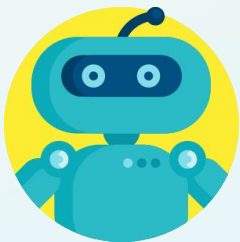
As we explored different types of poems in step #1 of this project, we learned that the format and structure of poems are very important to the art. Poems are often created based on a certain amount of syllables per line, therefore it is important that the poem we print in Python honors the way the poem was written. Let's learn how to write out our poem with multiple lines, using one of the following two options:

1. **Use multiple `print()` statements:** Rather than typing your entire poem in one `print()` statement, use multiple `print()` statements to indicate each individual line of the poem.

Sample Code:	Output:
<pre>1. print("Hold fast to dreams") 2. print("For if dreams die") 3. print("Life is a broken-winged bird") 4. print("That cannot fly.")</pre>	<pre>Hold fast to dreams For if dreams die Life is a broken-winged bird That cannot fly.</pre>

2. **Use the `\n` command:** Another way to create multiple lines in Python is by using the newline command. By placing `\n` in a string of characters, this will tell Python to create a new line for the characters that directly follow the `\n` command.

Sample Code:	Output:
<pre>1. print("Hold fast to dreams\nFor if dreams die\nLife is a broken-winged bird\nthat cannot fly.")</pre>	<pre>Hold fast to dreams For if dreams die Life is a broken-winged bird That cannot fly.</pre>



Print your Poem (5 mins)

- ☐ Print out your poem using one of the two options above. Remember, your poem should honor the way that the poem was written.
- ☐ Test what you have written so far to make sure your program runs the way you want it to. Click the **Run** button to run your sketch. You should have:
 - Your poem displayed in the output portion of Replit, in the same format of the original poem.

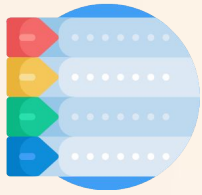
Not working the way you want? Try these debugging tips:

- ☐ Did you type a lowercase `print()`?
- ☐ Do you type single or double quotations around the lines of your poem?
- ☐ If you used the newline method did you use a backslash `\` rather than a normal slash `/`?



See a full example in the Reference Guide on pg 3 and 4.

Step 5: Create a Poem Generator (20 mins)



Now that we have formatted our poem correctly, it is time to use variables to make our poem more interactive. We will begin by selecting three different words in our poem to replace with variables, so that the user can customize the poem. This will create a fun new poem each time the program is run!

Determine User Input (5 mins)

Choose three words in your poem that will be replaced with variables. Usually, it is best to pick words in your poem that are nouns or verbs. If you are unsure about what different parts of speech are, you can read more about them [here](#). Fill out the table below to determine which words will be the variables in your poem.

Word #1	What prompt will your user see?
Example Bird	Type in an animal.
Word #2	What prompt will your user see?
Word #3	What prompt will your user see?



See a full example in the Reference Guide on pg 4.

Input in Python (5 mins)

Now that you have chosen the three variables for your poem, we need to adjust our program so that we can get input from the user. The `input()` function in Python works very similarly to the `print()` function.

- ❑ In line number one of your program, let's try using the `input()` command. Type in your prompt for **Word #1**, in an `input()` function. Complete this same step for **Word #2** and **Word #3**.

Sample Code:	Output:
1. <code>input("Type in an animal.")</code>	Type in an animal.

Step 5: Create a Poem Generator (cont.)

When you run your program, what is different about the output section of Replit?

What is the difference between a `print()` command and an `input()` command?



See a full example in the Reference Guide on pg 5.

Variables in Python (5 mins)

Using the `input()` command the user can now type in information that we can use in our program. We just need somewhere to store this information!

A **variable** is a container that is used to store information in a program. Variables have unique names that represent a value. When choosing a variable name in Python make sure that the name does not have any spaces. If you would like to use multiple words to name a variable, programmers often combine multiple words with an underscore `_` character or use camelCase. For example, `type_of_animal`. Use the table below to create a variable name for each of the words the user will replace.

Example

Word #1	Variable Name
Bird	animal
Word #2	Variable Name
Word #3	Variable Name



See a full example in the Reference Guide on pg 5.

Assign Variable Names

Next, we need to assign the variable names to the input that the user typed in. We can do this by placing an equal sign `=` between the variable name and the `input()` statement. In the example below, when the user types in input to the statement, their answer is stored in the variable `animal`.

- ❑ **Adjust the lines 1-3 in your program, by assigning each `input()` function with the matching variable name you chose earlier.**

Sample Code:	Output:
1. <code>animal = input("Type in an animal.")</code>	Type in an animal.

Tip: Now that your program is getting longer, consider adding comments before each section of code to make your program easier to read.

