Week 1

CS106R

Sabri **Eyuboglu** & Geoffrey **Angus**

Python

print( )

```
print(“Message goes here”)
```

Example

Python Code

```
print("I am a Python program!")
```

Result

```
> I am a Python program!
```

PyBot **Functions**

Example: Print Message

Today's **Exercises**

Print Name

Harvest Your First Fruit

Turn Left

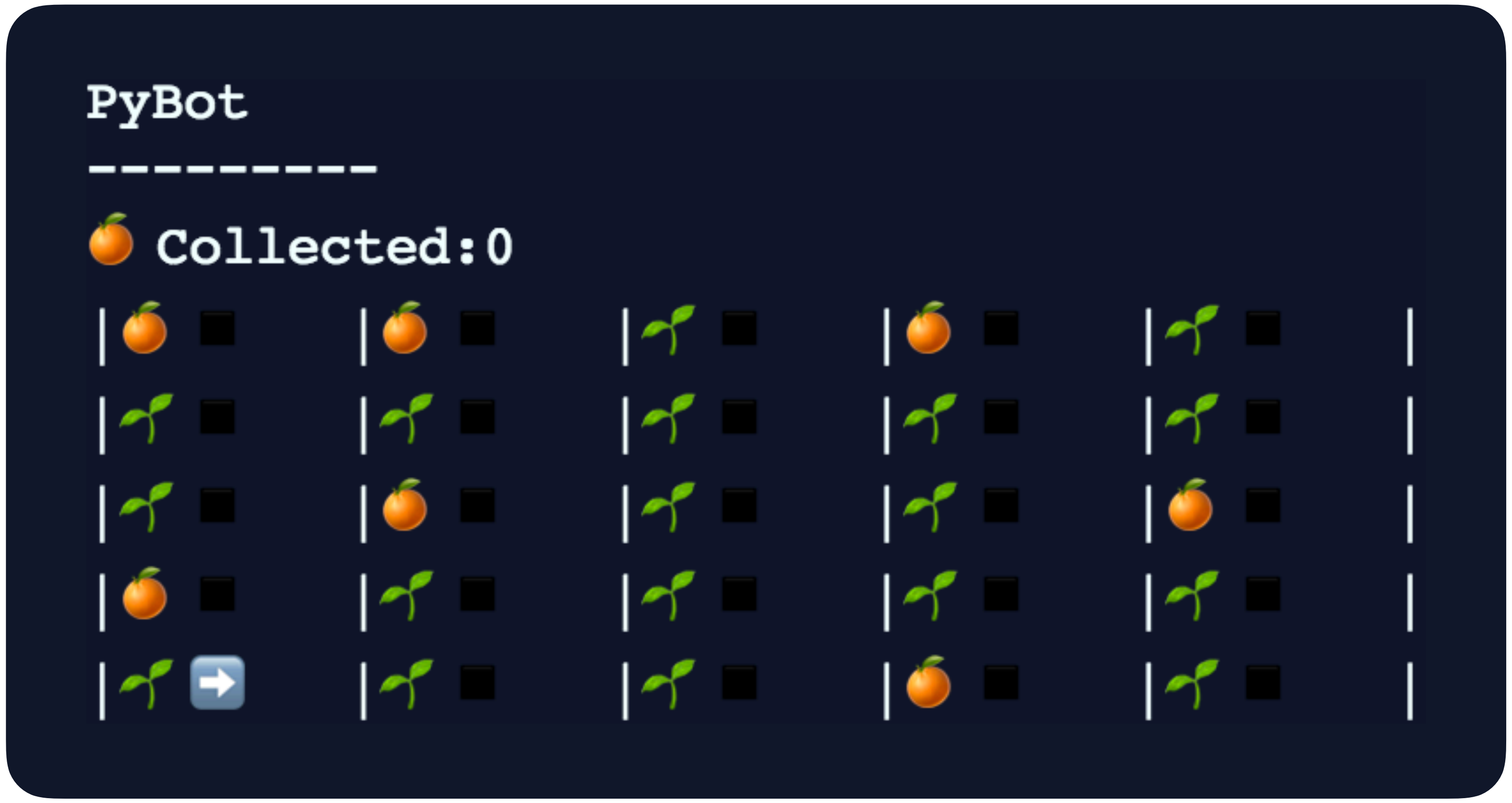
Introducing **PyBot**

➡️

PyCountry **Fields**

NORTH

WEST

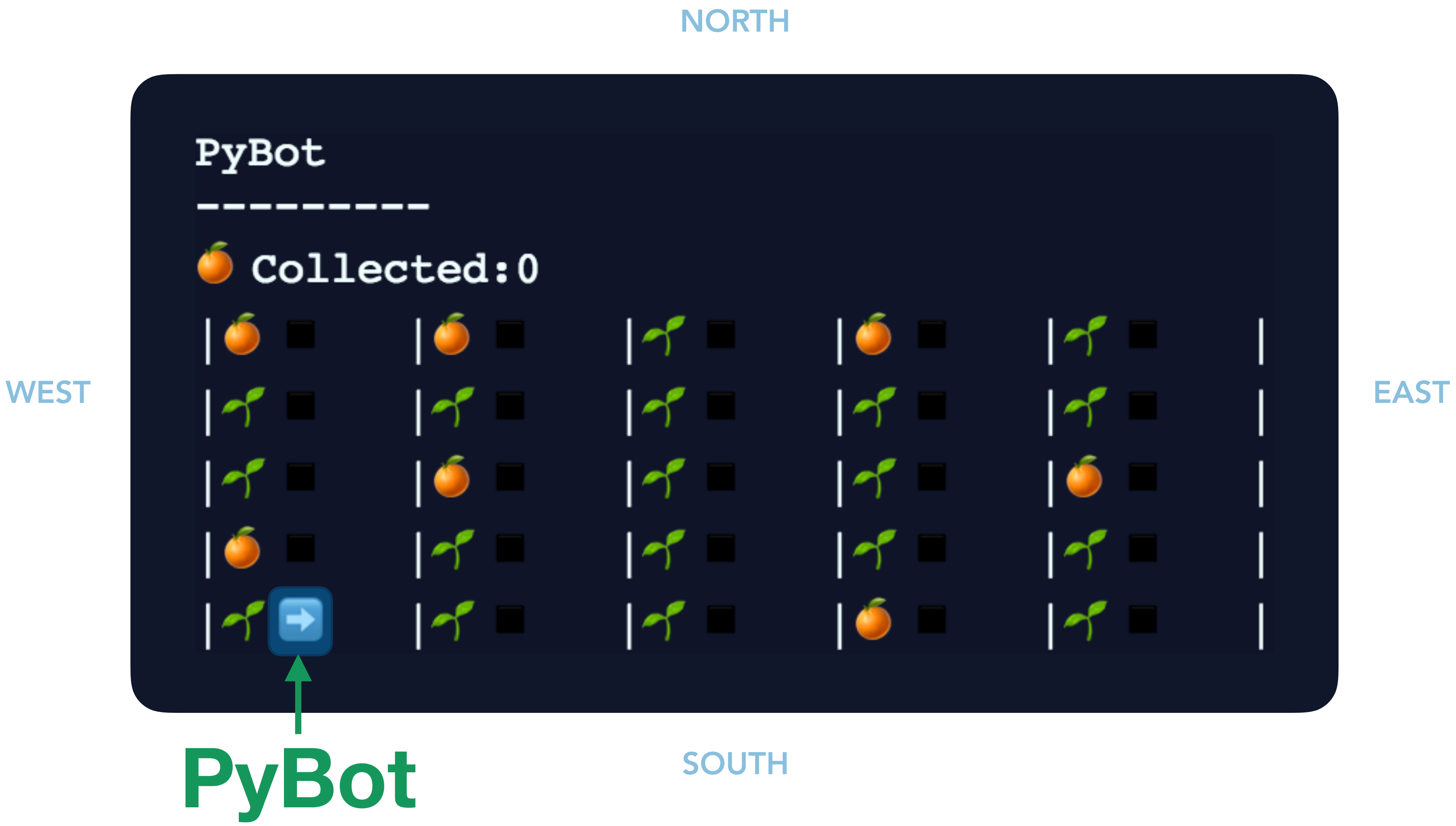


```
PyBot
---------
🍊 Collected:0
|🍊 ⬛     |🍊 ⬛     |🌱 ⬛     |🍊 ⬛     |🌱 ⬛     |
|🌱 ⬛     |🌱 ⬛     |🌱 ⬛     |🌱 ⬛     |🌱 ⬛     |
|🌱 ⬛     |🍊 ⬛     |🌱 ⬛     |🌱 ⬛     |🍊 ⬛     |
|🍊 ⬛     |🌱 ⬛     |🌱 ⬛     |🌱 ⬛     |🌱 ⬛     |
|🌱 ➡️     |🌱 ⬛     |🌱 ⬛     |🍊 ⬛     |🌱 ⬛     |
```

