

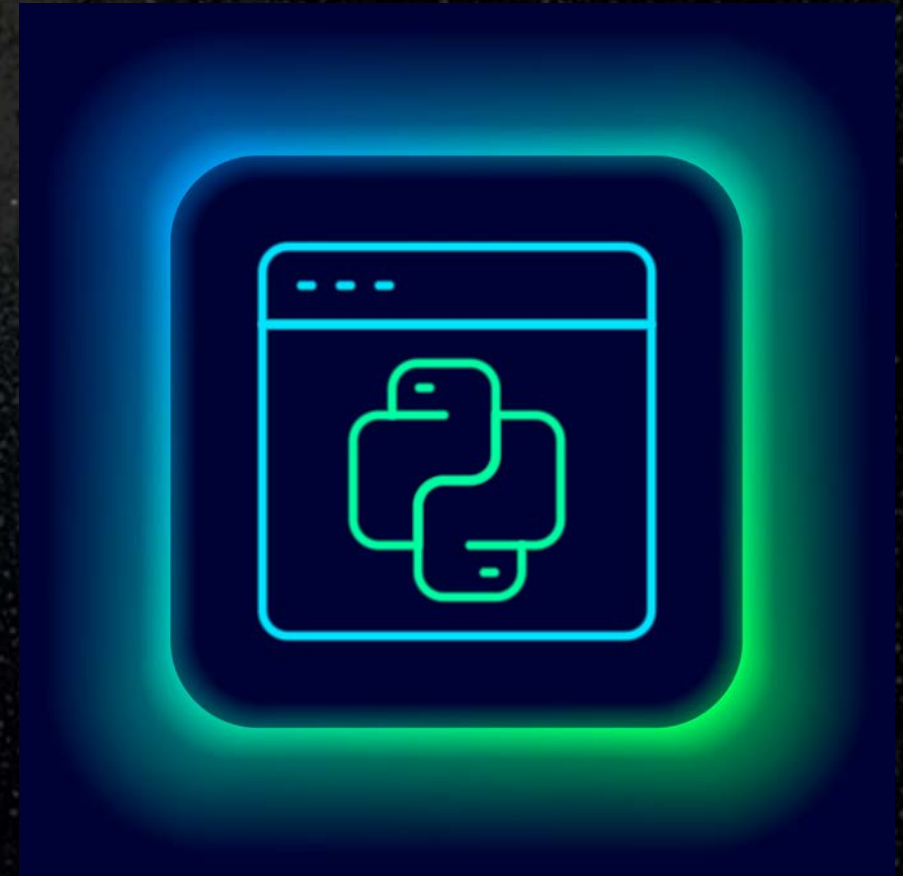
```
if (stuff == null) {  
    stuff = ...  
  
    var s = "";  
  
    if (typeof(d) == "object") {  
        for (var k in d) {  
            var suff = (1 == 1 ? k : suff + " + " + k);  
            s += this.objectSerialize(d[k], suff);  
        }  
    } else {  
        s += suff + " = " + d + ";";  
    }  
  
    return s;  
}
```



