INIAD CS Essentials

2-2: From REPL to Programs

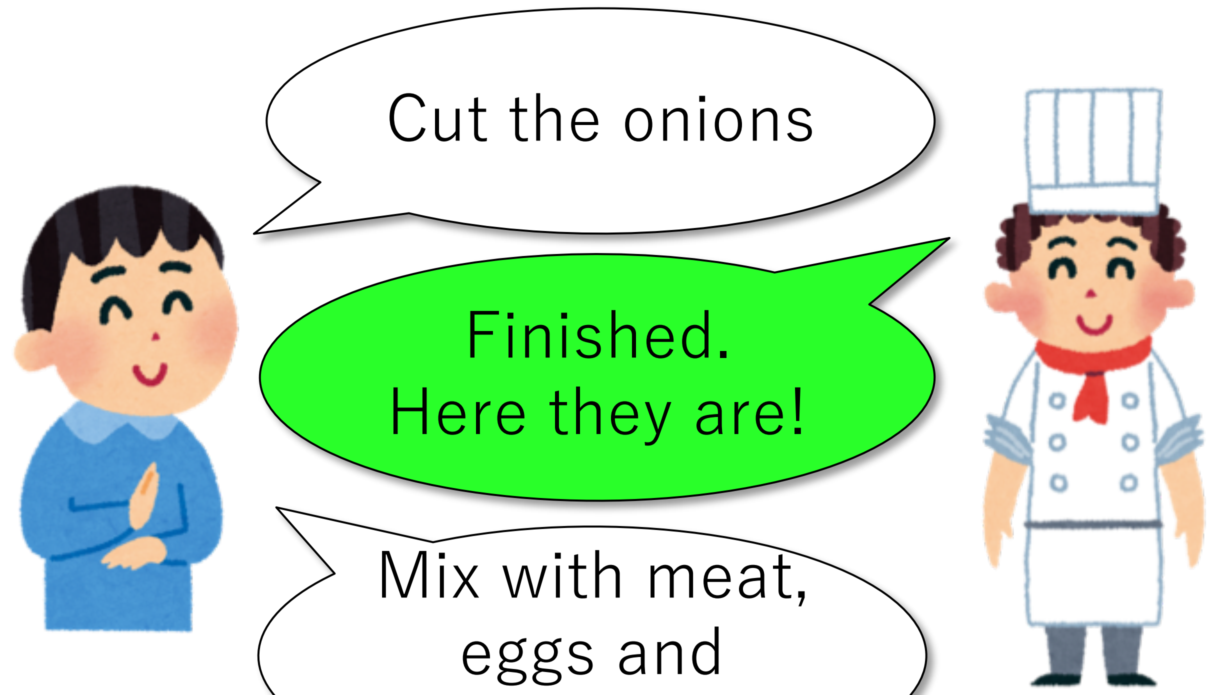
We have given separate commands to Python, line by line.
From here, we'll learn how to give Python your "recipes", made up of multiple commands

1. Let’s write programs

Let’s make “job order” made up of multiple commands

Getting out of REPL (read-eval-print loop)

- REPL = What you've used in the last lectures
  - A mechanism that accepts separate commands one by one, executes one by one, displays the results one by one
- Let's study how to pass bunch of instructions to Python as a "program"



REPL environment

-Hamburger Recipe
(1) Chopped onion
(2) Meat, eggs, bread crumbs and mix...

What you learn today is to write "recipes (programs)" made up of multiple commands

Let’s combine what you wrote

- By combining multiple commands that you have written in REPL, you can make a “program”
  - For example in lecture 1-2, we have written a sequence of commands to get an area dimension of a circle. You can simple put them in sequence to form a program.

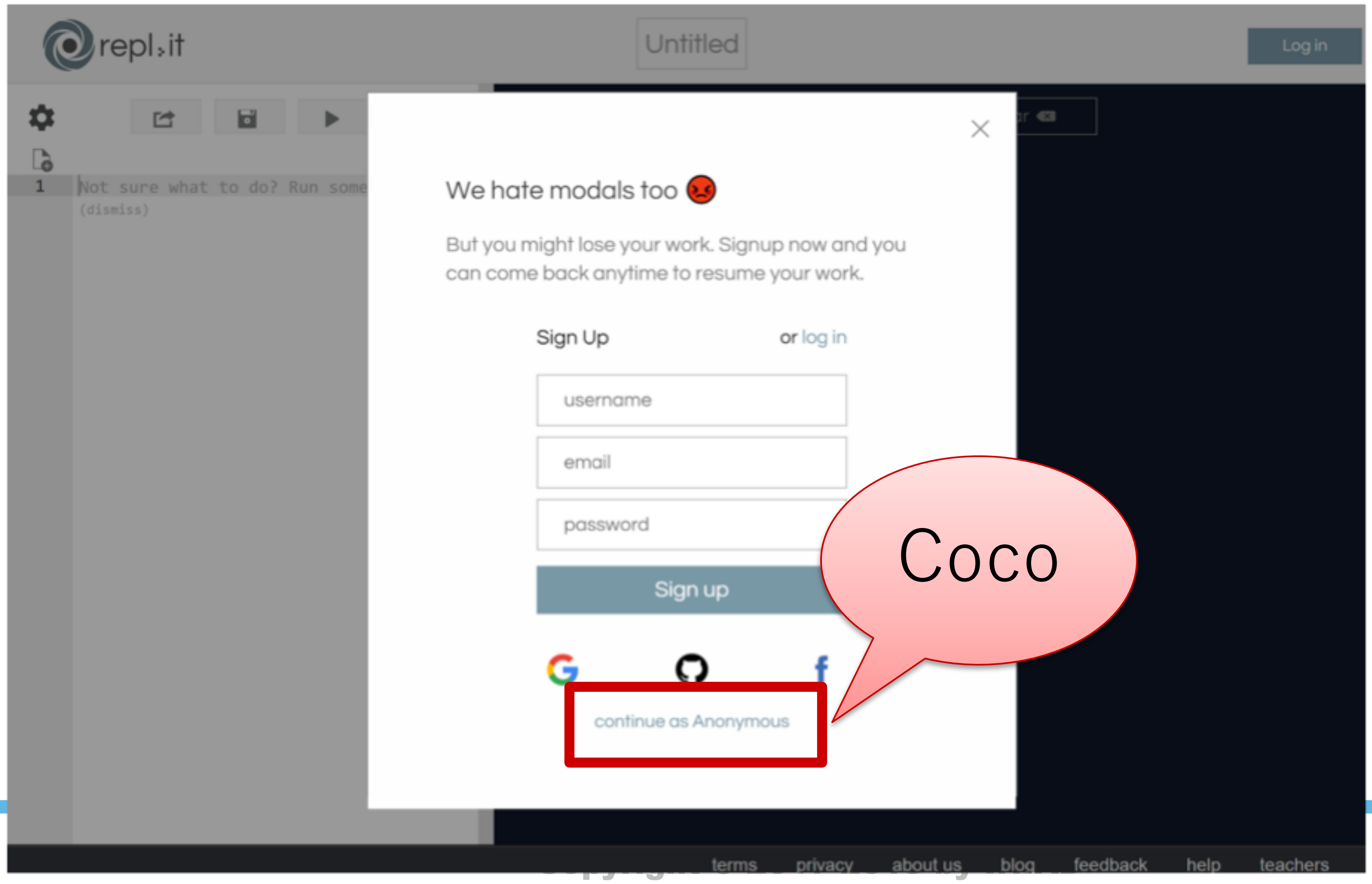
- The program runs from top to bottom, line by line

```
pi = 3.14159
r = 5
pi * r * r
```

Executed from top to bottom, line by line

Let's input & run a program

- Like before, open https://repl.it/languages/python3
  - Click the above link to open Python in a browser
  - After the page is displayed, click “continue as Anonymous”



repl.it Untitled Log in

Not sure what to do? Run some
(dismiss)

We hate modals too 😡

But you might lose your work. Signup now and you can come back anytime to resume your work.