EAST

SOUTH

PyCountry **Fields**

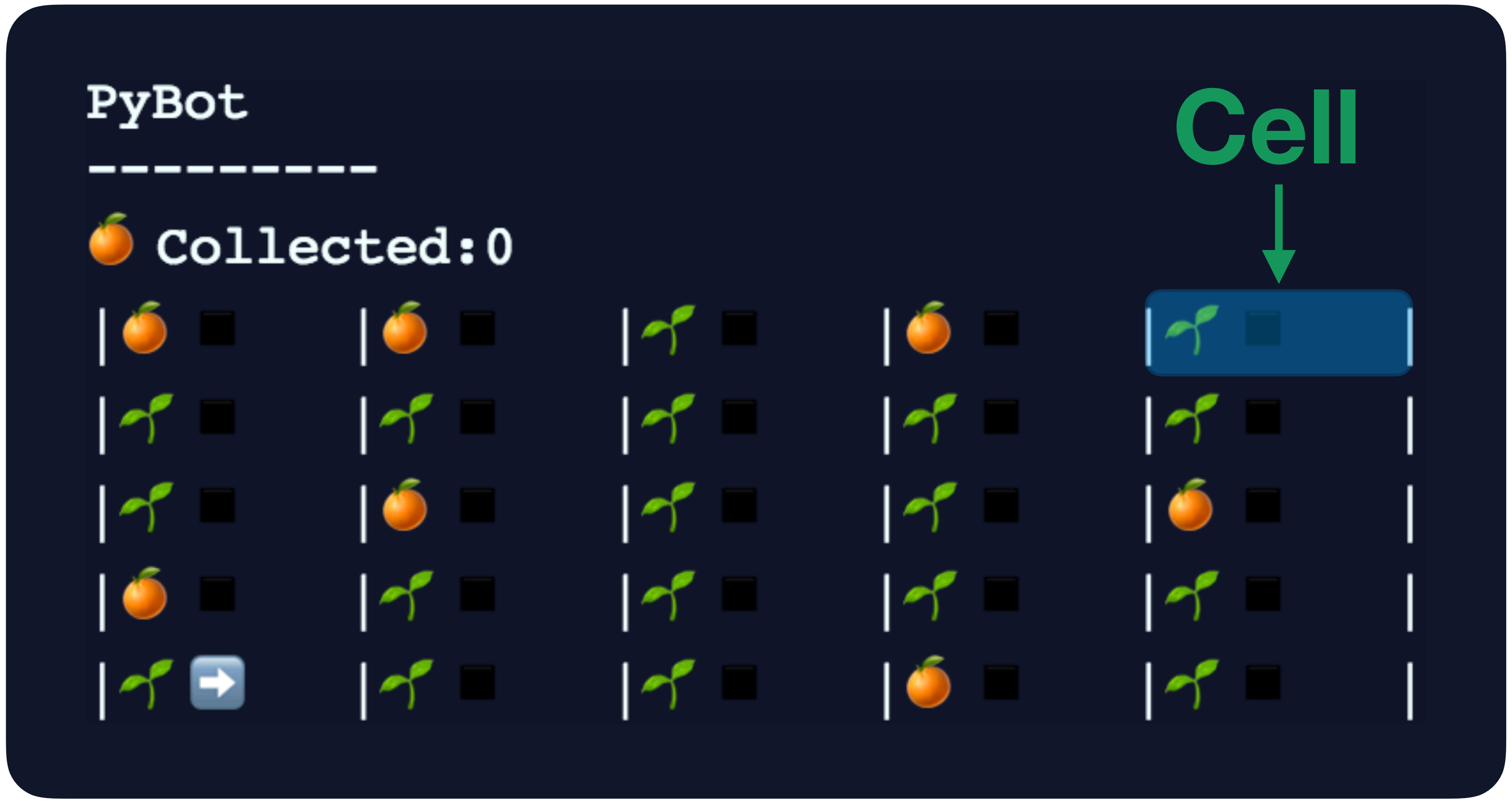


```
PyBot
----------
🍊 Collected:0
|🍊 ⬛    |🍊 ⬛    |🌱 ⬛    |🍊 ⬛    |🌱 ⬛    |
|🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |
|🌱 ⬛    |🍊 ⬛    |🌱 ⬛    |🌱 ⬛    |🍊 ⬛    |
|🍊 ⬛    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |
|🌱 ➡️    |🌱 ⬛    |🌱 ⬛    |🍊 ⬛    |🌱 ⬛    |
```

PyCountry **Fields**

NORTH

WEST



```
PyBot
---------
🍊 Collected:0
|🍊 ⬛     |🍊 ⬛     |🌱 ⬛     |🍊 ⬛     |🌱 ⬛     |
|🌱 ⬛     |🌱 ⬛     |🌱 ⬛     |🌱 ⬛     |🌱 ⬛     |
|🌱 ⬛     |🍊 ⬛     |🌱 ⬛     |🌱 ⬛     |🍊 ⬛     |
|🍊 ⬛     |🌱 ⬛     |🌱 ⬛     |🌱 ⬛     |🌱 ⬛     |
|🌱 ➡️     |🌱 ⬛     |🌱 ⬛     |🍊 ⬛     |🌱 ⬛     |
```

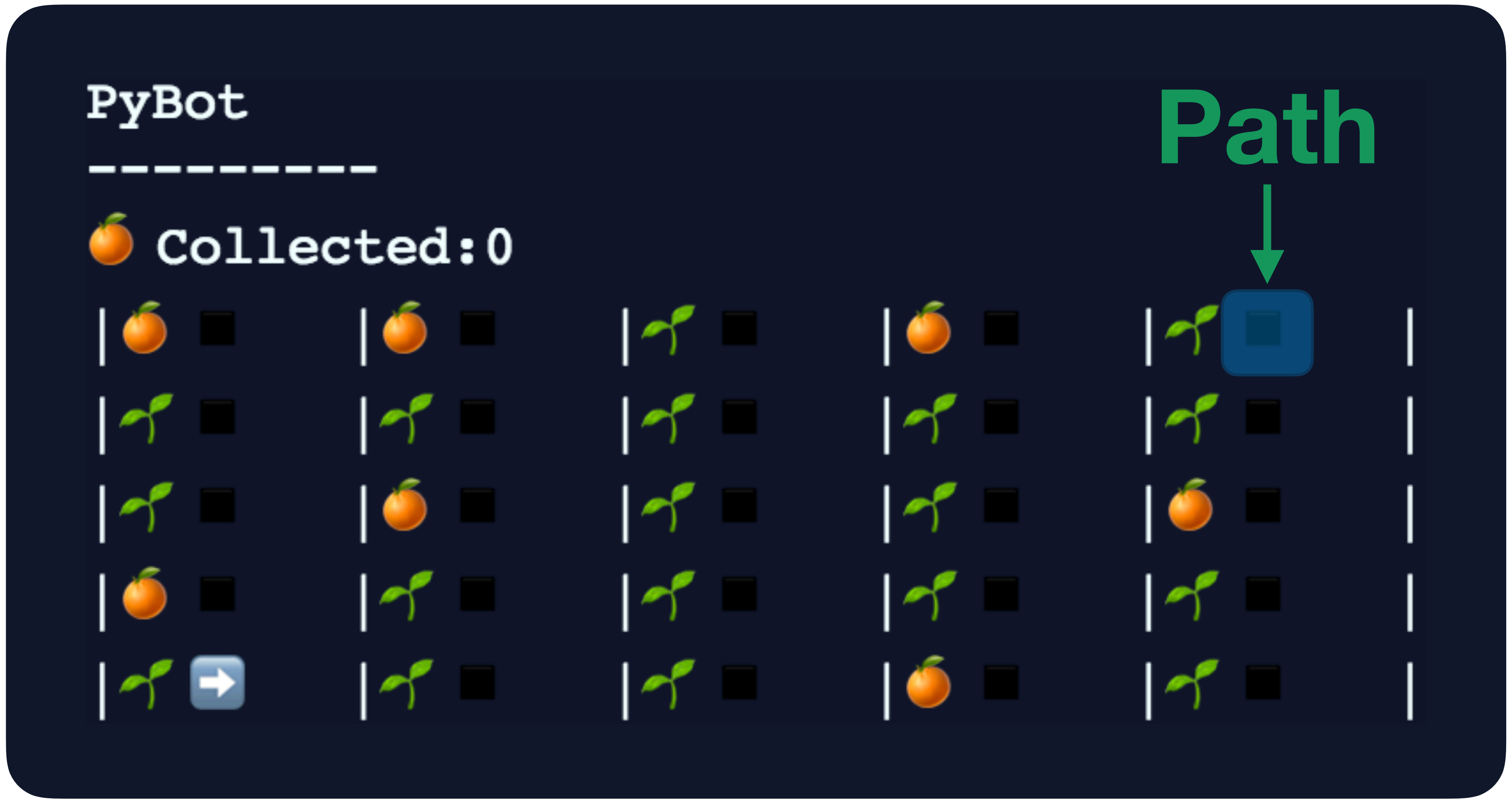
EAST

SOUTH

PyCountry **Fields**

NORTH

WEST



```
PyBot
---------
🍊 Collected:0
|🍊 ■    |🍊 ■    |🌱 ■    |🍊 ■    |🌱 ■    |
|🌱 ■    |🌱 ■    |🌱 ■    |🌱 ■    |🌱 ■    |
|🌱 ■    |🍊 ■    |🌱 ■    |🌱 ■    |🍊 ■    |
|🍊 ■    |🌱 ■    |🌱 ■    |🌱 ■    |🌱 ■    |
|🌱 ➡️    |🌱 ■    |🌱 ■    |🍊 ■    |🌱 ■    |
```