- ❑ **Click the Run button to run your sketch.** Test what you have written so far to make sure your program runs the way you want it to. You should have:
 - ➔ The program allows the user to type in responses to three input statements. The program assigns the values to variables. Your poem is displayed in the output portion of Replit, in the same format of the original poem.

Not working the way you want? Try these debugging tips:

- ❑ Did you spell `input()` method correctly?
- ❑ Remember your variable name can only be one word. If you have multiple words, did you combine them with an underscore?
- ❑ Did you spell the `print()` method correctly?
- ❑ Do you type single or double quotations around the lines of your poem?



See a full example in the Reference Guide on pg 5.

Step 6: Displaying the New Poem (10 mins)



Now that we have created three new variables, we need to adjust our poem to display these new words.

Concatenation (5 mins)

Concatenation is a programming term that means combining a string with another data type without any gaps. For our Python poetry generator we need to combine the lines of our original poem with the three variables we created. In order to combine our string with these three variables, we will add **+** signs before and/or after our variable.

Sample Code:	Output:
<pre>1. animal = input("Type in an animal.") 2. print("Hold fast to dreams") 3. print("For if dreams die") 4. print("Life is a broken-winged " +animal)</pre>	<pre>Type in an animal. Hold fast to dreams For if dreams die Life is a broken-winged _____</pre>

In line number three, the original line of the poem is concatenated with the variable **animal**. In the output, after broken-winged, whatever the user typed in for the variable **animal** will be displayed. Notice, after the word winged there is a space before the quotation mark, this tells the computer to put a space before the variable **animal** is displayed.

To concatenate a string with a variable in the middle of a line, use two **+** signs, one on each side of the variable.

Sample Code:	Output:
<pre>1. bird_part = input("Type in a body part of a bird.") 2. print("Hold fast to dreams") 3. print("For if dreams die") 4. print("Life is a broken-" +bird_part+ " bird")</pre>	<pre>Type in a body part of a bird. Hold fast to dreams For if dreams die Life is a broken-winged _____</pre>

Step 6: Displaying the New Poem (cont.)

Print your Customized Poem (3-5 mins)

- **Adjust your print statement(s) to include your three variables.** Each time the poem is run, it will customize the poem based on what the user types in.
- **Click the Run button to run your sketch.** Test what you have written so far to make sure your program runs the way you want it to. You should have:
- The program allows the user to type in responses to three input statements. The program assigns the values to variables. Your poem is displayed in the output portion of Replit, in the same format of the original poem, with the three words the user typed in replacing the original words of the poem.

Not working the way you want? Try these debugging tips:

- ☐ Did you spell the `input()` method correctly?
- ☐ Remember your variable name can only be one word. If you have multiple words, did you combine them with an underscore?
- ☐ Did you spell the `print()` method correctly?
- ☐ Do you type single or double quotations around the lines of your poem?
- ☐ Did you put spaces before the quotation marks to create spaces before a variable?



See a full example in the Reference Guide on pg 6.

Step 7: Extensions (15-20 mins)



There are so many ways you can take your Python Poetry Generator to the next level! Try some of the extensions below.

Extension 1: Add More Variables into your Poem (5 mins)

Try adding more variables into your poem. You could even choose a poem that is longer in length. This extension will allow you to practice concatenation as you create many more `print()` statements.



See a full example in the Reference Guide on pg 6.

Extension 2: Format your Poem with the Tab Command (5 mins)

Since poetry is very specific about format and structure, rather than just creating new lines in a poem, some poems actually indent certain lines. By using indentation the poet is able to provide more emphasis on specific lines of the poem. Let's read the poem below by Charles Wright to see how a poet might use this structure.

Poem that Includes Indentation

I want to sit by the bank of the river,
 in the shade of the evergreen tree,
And look in the face of whatever,
 the whatever that's waiting for me.

To indent lines in Python, we will use the tab command. This is very similar to the newline command we used above. Instead of typing `\n`, type `\t` in front of a word in your string and it will indent the line that directly follows it. Type `\t` multiple times to create a bigger indentation.

Sample Code:

```
1.print("I want to sit by the bank of the river,")
2. print("\tin the shade of the evergreen tree,")
3. print("And look in the face of whatever,")
4. print("\t\t\t\t\tthe whatever that's waiting for me.")
```



See a full example in the Reference Guide on pg 6.

Extension 3: Add Wait Time to your Poem (5 mins)

A **library** is a collection of functions and variables. Programmers use libraries since many functions are already written for you to use! This built-in Python library, called **time** will allow us to use different time features to make our poem more dynamic. Before we can use any of the functions in a library, we first need to tell the computer to **import**, or load the library.