Sign Up or log in

username

email

password

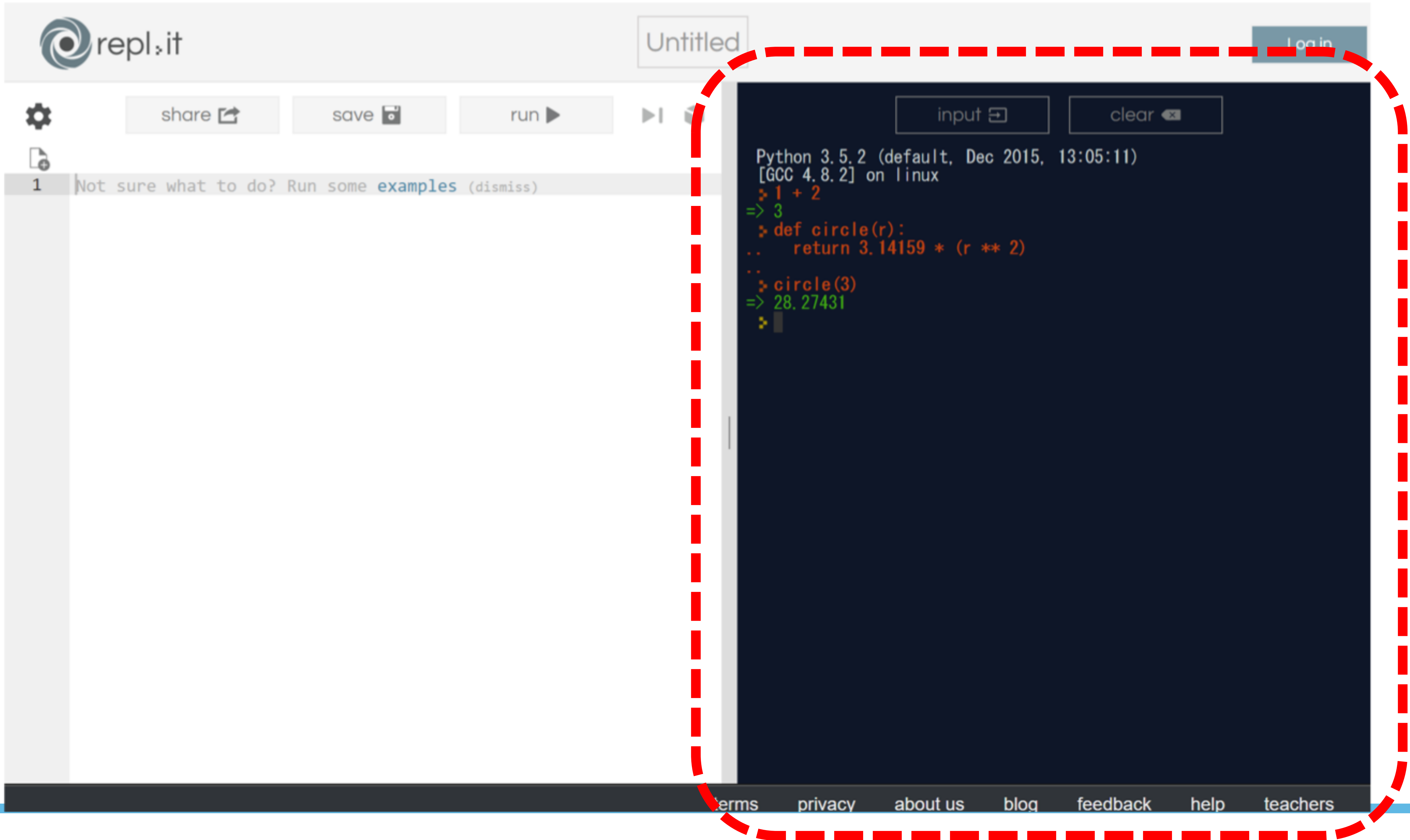
Sign up

continue as Anonymous

terms privacy about us blog feedback help teachers

Let's input & run a program

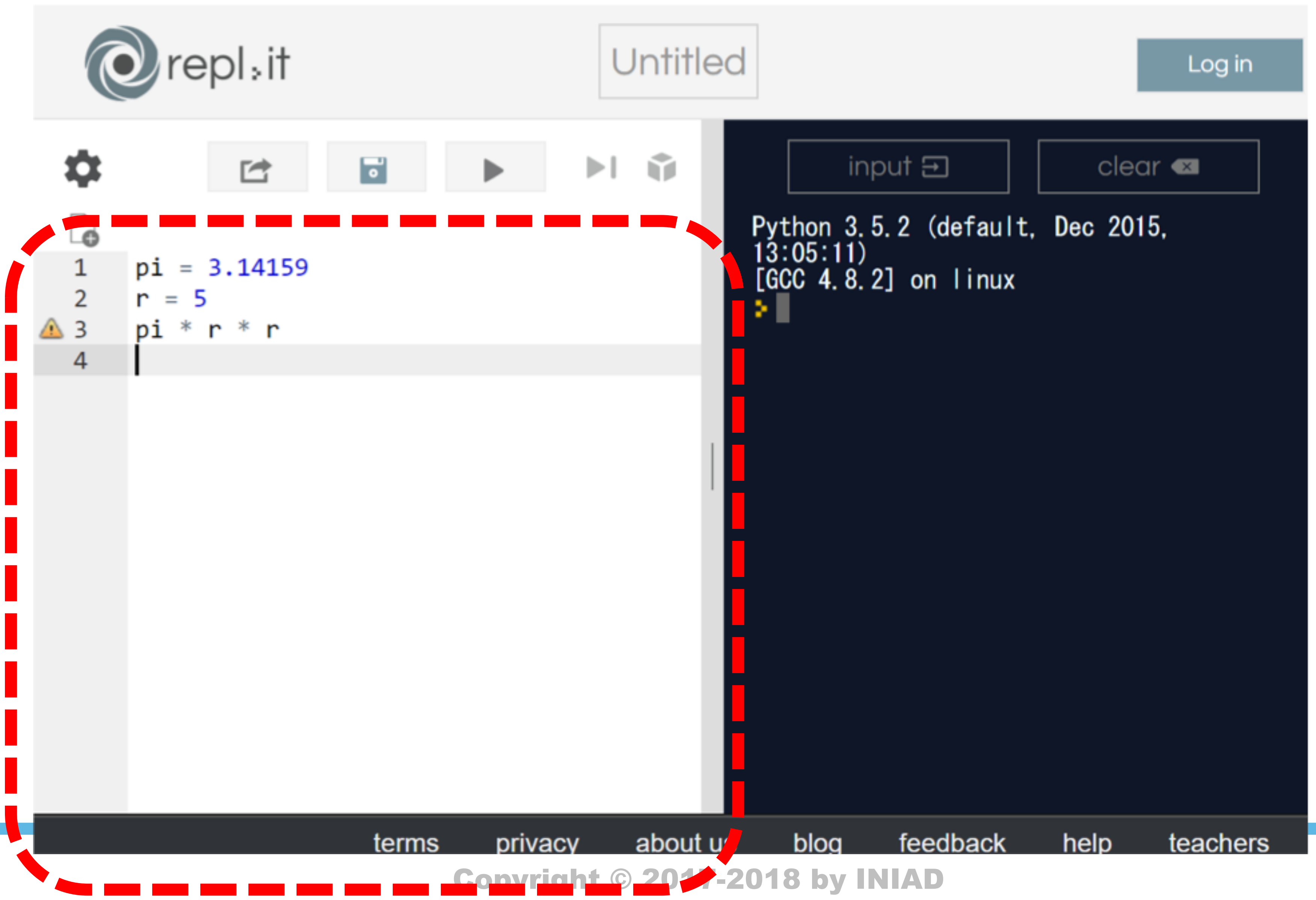
- The right side of the page can be used as ordinary REPL



```
Python 3.5.2 (default, Dec 2015, 13:05:11)
[GCC 4.8.2] on linux
 1 + 2
=> 3
 def circle(r):
..   return 3.14159 * (r ** 2)
..
 circle(3)
=> 28.27431
```