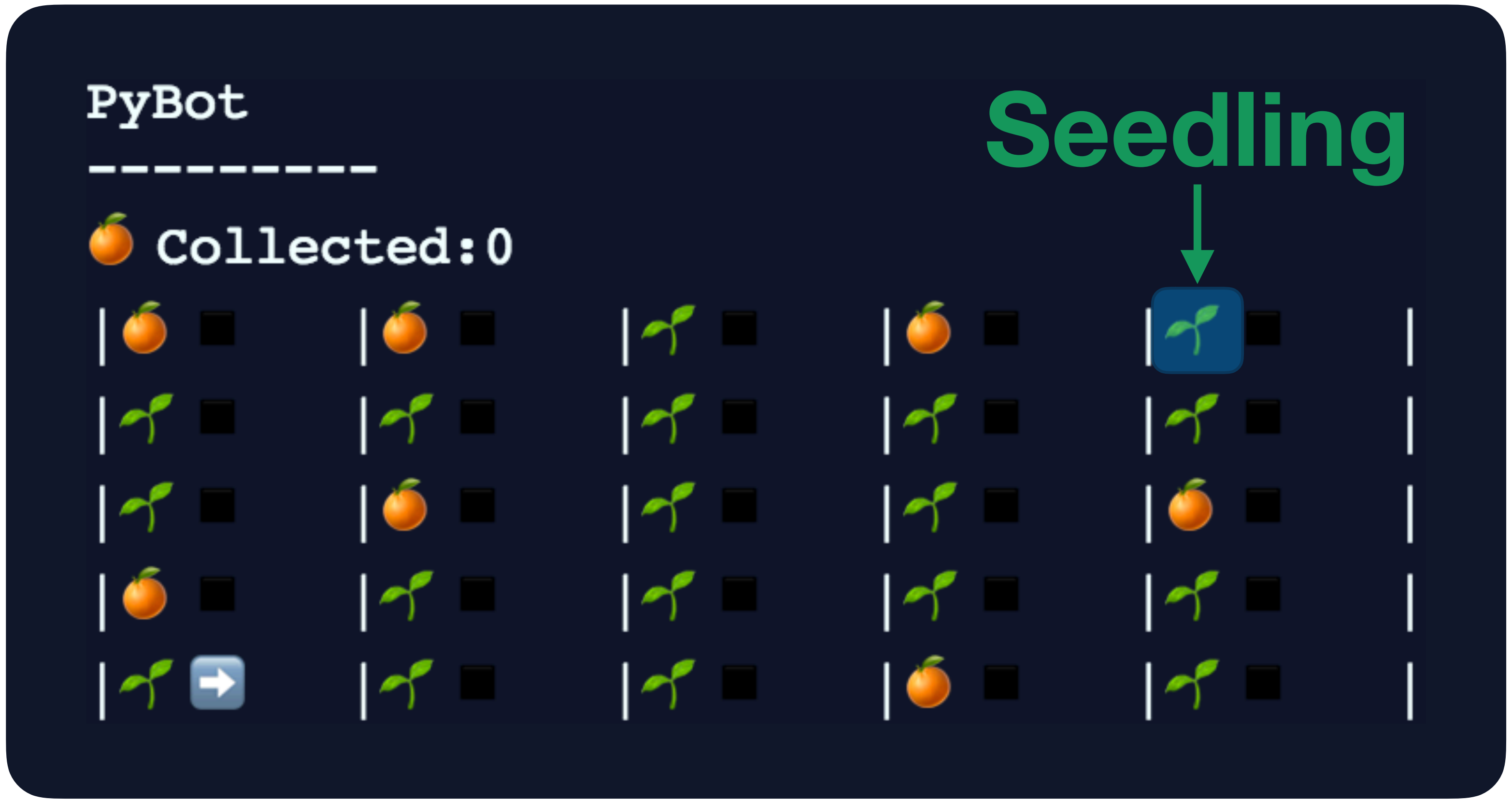
EAST

SOUTH

PyCountry **Fields**

NORTH

WEST



```
PyBot
----------
🍊 Collected:0
|🍊 ⬛ |🍊 ⬛ |🌱 ⬛ |🍊 ⬛ |🌱 ⬛ |
|🌱 ⬛ |🌱 ⬛ |🌱 ⬛ |🌱 ⬛ |🌱 ⬛ |
|🌱 ⬛ |🍊 ⬛ |🌱 ⬛ |🌱 ⬛ |🍊 ⬛ |
|🍊 ⬛ |🌱 ⬛ |🌱 ⬛ |🌱 ⬛ |🌱 ⬛ |
|🌱 ➡️ |🌱 ⬛ |🌱 ⬛ |🍊 ⬛ |🌱 ⬛ |
```