PYTHON	DESCRIPTION
<code>import time</code>	<ul style="list-style-type: none"> → import: This keyword lets the computer know to load the information from a Python library. → time: This Python library allows access to different time-related functions.

The specific function we will use is called **time.sleep()**. The number we put in the parenthesis will tell our program how many seconds to wait until performing the next line of code. If you place a `time.sleep()` command after each print statement in your poem, the poem will appear, line by line, in the output section of Replit.

Sample Code:

```
1. print("Hold fast to dreams")
2. time.sleep(1)
3. print("For if dreams die")
4. time.sleep(1)
5. print("Life is a broken-winged " + animal)
```



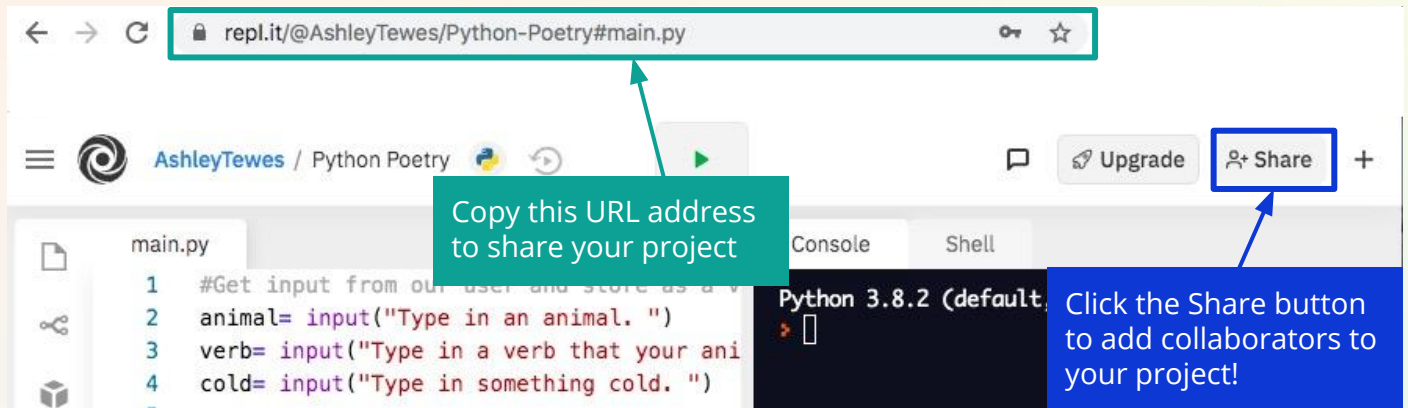
See a full example in the Reference Guide on pg 6.

Step 8: Share Your Project (5-10 mins)



View Only Access (2 min)

Sharing your work on Replit is easy! Just copy and paste the URL address of your Repl project in the web address bar at the top! This will allow others to run your project, view your code, and “fork” (duplicate) your project and remix on their own! **Be sure to share this link on your social media accounts and don't forget to tag @girlswhocode #codefromhome and we might even feature you on our account!**

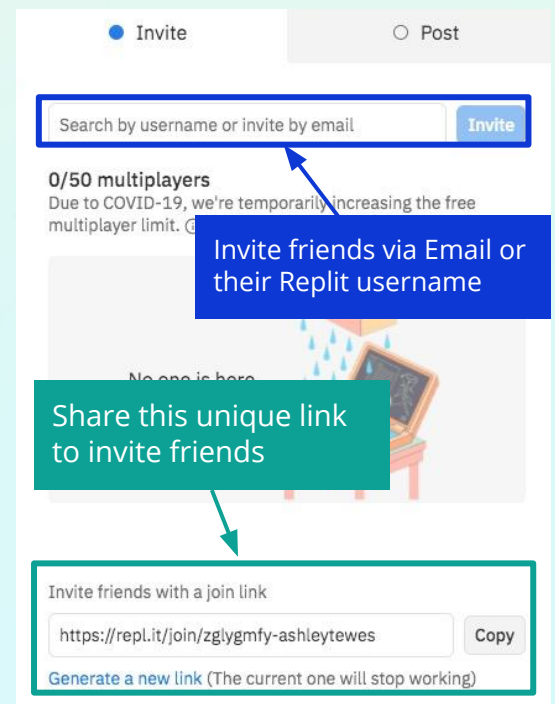


Adding Collaborators (2 mins)

If you want to work with a group of friends on a project, you can easily invite them to collaborate using the **Share** button at the top-right of the window. This should pop out a new window with two options for inviting others to collaborate on your project.

- **Invite by email or Replit username.** This option allows you to share your project with specific people by typing in their email address or Replit username if they already have an account with Replit. We recommend this option to ensure that you are sharing your project with the correct people!
- **Share invite link.** At the bottom of the window there is a unique invite link. You can copy and paste this link to friends which will allow them to access your project.