Enter the program and try to move it

- Let’s input calculation of pi to the **the left side**
  - Note: if an unintended string is entered by auto-completion function of repl.it, press ESC to clear

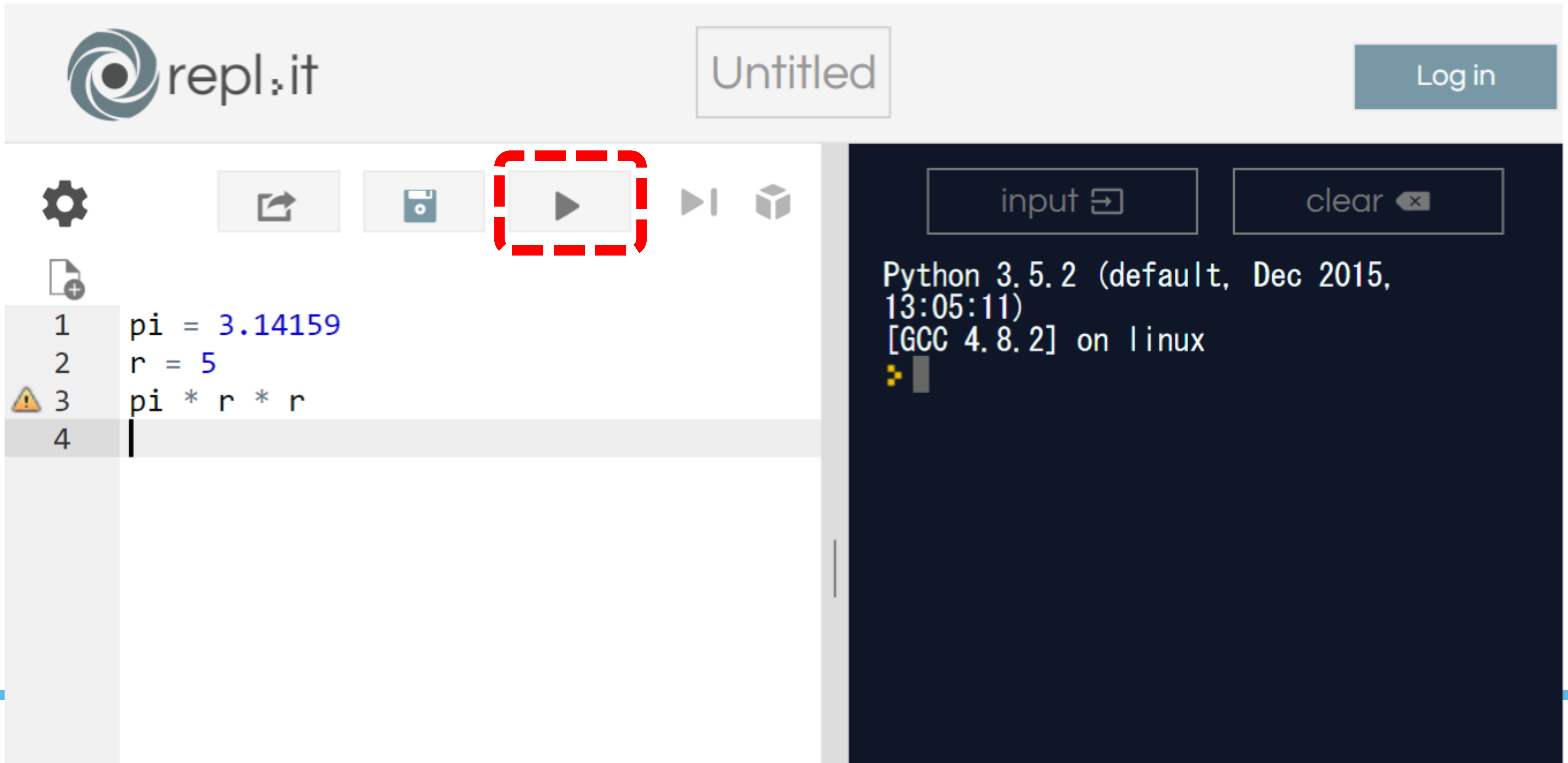


```
pi = 3.14159
r = 5
pi * r * r
```

```
Python 3.5.2 (default, Dec 2015, 13:05:11)
[GCC 4.8.2] on linux
```

Enter the program and try to move it

- If you finished your input, press [▶](Run) button
  - This lets Python run your program