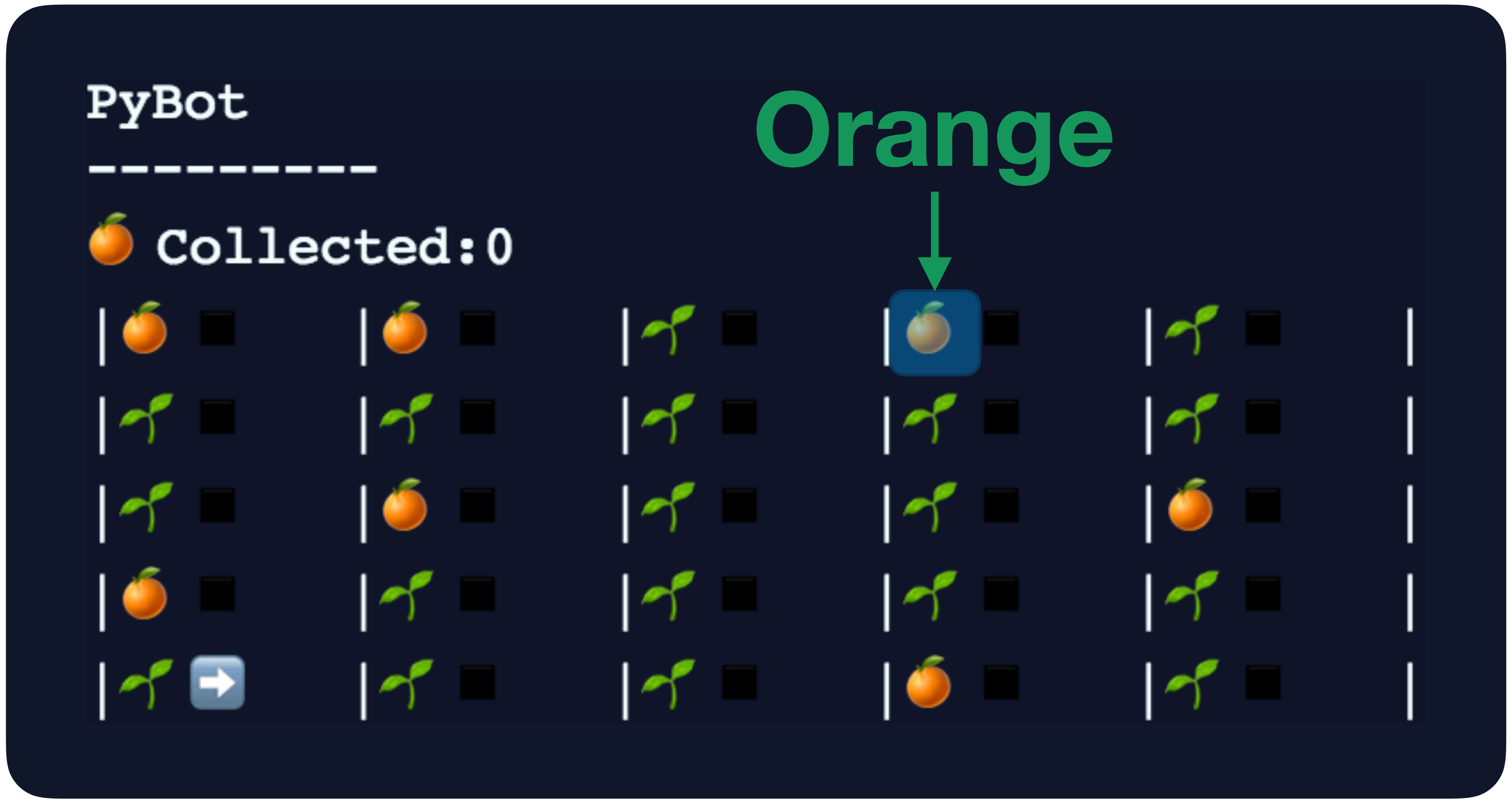
EAST

SOUTH

PyCountry **Fields**

NORTH

WEST

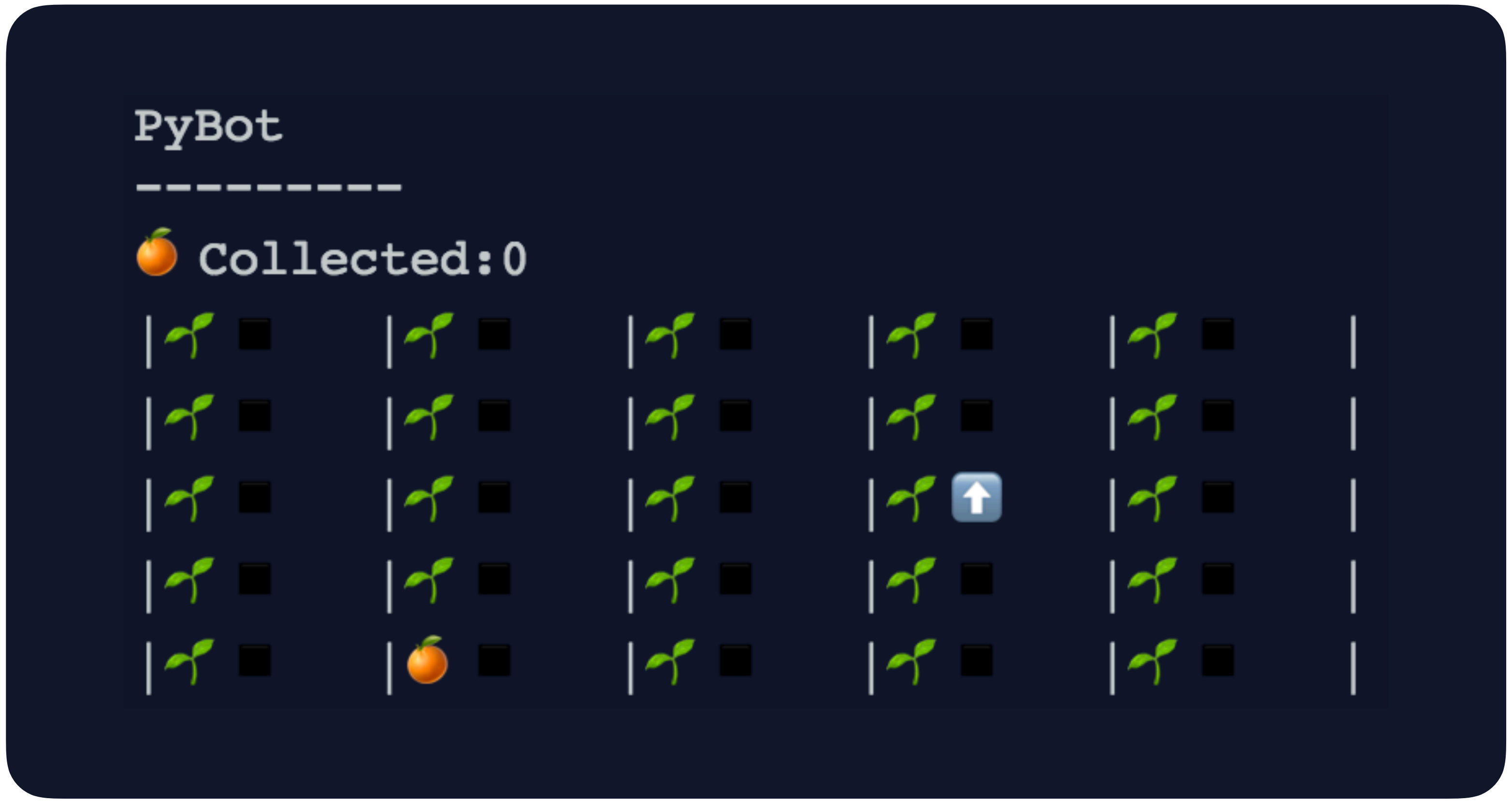


```
PyBot
---------
🍊 Collected:0
|🍊 ■    |🍊 ■    |🌱 ■    |🍊 ■    |🌱 ■    |
|🌱 ■    |🌱 ■    |🌱 ■    |🌱 ■    |🌱 ■    |
|🌱 ■    |🍊 ■    |🌱 ■    |🌱 ■    |🍊 ■    |
|🍊 ■    |🌱 ■    |🌱 ■    |🌱 ■    |🌱 ■    |
|🌱 ➡️    |🌱 ■    |🌱 ■    |🍊 ■    |🌱 ■    |
```

EAST

SOUTH

PyBot **Position**



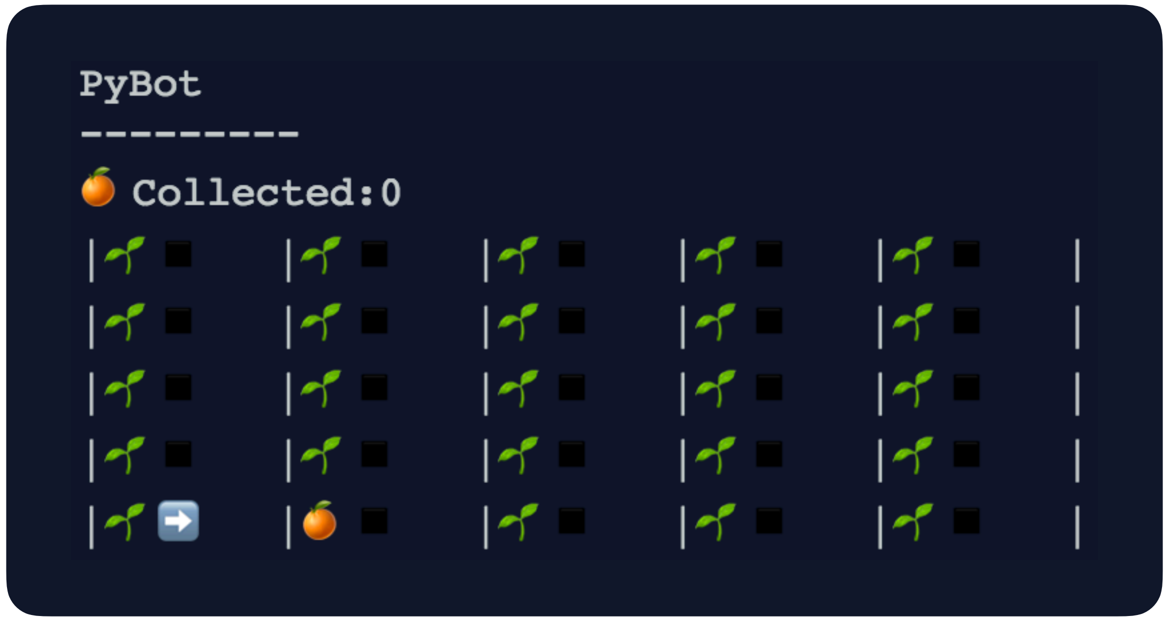
```
PyBot
---------
🍊 Collected:0
|🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |
|🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |
|🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |🌱 ⬆️   |🌱 ⬛   |
|🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |
|🌱 ⬛   |🍊 ⬛   |🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |
```

What can **PyBot** do?

PyBot **Actions**

Move!

PyBot moves forward one cell in the direction she is facing.



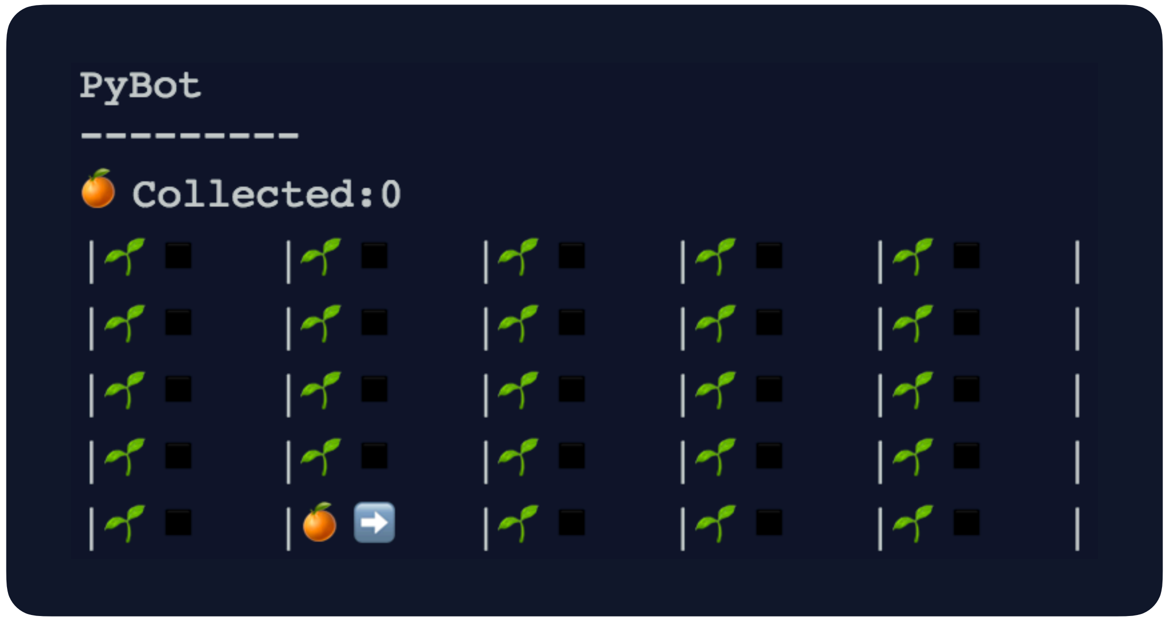
```
PyBot
----------
🍊 Collected:0
|🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |
|🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |
|🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |
|🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |
|🌱 ➡️    |🍊 ⬛    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |
```

BEFORE

PyBot **Actions**

Move!

PyBot moves forward one cell in the direction she is facing.