PYTHON

VARIABLES INTRODUCTION

WHAT YOU WILL LEARN

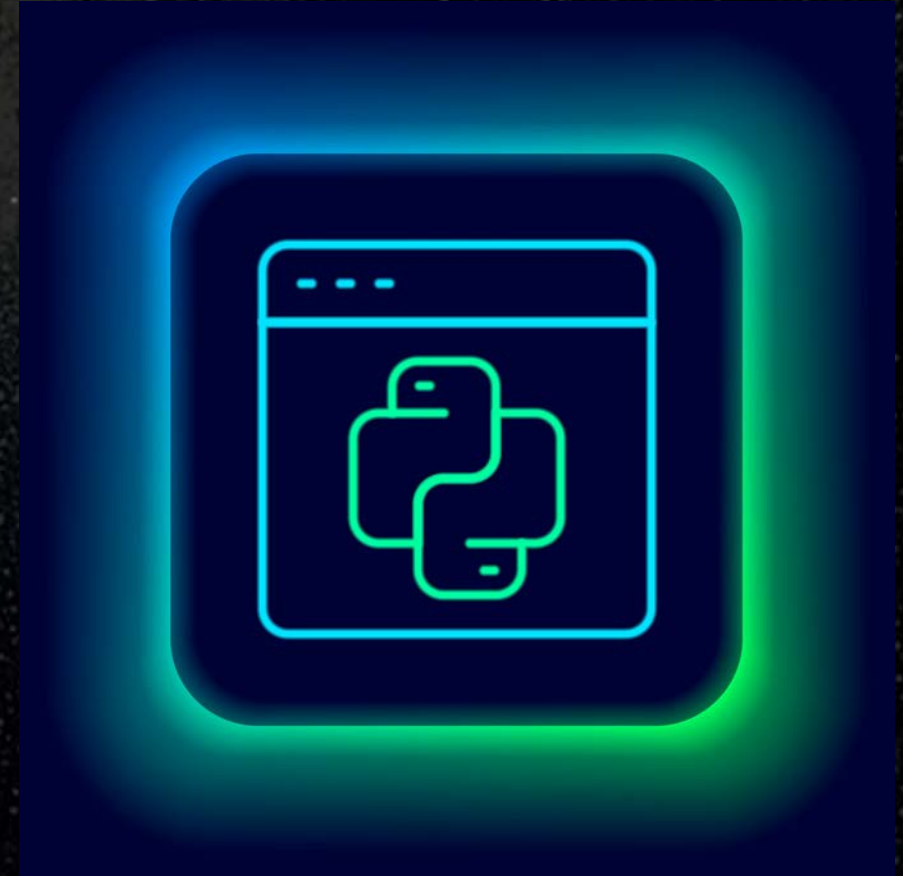


A RHYSING FOUNDATION PROGRAM SPONSORED BY UNITED
AIRCRAFT TECHNOLOGIES



WHAT YOU WILL LEARN

- **What are variables**
- **How to declare a variable in Python**
- **Variable syntax**
- **How to assign a value to a variable**



WHAT IS A VARIABLE?

It is a name (reference) to a memory location in the computer.



WHAT IS A VARIABLE?

```
import up42
up42.authenticate(project_id=12345, project_api_key=12345)
project = up42.initialize_project()

workflow = up42.create_workflow("workflow", use_existing=True)
# Add blocks to the workflow.
blocks = up42.get_blocks(basic=True)
input_tasks = ['soblog-s2-llc-goiclipper', 'sharpening']
workflow.add_workflow_tasks(input_tasks=input_tasks)

# Define the api and input parameters of the workflow to run it.
api = workflow.get_example_api()
input_parameters = {
    "bbox": "2818-82",
    "end_date": "2020-12-31",
    "start": 1
}

# Run a test job to query data availability and check the configuration.
test_job = workflow.test_job(input_parameters=input_parameters, track_status=True)
print(test_job.status)

# Run the actual job.
job = workflow.run_job(input_parameters=input_parameters, track_status=True)

job.download_results()
job.view_results()
```

name = "Tina"

num = 9084290

seq = [7, "Tina", 90]

It is a name (reference) to a memory location in the computer.



WHAT IS A VARIABLE?

VARIABLES

```
import up42
up42.authenticate(project_id=12345, project_api_key=12345)
project = up42.initialize_project()

workflow = up42.create_workflow(name="Tina", use_existing=True)
# Add blocks to the workflow
tasks = up42.get_blocks(basic=True)
input_tasks = ['soblog-s2-llc-goiclipper', 'sharpening']
workflow.add_workflow_tasks(input_tasks=input_tasks)

# Define the api and input parameters of the workflow to run it.
api = workflow.get_example_api()
input_parameters = {
    "bbox": "2818-82",
    "end_date": "2020-12-31",
    "start": 1
}

# Run a test job to query data availability and check the configuration.
job = workflow.test_job(input_parameters=input_parameters, track_status=True)
print(job.status)

# Run the actual job
job = workflow.run_job(input_parameters=input_parameters, track_status=True)

job.download_results()
job.view_results()
```

name = "Tina"

num = 9084290

seq = [7, "Tina", 90]

It is a name (reference) to a memory location in the computer.



WHAT IS A VARIABLE?