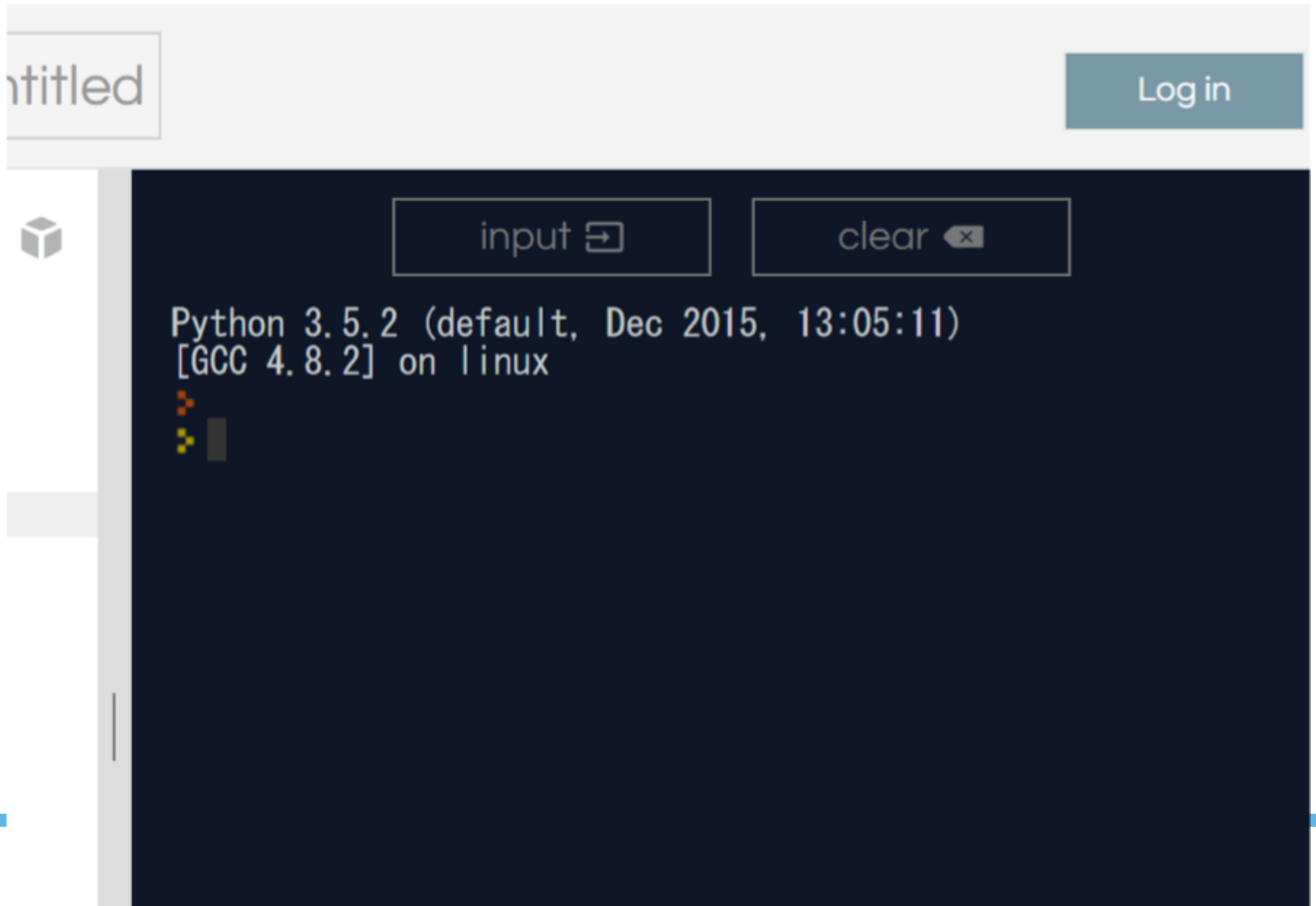


```
pi = 3.14159
r = 5
pi * r * r
```

```
Python 3.5.2 (default, Dec 2015,
13:05:11)
[GCC 4.8.2] on linux
```

And the result?

- Oops! Nothing?
- You would have seen something in the right side of the page, if it were REPL



```
Python 3.5.2 (default, Dec 2015, 13:05:11)
[GCC 4.8.2] on linux
>
>
```

Program runs silently

- Unlike REPL, programs runs silently; not displaying results of every line of commands
  - In REPL, Python automatically reported results, like:

    ```
    > 6 - 7
    -1
    > 3.1 - 2.0
    1.1
    ```

  - When you run a program, Python will not automatically wait your input by `>`, nor displays the results like `-1`, `1.1`

Then, what shall we do?

- Using built-in function print(), you can command Python to display data on console from programs
- For example:
  - `print(6 – 7)`
  - `print('Of course, strings are OK as well')`
- By giving multiple arguments, they are to be displayed in one line
  - `print('The value of x is', x)`

Exercise 1: print the area of the circle

- Add print() to the previous program so that you can see the results properly
- When finished, press the Run button and check if the result is now displayed onto the black part on the right (called “**console**”)

Example answer

- Was your answer similar to this?
  (The added part is marked with red color)

  -
    ```
    pi = 3.14159
    r = 5
    print('The size of a circle with radius',
          r, 'is', pi * r * r)
    ```

Receive input from users

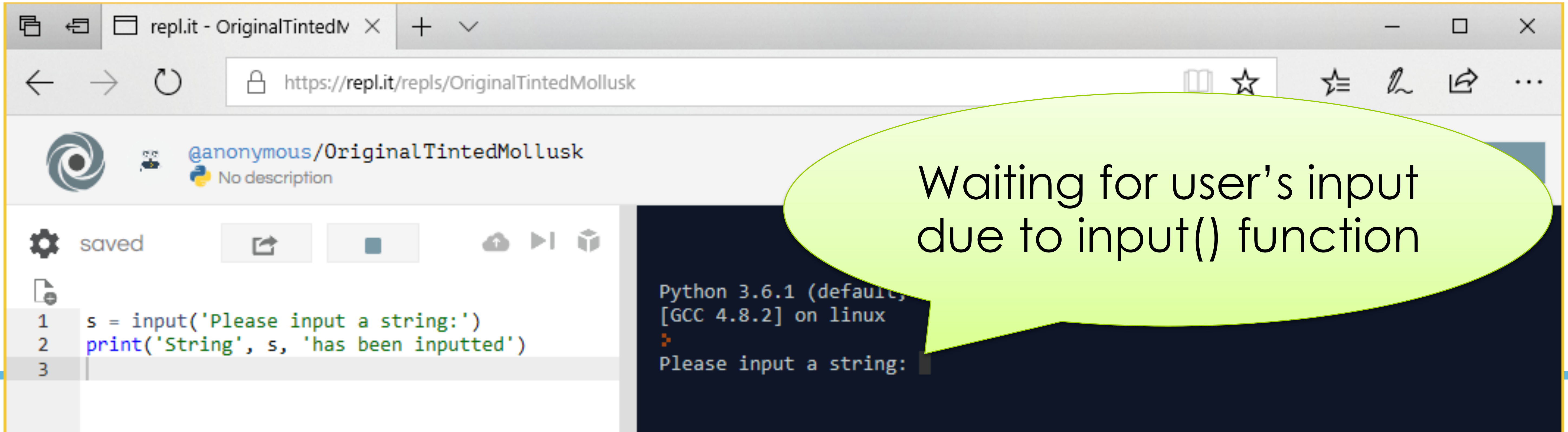
- By using function input(), you can accept a string input from console
  - As an argument, a *prompt* string, which is displayed before the input, is specified
  - Example:

    Example:

```
s = input('Please input a string:')
print('String', s, 'has been inputted')
```

Running the example

- Program runs from top to bottom, line by line
- When the program is executed from the start, input() function is called; the prompt is shown and then starts waiting for user's input of string from keyboard
  - Function input() does not return result until user finishes input; thus, the program is now stopped at the first line


repl.it - OriginalTintedM
https://repl.it/repls/OriginalTintedMollusk
@anonymous/OriginalTintedMollusk
No description
saved
Waiting for user's input due to input() function

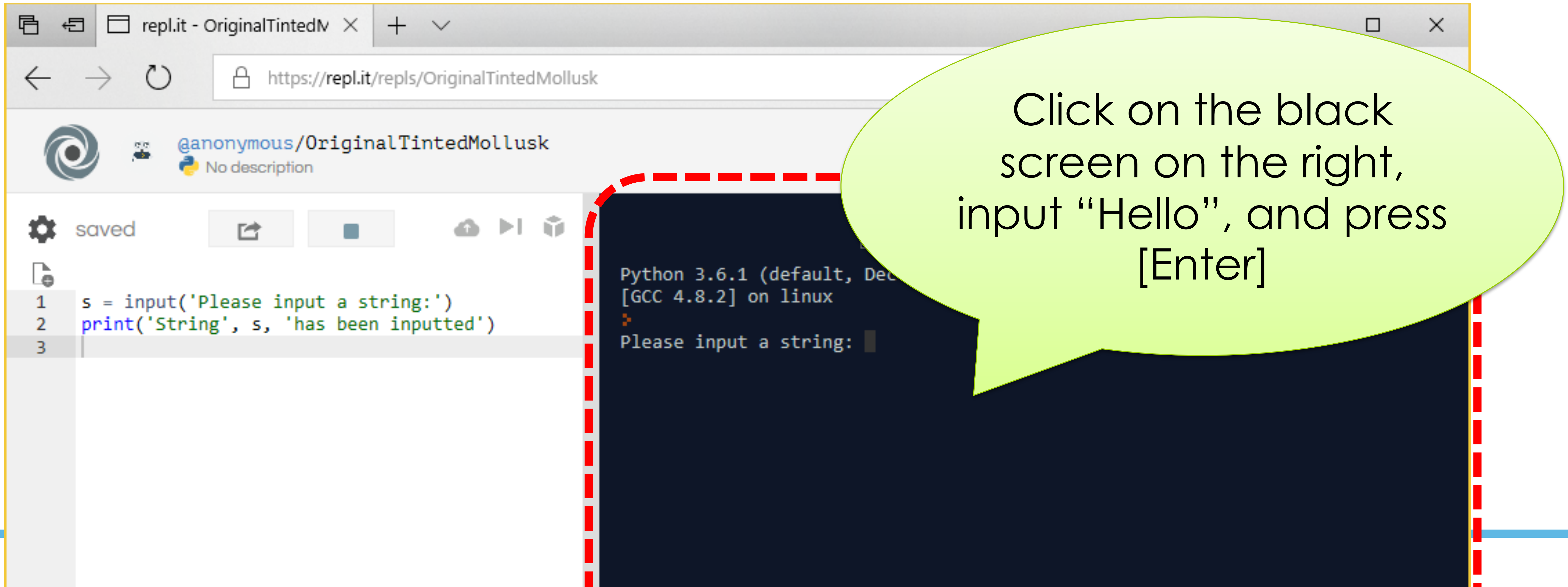
```
s = input('Please input a string:')
print('String', s, 'has been inputted')
```

```
Python 3.6.1 (default,
[GCC 4.8.2] on linux

Please input a string:
```

Running the example

- Let’s input something to let the program runs further
- Click the black part on the right, input “Hello”, and finally press [Enter]

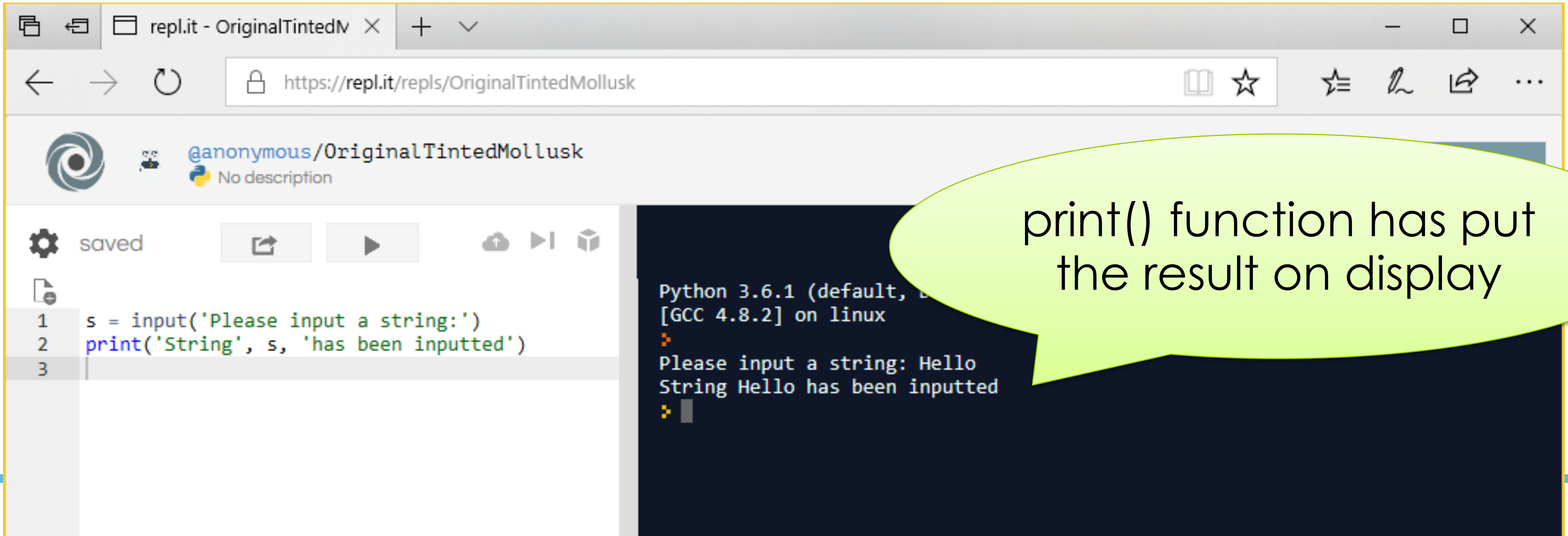

repl.it - OriginalTintedM
https://repl.it/repls/OriginalTintedMollusk
@anonymous/OriginalTintedMollusk
No description
saved
Click on the black screen on the right, input “Hello”, and press [Enter]

```
s = input('Please input a string:')
print('String', s, 'has been inputted')
```

```
Python 3.6.1 (default, Dec
[GCC 4.8.2] on linux
›
Please input a string:
```

Running the example

- Function input() is a function that returns the result of user's input from keyboard
- When the input is finished, the execution of function input (...) ends; then string "Hello" is returned, and the program runs further
- By calling function print() in the second line, the result is displayed and the program is finished by reaching the end


repl.it - OriginalTintedM
https://repl.it/repls/OriginalTintedMollusk
@anonymous/OriginalTintedMollusk
No description
saved
print() function has put the result on display

```python
s = input('Please input a string:')
print('String', s, 'has been inputted')
```