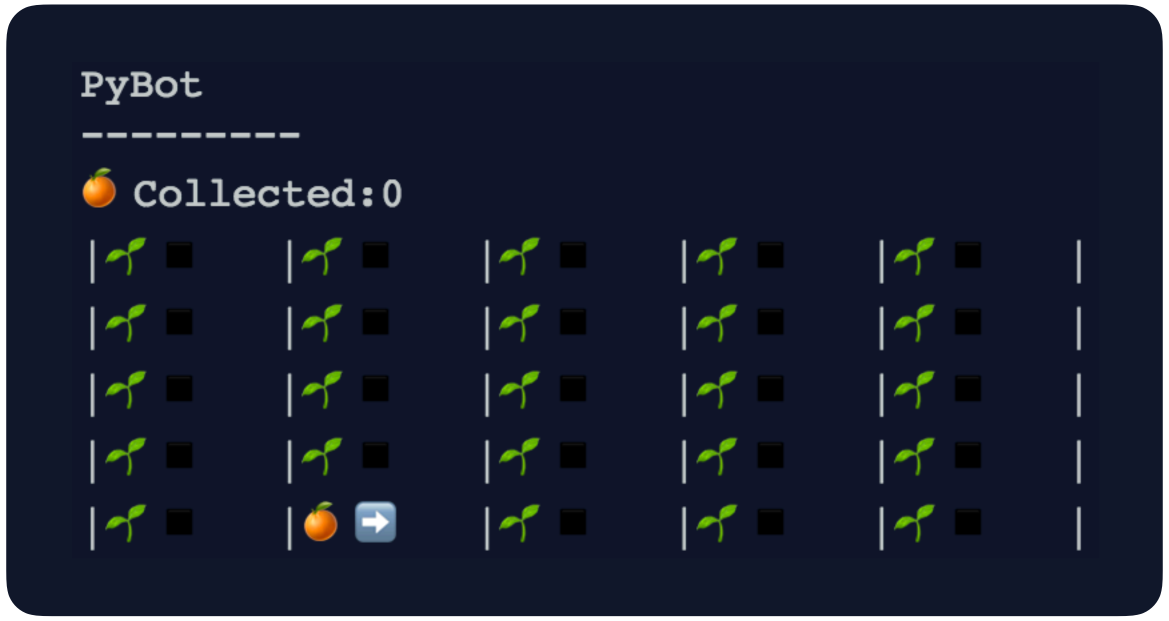
```
PyBot
---------
🍊 Collected:0
|🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |
|🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |
|🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |
|🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |
|🌱 ⬛    |🍊 ➡️    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |
```

AFTER

PyBot **Actions**

Pick Oranges!

PyBot picks the fruit in the current cell.



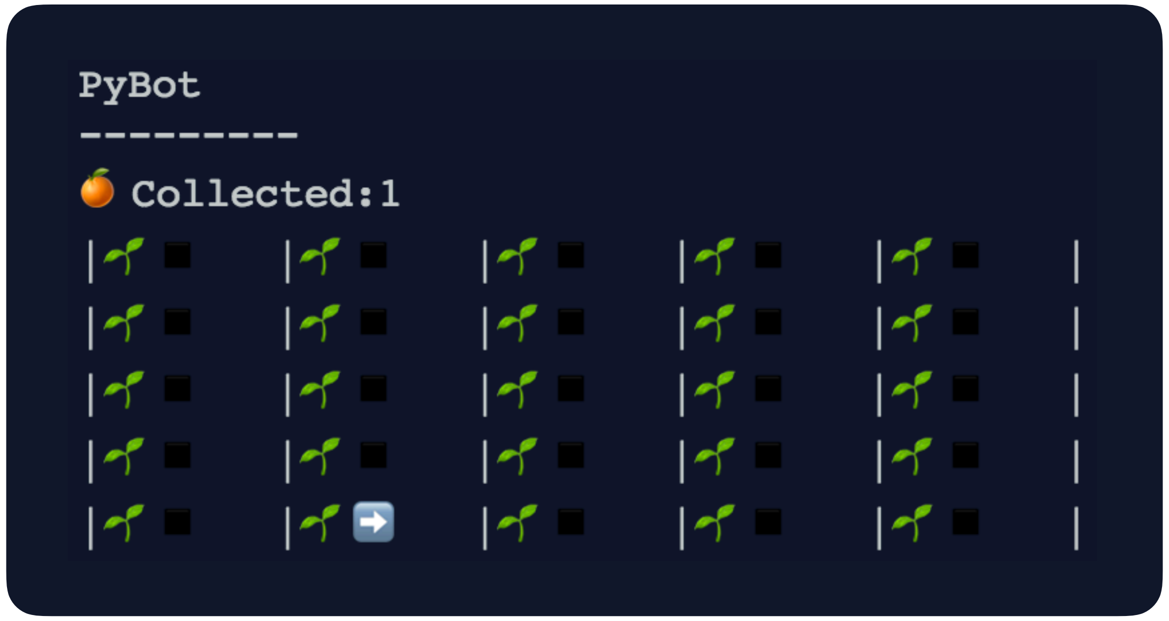
```
PyBot
----------
🍊 Collected:0
|🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |
|🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |
|🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |
|🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |
|🌱 ⬛   |🍊 ➡️   |🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |
```

BEFORE

PyBot **Actions**

Pick Oranges!

PyBot picks the fruit in the current cell.

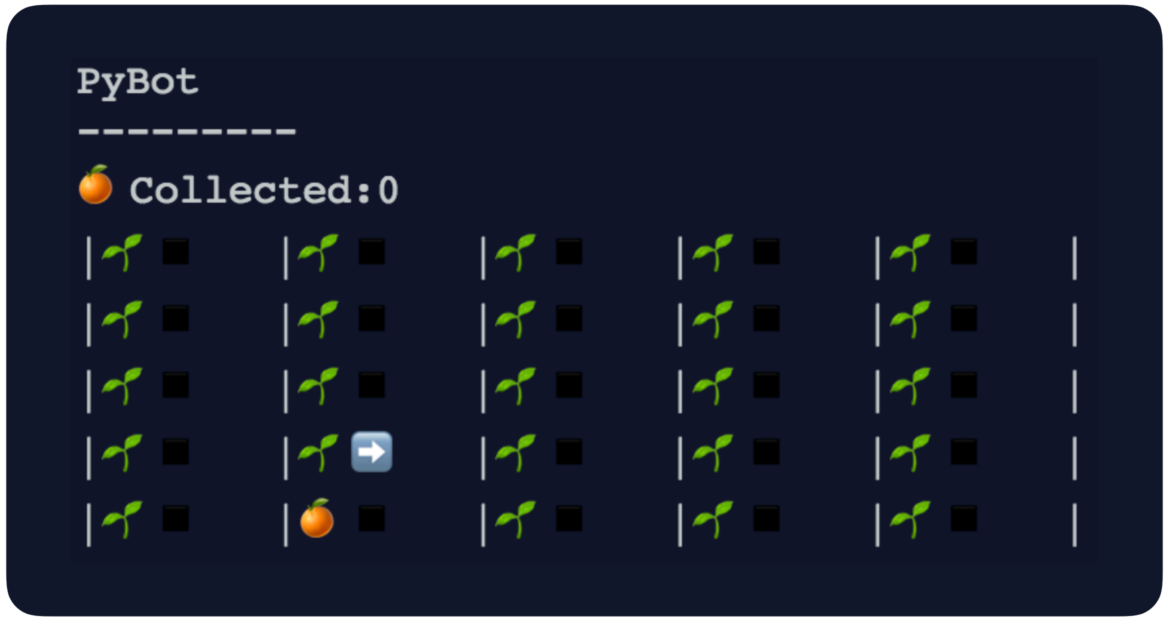


```
PyBot
----------
🍊 Collected:1
|🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |
|🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |
|🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |
|🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |
|🌱 ⬛    |🌱 ➡️    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |
```

AFTER

Turn Right!

PyBot rotates 90 degrees to the right, facing a new direction.



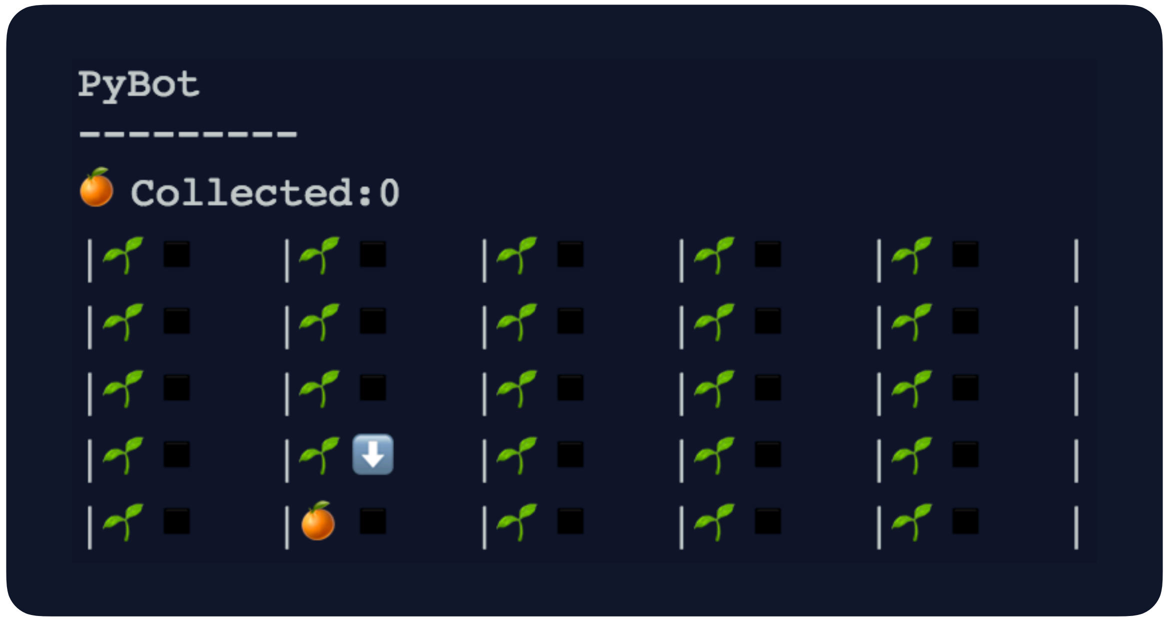
```
PyBot
----------
🍊 Collected:0
|🌱 ⬛     |🌱 ⬛     |🌱 ⬛     |🌱 ⬛     |🌱 ⬛     |
|🌱 ⬛     |🌱 ⬛     |🌱 ⬛     |🌱 ⬛     |🌱 ⬛     |
|🌱 ⬛     |🌱 ⬛     |🌱 ⬛     |🌱 ⬛     |🌱 ⬛     |
|🌱 ⬛     |🌱 ➡️     |🌱 ⬛     |🌱 ⬛     |🌱 ⬛     |
|🌱 ⬛     |🍊 ⬛     |🌱 ⬛     |🌱 ⬛     |🌱 ⬛     |
```

BEFORE

PyBot **Actions**

Turn Right!