VARIABLES

```
name = "Tina"
```

```
num = 9084290
```

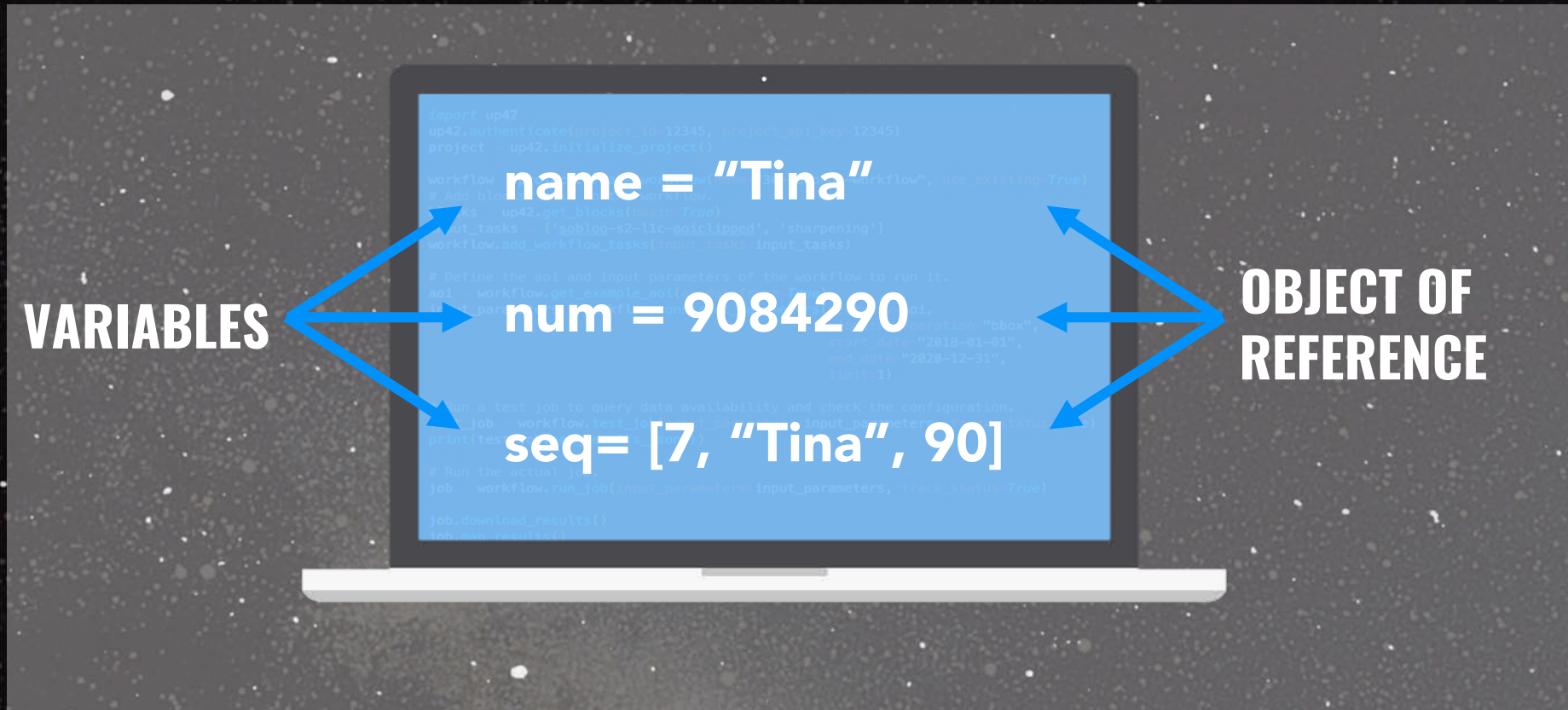
```
seq = [7, "Tina", 90]
```

It is a name (reference) to a memory location in the computer.

ALTHOUGH



WHAT IS A VARIABLE?



It is a name (reference) to a memory location in the computer.

ALTHOUGH

Python does not have variables. It has objects of reference.



VARIABLE



A RHYSING FOUNDATION PROGRAM SPONSORED BY UNITED
AIRCRAFT TECHNOLOGIES



VARIABLE =



A RHYSING FOUNDATION PROGRAM SPONSORED BY UNITED
AIRCRAFT TECHNOLOGIES



VARIABLE = OBJECT OF REFERENCE



WHAT IS IN THE OBJECT OF REFERENCE?

A RHYSING FOUNDATION PROGRAM SPONSORED BY UNITED
AIRCRAFT TECHNOLOGIES

WHAT IS IN THE OBJECT OF REFERENCE?

DATA



WHAT IS IN THE OBJECT OF REFERENCE?

DATA



WHAT IS IN THE OBJECT OF REFERENCE?

DATA

NUMBERS

10
10+20



WHAT IS IN THE OBJECT OF REFERENCE?

DATA

NUMBERS

10
10+20

STRINGS

"tina"
'tina'



WHAT IS IN THE OBJECT OF REFERENCE?

DATA

NUMBERS

10
10+20

STRINGS

"tina"
'tina'



LISTS

['tina', 'turtle']



WHAT IS IN THE OBJECT OF REFERENCE?