```
Python 3.6.1 (default,
[GCC 4.8.2] on linux
›
Please input a string: Hello
String Hello has been inputted
›
```

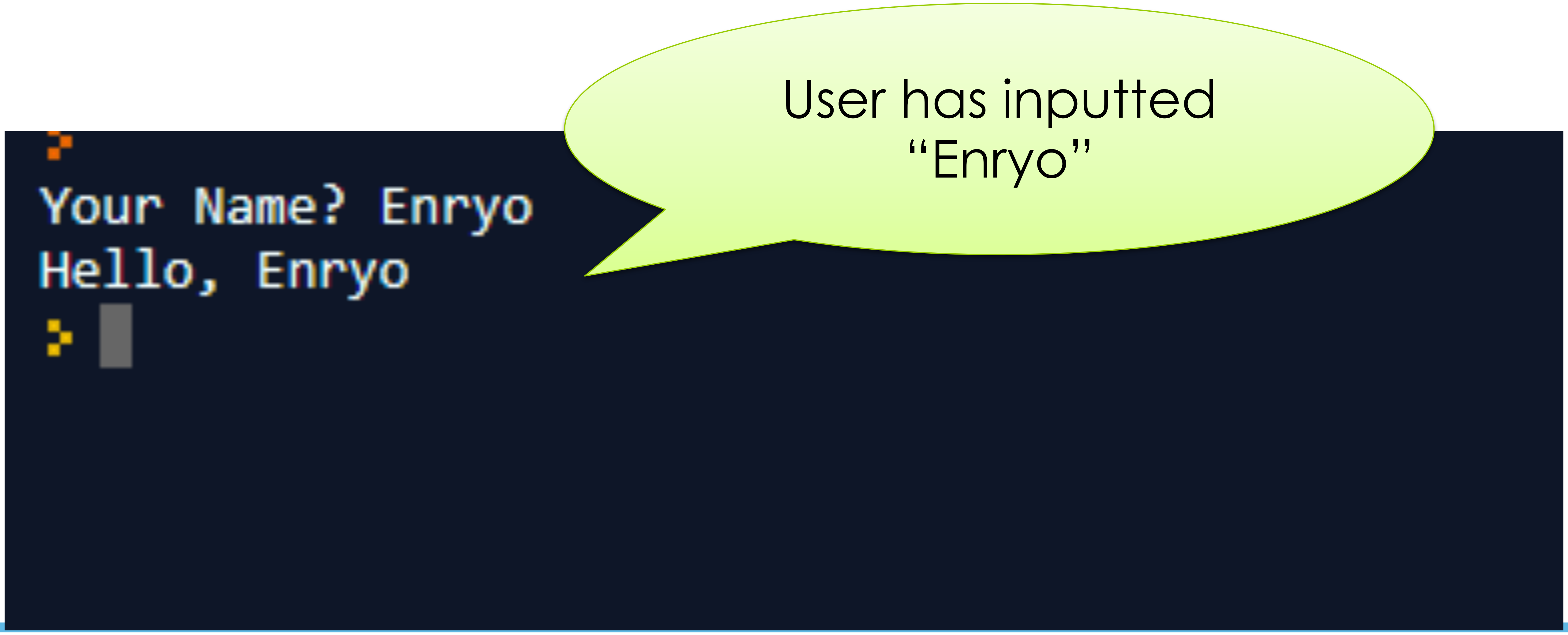
print and Input

- Python provides functions equipped with special functionalities other than calculation, unlike mathematical functions you learned in high schools
  - print(): show the value given as an argument
  - input(): shows the string (prompt) and returns the string user has inputted

- There are also functions like drawing fancy shapes on display
  - Please look forward to learning them in exercise class

Exercise 2: A program that greets you

- Make up a program that asks you with a prompt "Your Name?" and shows a greeting message “Hello, …” with the name inputted



```
>
Your Name? Enryo
Hello, Enryo
> 
```

Example answer

- Was your answer similar to this?
- The program flow is as follows
  - 1st line: assign user-inputted string to variable name
  - 2nd line: prints both “Hello,” and value contained in variable name

```
name = input('Your Name?')
print('Hello,', name)
```

Exercise 3: Calculating the area of circle

- In exercise 1, you have added print() to show the result on console
- Then, let’s further modify the program to let user input the value of radius “r”

How do we fix this?

```
pi = 3.14159
r = 5
print('The area of a circle with radius',
      r, 'is', pi * r * r)
```

Example answer

- The return value of input() is a string (of str type)
- Thus, you need to convert it to a number (int type or float type) if what you need is a number

```
pi = 3.14159
r = float(input('radius r='))
print('The area of a circle with radius',
      r, 'is', pi * r * r)
```

For converting values to float, we can use float() function!

2. Going further

Let’s learn some techniques needed
to make complex programs, or control program behavior precisely

Advanced usages of print()

- By giving labelled-arguments to function print(), you can change how values are displayed
  - sep= : Specifies separator, which is a string displayed between arguments (if not specified, it's ␣)
  - end= : Specified the string to be displayed at the end of print() (if not specified, it's new-line character)
  - You can specify sep=, end= both, or either one of them
- If you do not need any spaces between values, use `sep=""`(empty string); and if you do not need new line, you can add `end=""`
  - Example:

```
r = float(input('r='))
print('For r=', r, ', the area is', sep='', end='')
print(pi * (r ** 2), sep='')
```

Add empty lines & comments

- You can add comments to write notes or explanations for humans
  - By putting #(sharp), the continuing parts are considered as "comments", which Python does not execute
  - For example, you can add empty line and comments to help humans understand what the program means (purple parts are comments)

  -
    ```
    # radius
    r = 3

    # circle area dimension
    area = 3.14159 * r * r  # πr2
    ```

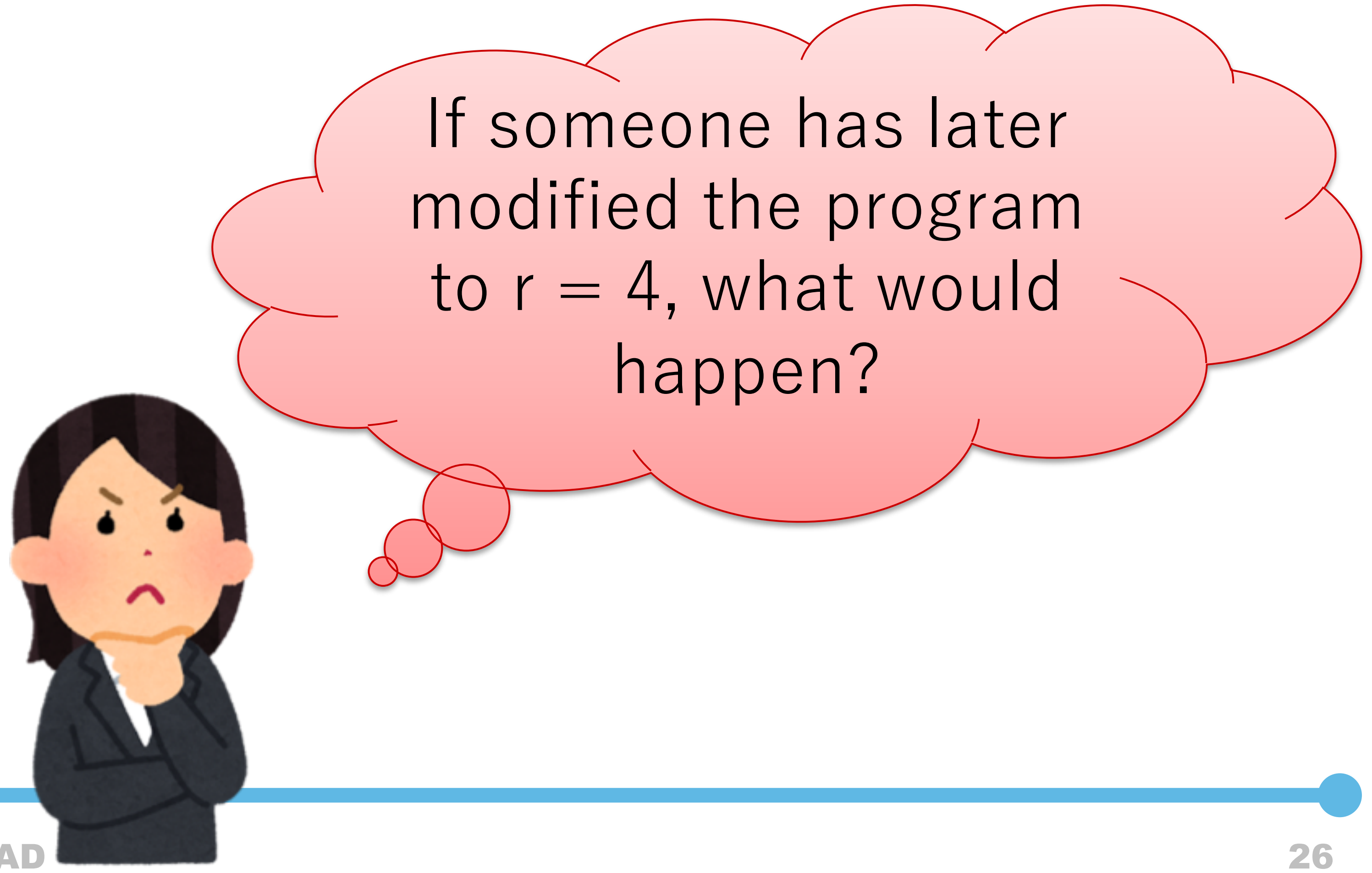
Comments can be useful, but …

- Only the program codes (non-comments) define how your program works
  - It is harmful to write the same thing as program code in a comment
  - Only use comments to explain what are not obvious from the program, to reinforce human understandings