PyBot rotates 90 degrees to the right, facing a new direction.



```
PyBot
---------
🍊 Collected:0
|🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |
|🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |
|🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |
|🌱 ⬛   |🌱 ⬇️   |🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |
|🌱 ⬛   |🍊 ⬛   |🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |
```

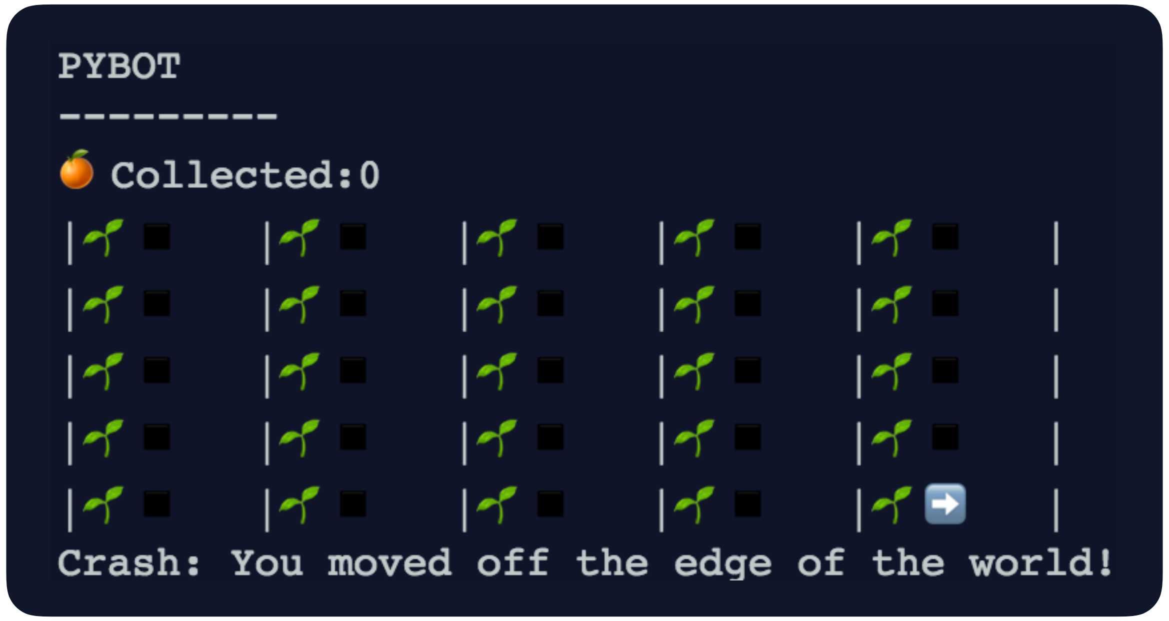
AFTER

PyBot **Actions**

PyBot can't turn left!

PyBot Crashes

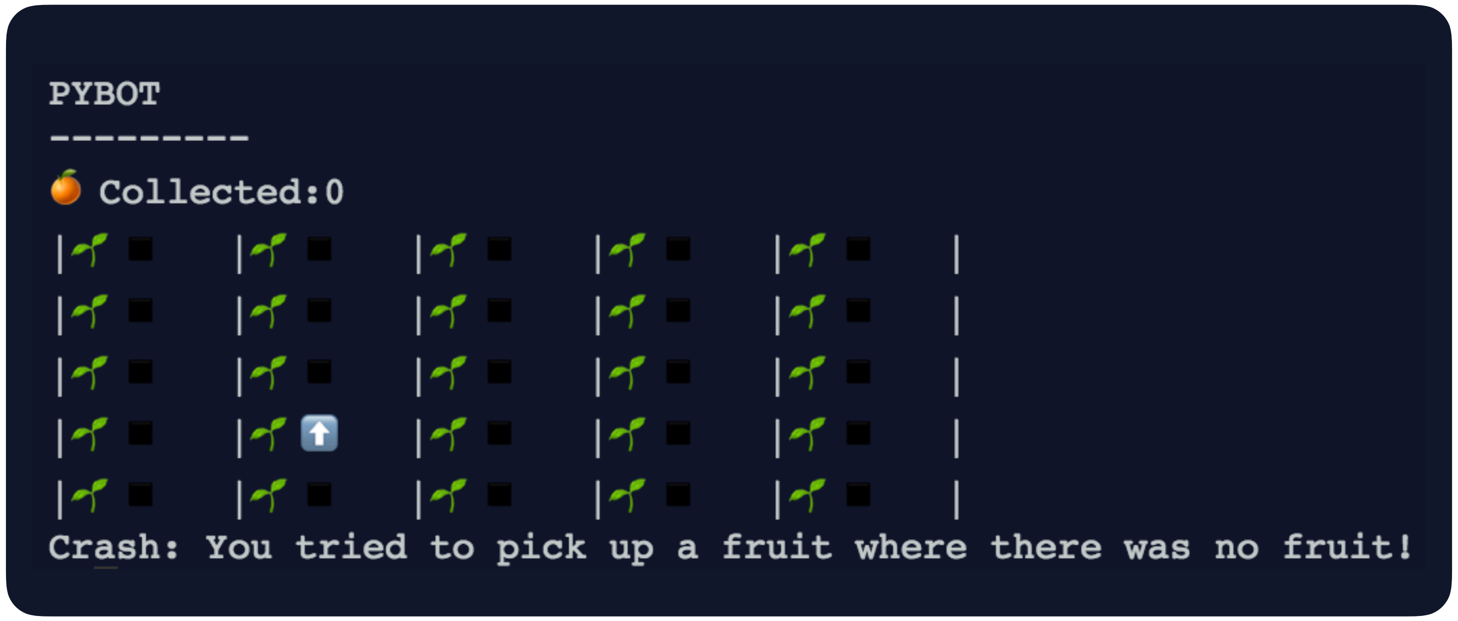
1) If PyBot moves off the edge of the board, **she crashes**



```
PYBOT
---------
🍊 Collected:0
|🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |🌱 ⬛    |
|🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |🌱 ⬛    |
|🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |🌱 ⬛    |
|🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |🌱 ⬛    |
|🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |🌱 ➡️    |
Crash: You moved off the edge of the world!
```

PyBot **Crashes**

2) If PyBot tries to pick a fruit where there is no fruit, **she crashes**



```
PYBOT
---------
🍊 Collected:0
|🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |
|🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |
|🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |
|🌱 ⬛    |🌱 ⬆️    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |
|🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |
Crash: You tried to pick up a fruit where there was no fruit!
```