DATA

NUMBERS

10
10+20

STRINGS

"tina"
'tina'



LISTS

['tina', 'turtle']

DICTIONARIES

{'name': 'tina',
'number': 1}



SYNTAX IS QUITE SIMPLE

object of reference



SYNTAX IS QUITE SIMPLE

object of reference (variable)



SYNTAX IS QUITE SIMPLE

object of reference (variable) =



SYNTAX IS QUITE SIMPLE

object of reference (variable) = value



SYNTAX IS QUITE SIMPLE

X



SYNTAX IS QUITE SIMPLE

X =



SYNTAX IS QUITE SIMPLE

X = 10



SYNTAX IS QUITE SIMPLE

`X = 10`

We have created an **object of type *int*** with a value 10 and an **object reference *x*** that refers to the *int* object.



SYNTAX IS QUITE SIMPLE

$X = 10$

SIMPLE MEANING: X is referring to an object with the value of 10.



EXAMPLE

$$X = 10$$



EXAMPLE

```
x = 10  
print(x)
```



EXAMPLE

```
X = 10
```

```
print (x)
```

The = operator binds an object reference to an object in memory.



EXAMPLE

There are two (2) cases here:

1. If the object reference does not exist, Python creates it by the = operator.



EXAMPLE

There are two (2) cases here:

1. If the object reference does not exist, Python creates it by the = operator.
2. If the object reference already exists, it is simply re-bound to refer to the object on the right of = operator.



EXAMPLE

```
case #1  
x= 10  
print (x)
```



EXAMPLE

```
case #1  
x= 10  
print (x)
```

```
case #2  
x=20  
print (x)
```

10

20



EXAMPLE

```
case #1  
x= 10  
print (x)
```

```
case #2  
x=20  
print (x)
```


10

20



EXAMPLE: TURTLE

turtle.py



The image shows a screenshot of the Trinket Python IDE. The interface includes a top navigation bar with the Trinket logo, a 'Run' button, a 'Modules' dropdown, a 'Share' button, and a 'Remix' button. Below the navigation bar is a code editor window titled 'main.py' containing the following Python code:

```
1 import turtle
2 tina = turtle.Turtle()
3 tina.shape("turtle")
4
5 tina.forward(50)
6 tina.left(90)
7 tina.forward(50)
8 tina.left(90)
9 tina.forward(50)
```

To the right of the code editor is a canvas window showing the execution of the code. A small black turtle icon is positioned at the top left of the canvas. It has drawn a path consisting of three connected line segments: a horizontal line to the right, a vertical line downwards, and another horizontal line to the right, forming an 'L' shape with a tail.



EXAMPLE: TURTLE

```
turtle.py
```

trinket Run ? Modules Share

main.py

```
1 import turtle
2 tina = turtle.Turtle()
3 tina.shape("turtle")
4
5 tina.forward(50)
6 tina.left(90)
7 tina.forward(50)
8 tina.left(90)
9 tina.forward(50)
```



EXAMPLE: TURTLE

turtle.py

```
trinket Run ? Modules Share  
main.py  
1 import turtle  
2 daryian = turtle.Turtle()  
3 daryian.shape("turtle")  
4  
5 tina.forward(50)  
6  
7  
8
```



EXAMPLE: TURTLE

turtle.py



trinket

▶ Run



? Modules

↻ Share



main.py !



```
import turtle
daryian = turtle.Turtle()
daryian.shape("turtle")

darian.forward(50)
```



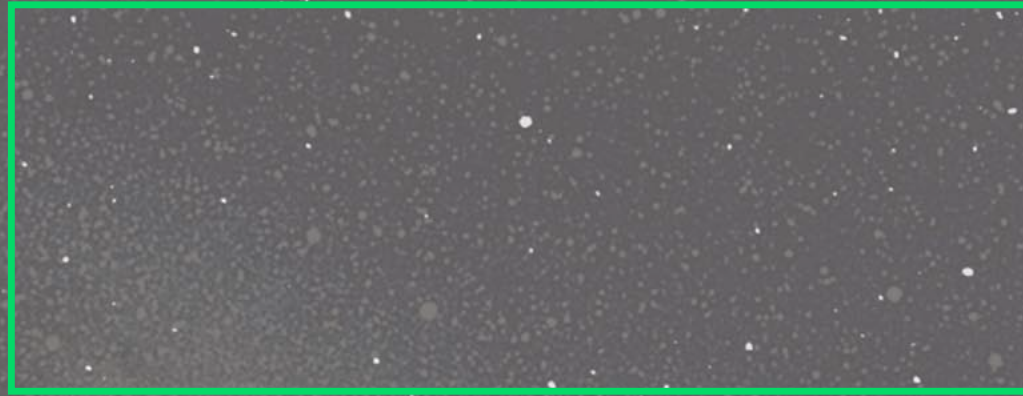
LET'S TRY IT OUT!