Note about collaborators: Remember that adding collaborators gives others edit access to your project. This will allow them to change your code, name, and description. **DO NOT share your invite link on social media!** Be selective on who you share these edit rights with.





Girls Who Code At Home

Python Poetry
Reference Guide

Python Poetry - Reference Guide



In this document you will find sample answers to some of the questions in the activity. Follow along with the activity and when you see this icon, stop and crosscheck your ideas here.

Step 1: What is Poetry?

Choose a Poem

We encourage you to choose a short poem that is meaningful to you! Below is an example of a poem by Langston Hughes called, *Dreams*.

Write your Poem Below

Hold fast to dreams
For if dreams die
Life is a broken-winged bird
That cannot fly.

Hold fast to dreams
For when dreams go
Life is a barren field
Frozen with snow.

If you are having difficulty choosing a poem, don't forget to explore some of the poems below!

- Poets.org
- PoetryFoundation.org
- ButtonPoetry.com
- FamousPoetsAndPoems.com

Step 3: Introduction to Python

Use the sample code below to reflect on how to use comments in Python.

Sample Code:

```
1. # This is a comment
2. print("Hello World.")
3. # Here we can write anything without the computer reading our messages
4. message = "hi"
```

Reflect

Take a moment to review the code snippet above. Which lines of code are comments? How do you know? Remember, you can refer to the specific lines of code by their line number.

Line #1 and #3 are comments in Python. We know that these two lines are comments because they start with the a # symbol.

Step 4: Print Statements in Python

You Try: Print your Poem

Once you choose your poem, it is time to print your poem in Python. We learned that the format and structure of poems are very important to the art. Below we shared the code for creating new lines in Python using both methods! We will continue to use our poem from above, *Dreams*.

→ Use multiple `print()` statements:

Sample Code:

```
1. print("Hold fast to dreams")
2. print("For if dreams die")
3. print("Life is a broken-winged bird")
4. print("That cannot fly.")
5. print("Hold fast to dreams")
6. print("For when dreams go")
7. print("Life is a barren field")
8. print("Frozen with snow.")
```

Step 4: Print Statements in Python (cont.)

→ Use the `\n` command:

Sample Code:

```
1. print("Hold fast to dreams\nFor if dreams die\nLife is a broken-winged bird\nthat cannot fly. Hold fast to dreams\nFor when dreams go\nLife is a barren field\nFrozen with snow.")
```

Step 5: Create a Poem Generator

Determine User Input

Use this table to help determine which words will be the variables in your poem. Here is an example of what your table would look like if you used the poem, *Dreams*.

Word #1	What prompt will your user see?
Bird	Type in an animal.
Word #2	What prompt will your user see?
Fly	Type in a verb that your animal would do.
Word #3	What prompt will your user see?
Snow	Type in something cold.

Reflect

When you run your program, what is different about the output section of Replit?

There is a flashing bar in the output window that allows you to type something in.

What is the difference between a `print()` command and an `input()` command?

A `print()` command lists a string in the output window. An `input()` command allows the user to type in feedback to the string.

Variables in Python

Use this table to create variable names for each of the words the user will replace.

Word #1	Variable Name
Bird	animal
Word #2	Variable Name
Fly	verb
Word #3	Variable Name
Ice	cold

You Try: Assign Variable Names

At this point in the program you should have three variables assigned to `input()` commands. Below we shared the code for assigning variables in Python.

Sample Code:

```
1. animal= input("Type in an animal.")
2. verb= input("Type in a verb that your animal would do.")
3. cold= input("Type in something cold.")
```

Step 6: Displaying the New Poem

You Try: Print your Customized Poem

Below we shared the code for how to complete your program. You can check out our [full sample project](#) for more ideas!

Sample Code:

```
1. animal= input("Type in an animal.")
2. verb= input("Type in a verb that your animal would do.")
3. cold= input("Type in something cold.")
4. print("Hold fast to dreams")
5. print("For if dreams die")
6. print("Life is a broken-winged " +bird)
7. print("That cannot " +verb+ ".")
8. print("Hold fast to dreams")
9. print("For when dreams go")
10. print("Life is a barren field")
11. print("Frozen with " +cold+ ".")
```

Step 7: Extensions

Extension 1: Add More Variables into your Poem (5 mins)

Check out our [full extension project](#) for more ideas!

Extension 2: Format your Poem with the Tab Command (5 mins)

Check out our [full extension project](#) for more ideas!

Extension 3: Add Wait Time to your Poem (5 mins)

Check out our [full extension project](#) for more ideas!