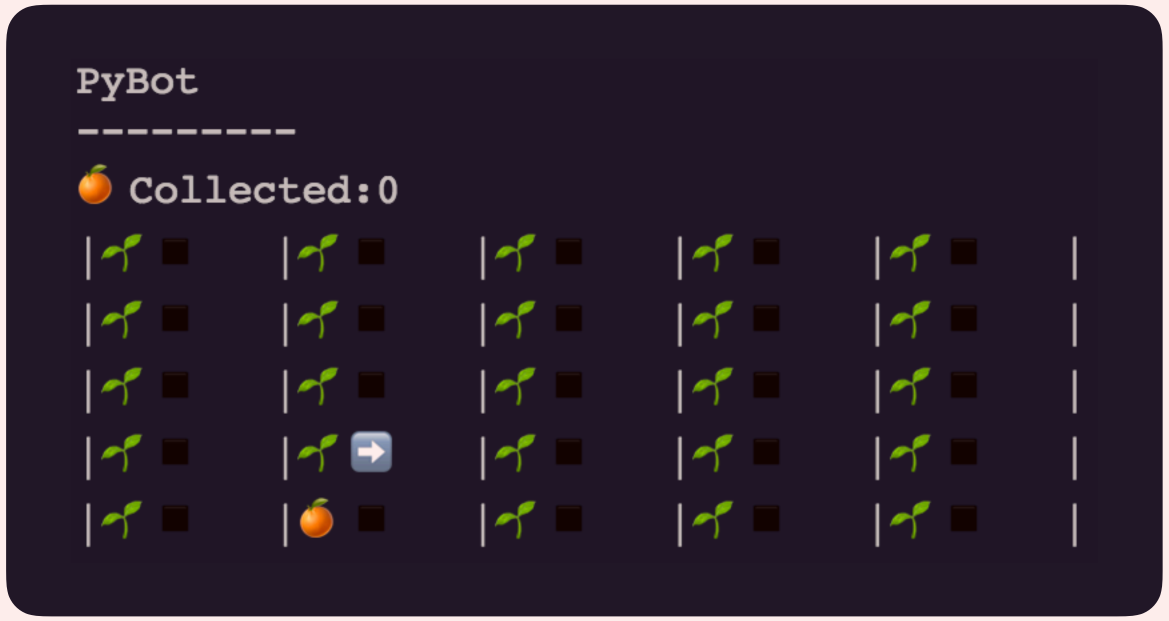
PyBot **Conditions**

True or False Questions you can ask **PyBot!**

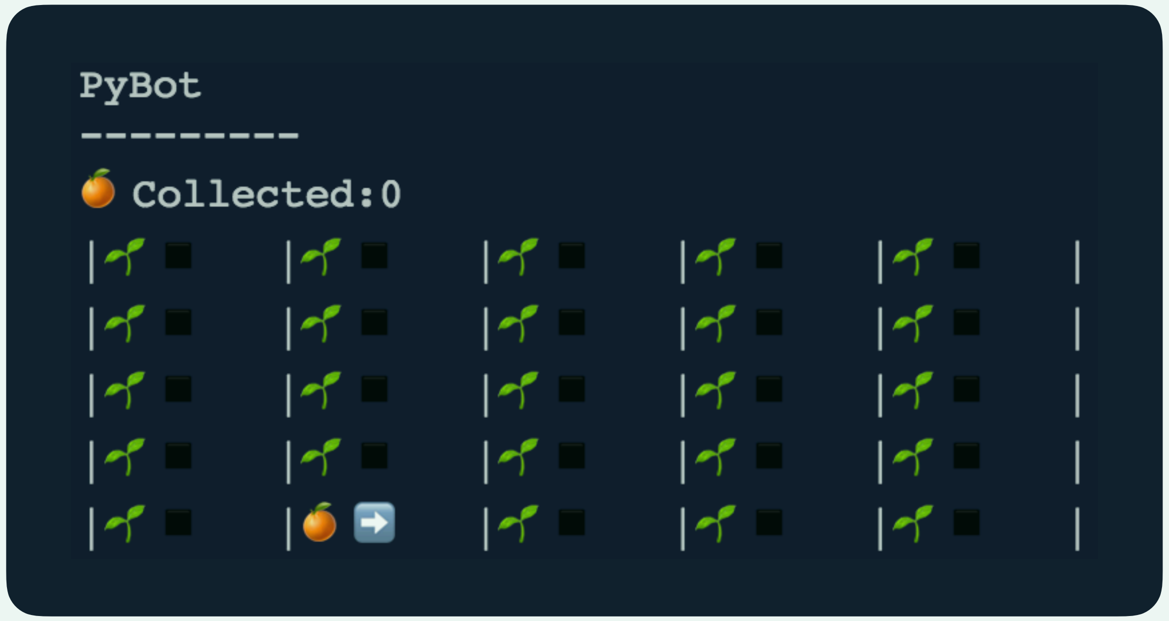
PyBot **Conditions**

Is there an orange?

Does PyBot's current cell have an orange in it?


```
PyBot
----------
🍊 Collected:0
|🌱 ■   |🌱 ■   |🌱 ■   |🌱 ■   |🌱 ■   |
|🌱 ■   |🌱 ■   |🌱 ■   |🌱 ■   |🌱 ■   |
|🌱 ■   |🌱 ■   |🌱 ■   |🌱 ■   |🌱 ■   |
|🌱 ■   |🌱 ➡️  |🌱 ■   |🌱 ■   |🌱 ■   |
|🌱 ■   |🍊 ■   |🌱 ■   |🌱 ■   |🌱 ■   |
```

FALSE

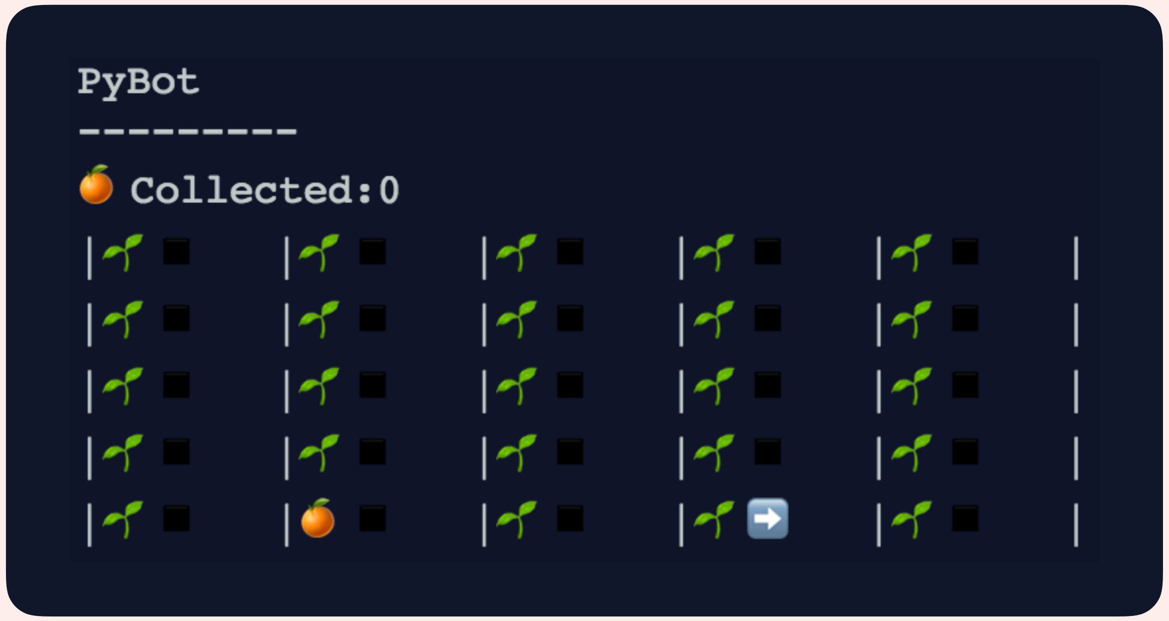

```
PyBot
----------
🍊 Collected:0
|🌱 ■   |🌱 ■   |🌱 ■   |🌱 ■   |🌱 ■   |
|🌱 ■   |🌱 ■   |🌱 ■   |🌱 ■   |🌱 ■   |
|🌱 ■   |🌱 ■   |🌱 ■   |🌱 ■   |🌱 ■   |
|🌱 ■   |🌱 ■   |🌱 ■   |🌱 ■   |🌱 ■   |
|🌱 ■   |🍊 ➡️  |🌱 ■   |🌱 ■   |🌱 ■   |
```

TRUE

PyBot **Conditions**

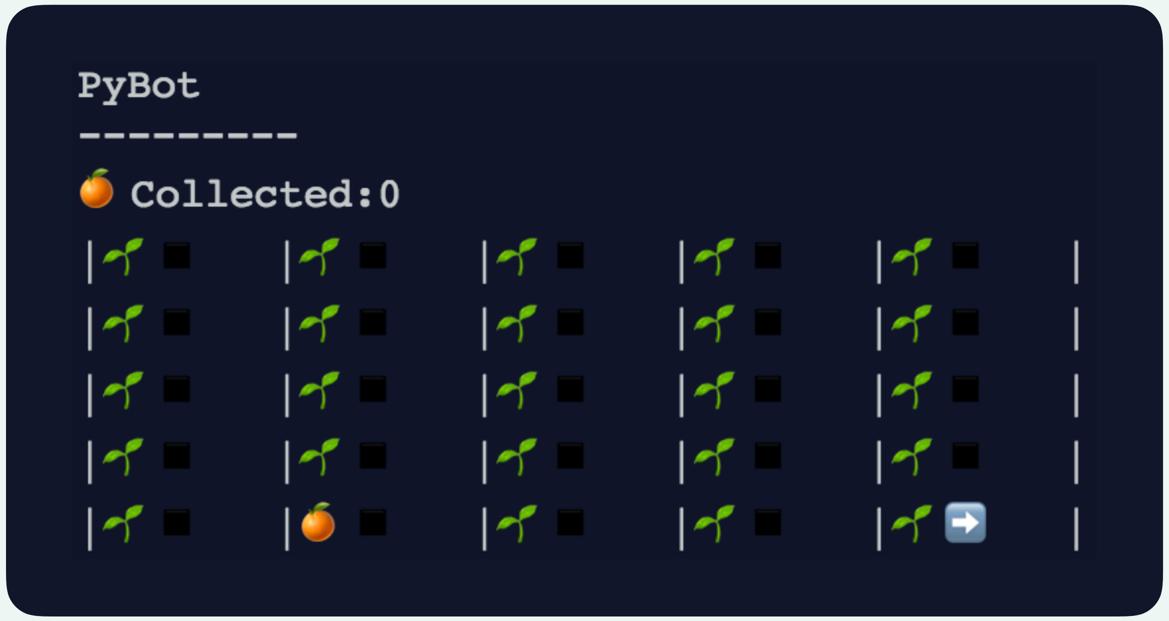
Facing a wall?