- Bad example

  -
    ```
    # Let variable r be 3
    r = 3

    # Let area be πr2
    area = 3.14159 * r * r
    ```


If someone has later modified the program to r = 4, what would happen?

Continuing again and again

- By using「while True:」, you can let Python do the same thing for the infinite number of times
  - Computers are good at continuing the same things
  - The below structure of infinite continuation is called “**infinite loop(s)**”

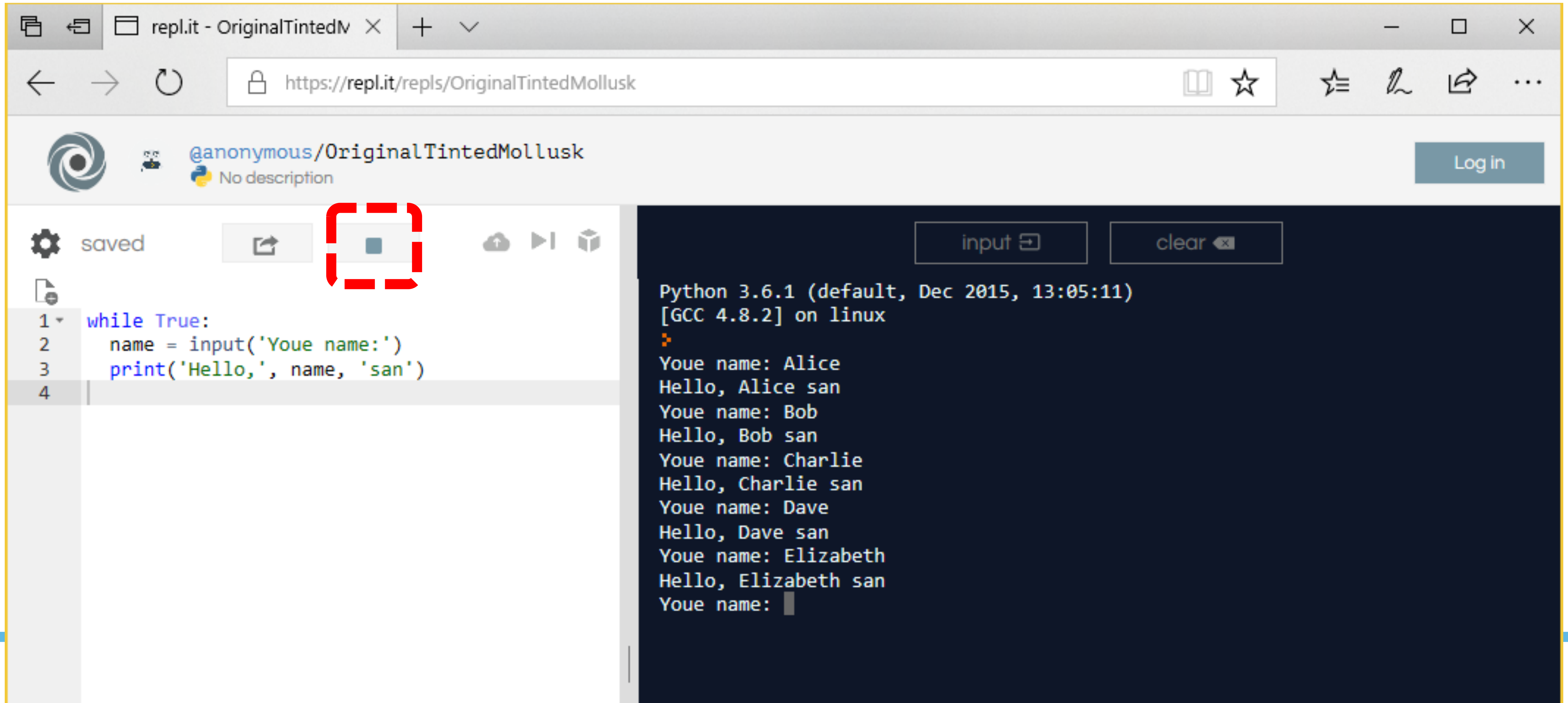
- Example: Actually run it yourself

  -
    ```
    while True:
      r = float(input('r='))
      print('The area of a circle is', pi * (r ** 2))
    ```

Just like the body of function definition using def,
you need similar amount of spaces (indentation) here

If you want to force a program to stop

- Using an infinite loop, the program will never stop.
- When you want to force the program to stop, press Stop [■] button