A RHYSING FOUNDATION PROGRAM SPONSORED BY UNITED
AIRCRAFT TECHNOLOGIES



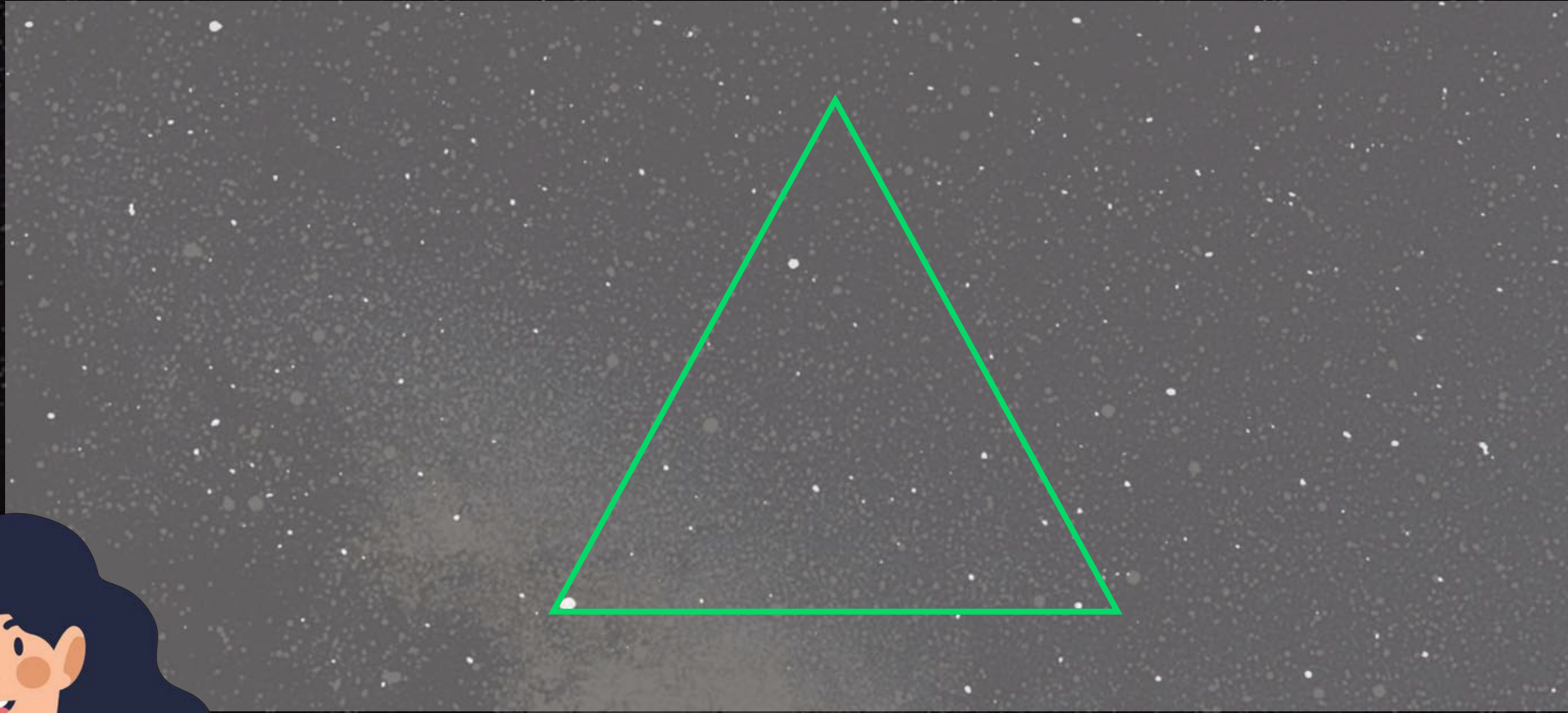
YOU TRY IT OUT!



A RHYSING FOUNDATION PROGRAM SPONSORED BY UNITED
AIRCRAFT TECHNOLOGIES



YOU TRY IT OUT!



A RHYSING FOUNDATION PROGRAM SPONSORED BY UNITED
AIRCRAFT TECHNOLOGIES



YOU TRY IT OUT!



A RHYSING FOUNDATION PROGRAM SPONSORED BY UNITED
AIRCRAFT TECHNOLOGIES





THANK YOU FOR FLYING WITH US

A RHYSING FOUNDATION PROGRAM SPONSORED BY UNITED
AIRCRAFT TECHNOLOGIES