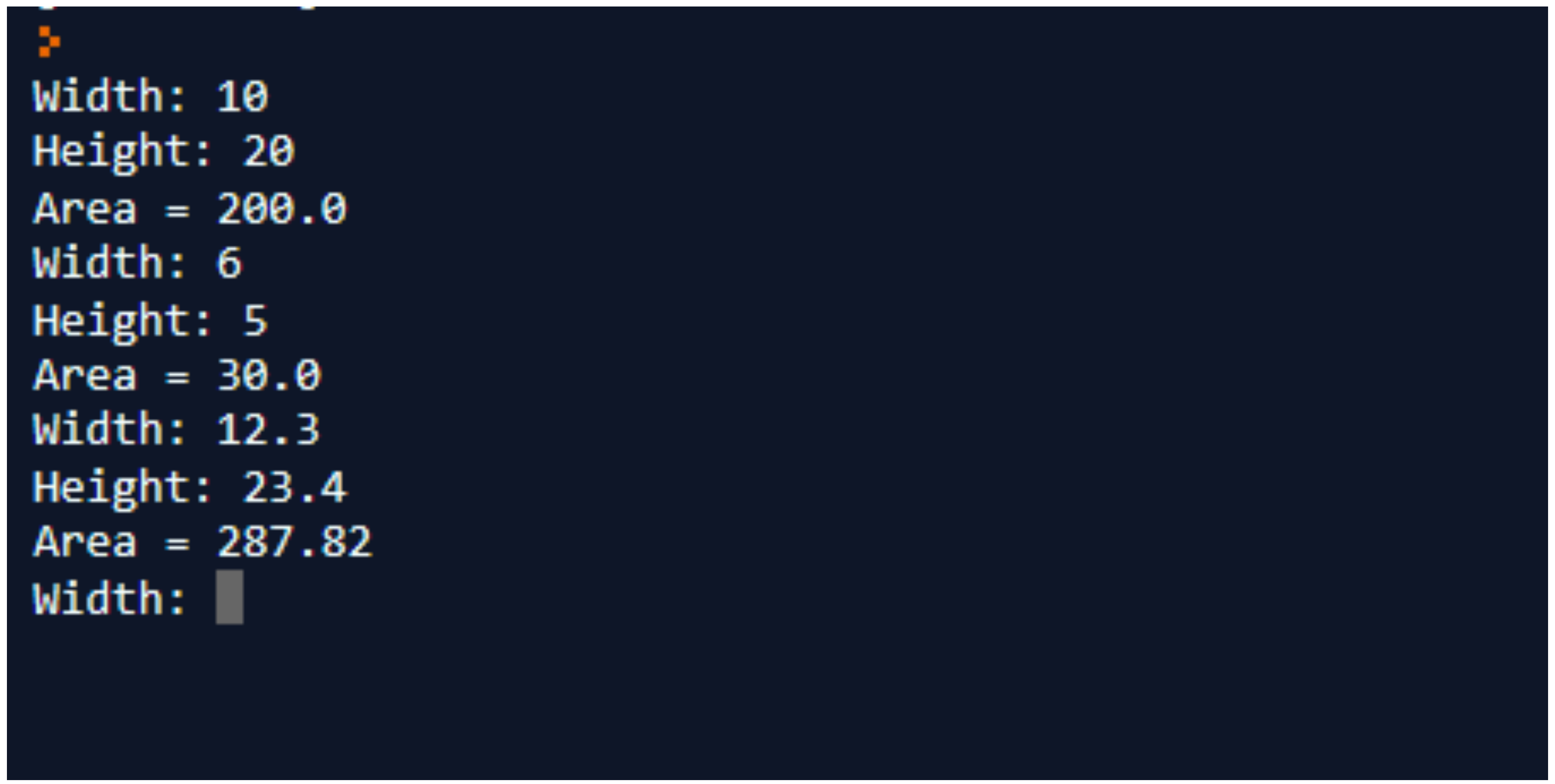

repl.it - OriginalTintedM
https://repl.it/repls/OriginalTintedMollusk
@anonymous/OriginalTintedMollusk
No description
Log in
saved
input
clear

```
while True:
  name = input('Youe name:')
  print('Hello,', name, 'san')
```

```
Python 3.6.1 (default, Dec 2015, 13:05:11)
[GCC 4.8.2] on linux
Youe name: Alice
Hello, Alice san
Youe name: Bob
Hello, Bob san
Youe name: Charlie
Hello, Charlie san
Youe name: Dave
Hello, Dave san
Youe name: Elizabeth
Hello, Elizabeth san
Youe name:
```

Exercise 4: Calculate rectangular area

- Make a program that lets user input width and height of a rectangle, shows its area, and continues to do that again and again



```
>
Width: 10
Height: 20
Area = 200.0
Width: 6
Height: 5
Area = 30.0
Width: 12.3
Height: 23.4
Area = 287.82
Width: 
```

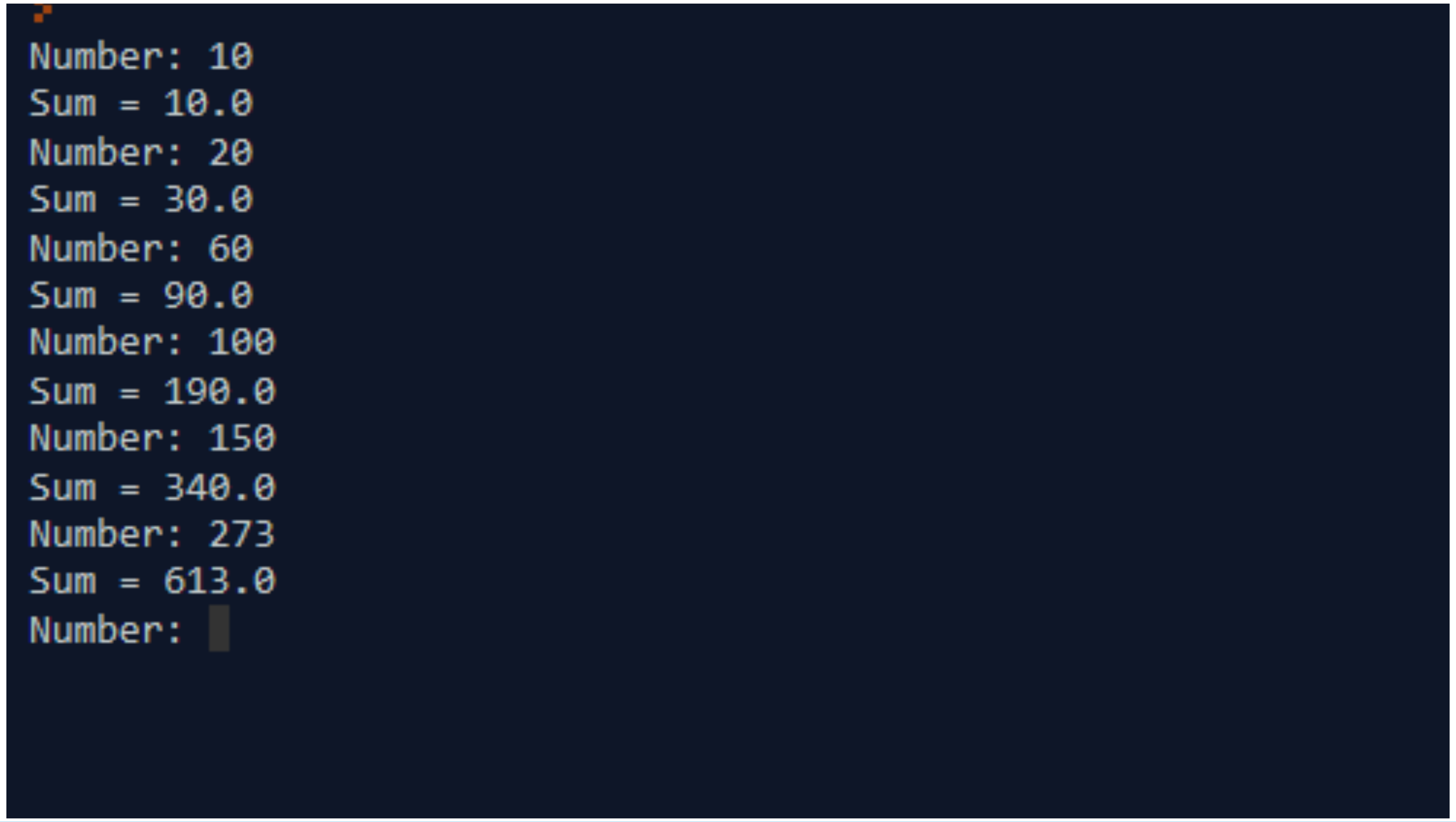
Example answer

- You could have solved this if you have comprehended how to use the techniques learned so far
  - Continue using “while True:”
  - Function input() lets user to input a string
  - Conversion of a string to float value can be done using float() function

```
while True:
  width = float(input('Width:’))
  height = float(input('Height:’))
  print('Area =', width * height)
```

Exercise 5: Calculate the total sum

- Create a program that lets user input numbers, displays the total sum of the numbers inputted so far, and continue to do that again and again



```
Number: 10
Sum = 10.0
Number: 20
Sum = 30.0
Number: 60
Sum = 90.0
Number: 100
Sum = 190.0
Number: 150
Sum = 340.0
Number: 273
Sum = 613.0
Number: 
```

Example answer

- Did you notice that you need to “update” the total sum?
  - To update a value, you can use the augmented assignment operator you learned last class
- Start from total of nothing, which is zero (0)

```
# The total sum starts from 0
total = 0

while True:
  n = float(input('Number:'))
  total += n
  print('Sum =', total)
```

On each iteration, update by adding n to variable total