Is PyBot facing the edge of the field?



```
PyBot
----------
🍊 Collected:0
|🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |
|🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |
|🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |
|🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |
|🌱 ⬛    |🍊 ⬛    |🌱 ⬛    |🌱 ➡️    |🌱 ⬛    |
```

FALSE



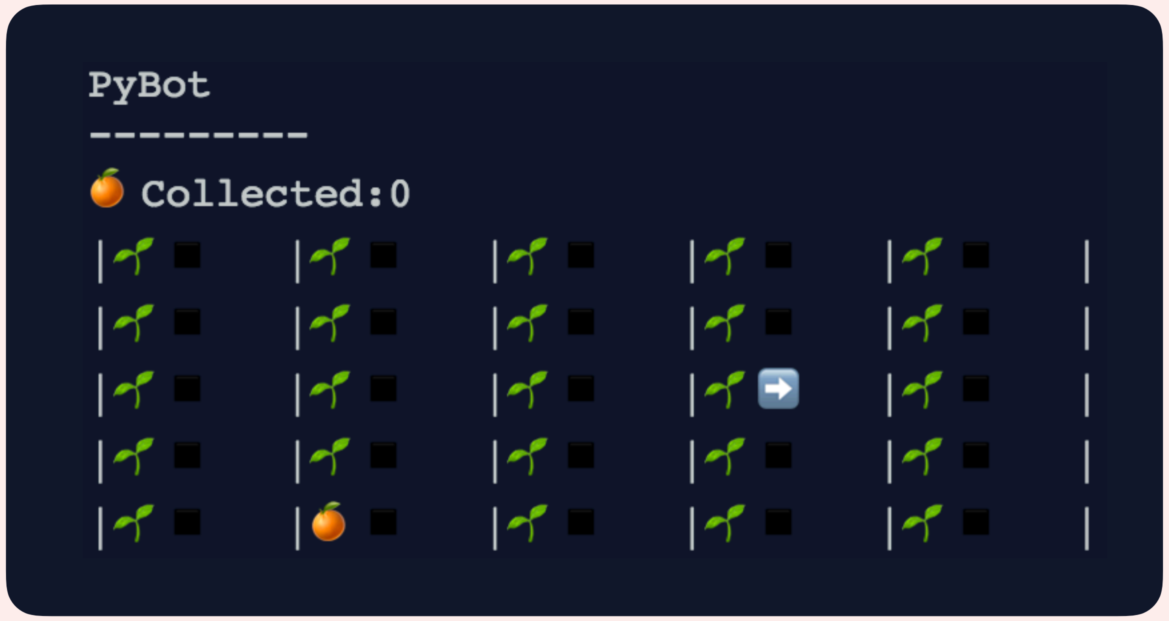
```
PyBot
----------
🍊 Collected:0
|🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |
|🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |
|🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |
|🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |
|🌱 ⬛    |🍊 ⬛    |🌱 ⬛    |🌱 ⬛    |🌱 ➡️    |
```

TRUE

PyBot **Conditions**

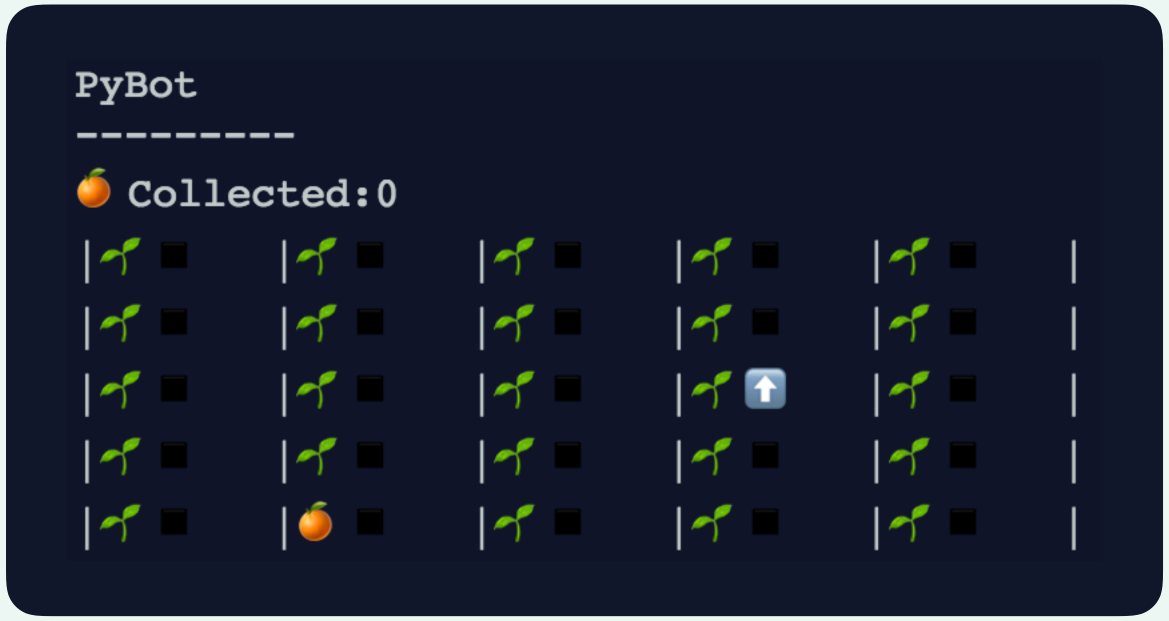
Facing north?

Is PyBot facing north ?



```
PyBot
---------
🍊 Collected:0
|🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |
|🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |
|🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |🌱 ➡️   |🌱 ⬛   |
|🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |
|🌱 ⬛   |🍊 ⬛   |🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |
```

FALSE



```
PyBot
---------
🍊 Collected:0
|🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |
|🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |
|🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |🌱 ⬆️   |🌱 ⬛   |
|🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |
|🌱 ⬛   |🍊 ⬛   |🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |
```

TRUE

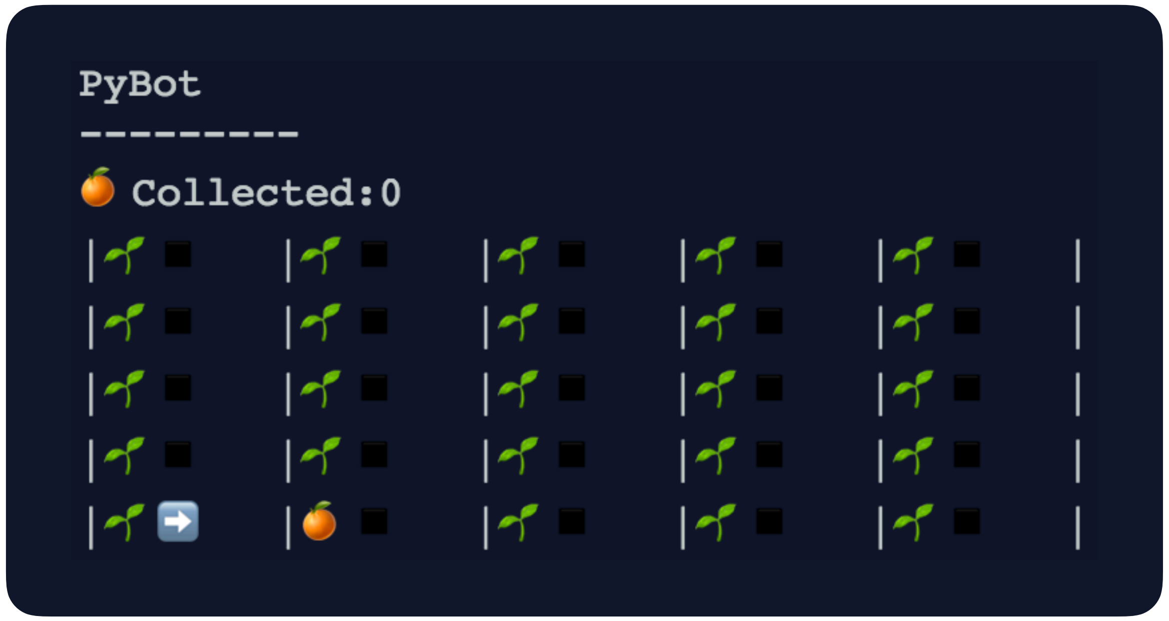
PyBot **Functions**

We can program PyBot using Python **functions**

PyBot **Functions**

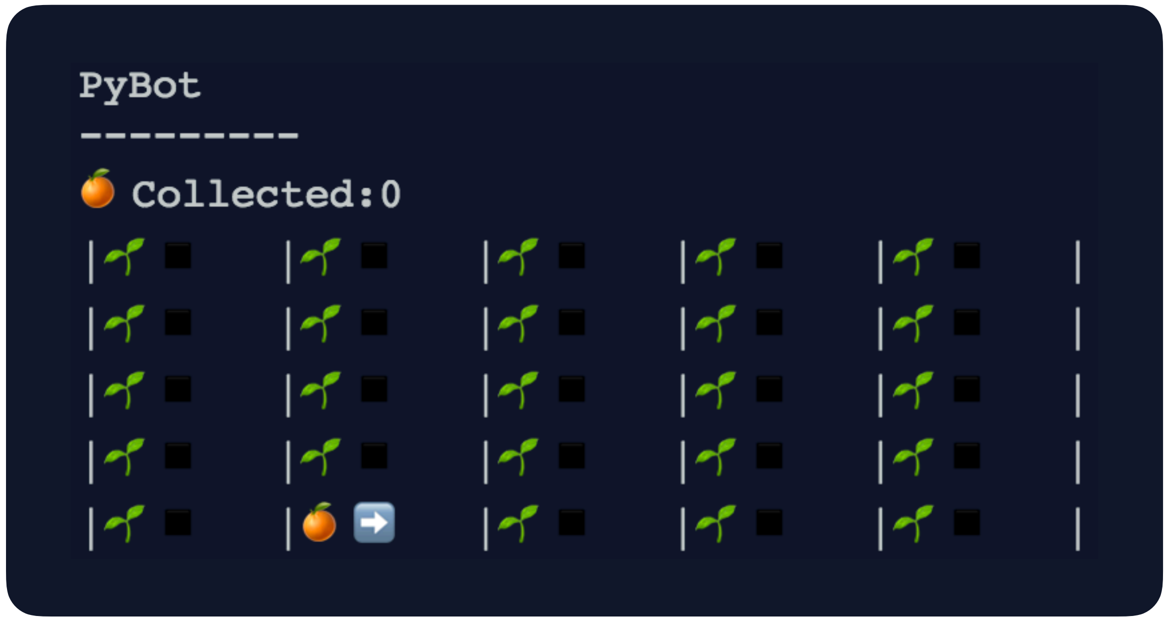
!

`move()` **PyBot moves forward one cell in the direction she is facing.**



```
PyBot
----------
🍊 Collected:0
|🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |
|🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |
|🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |
|🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |
|🌱 ➡️   |🍊 ⬛   |🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |
```

BEFORE



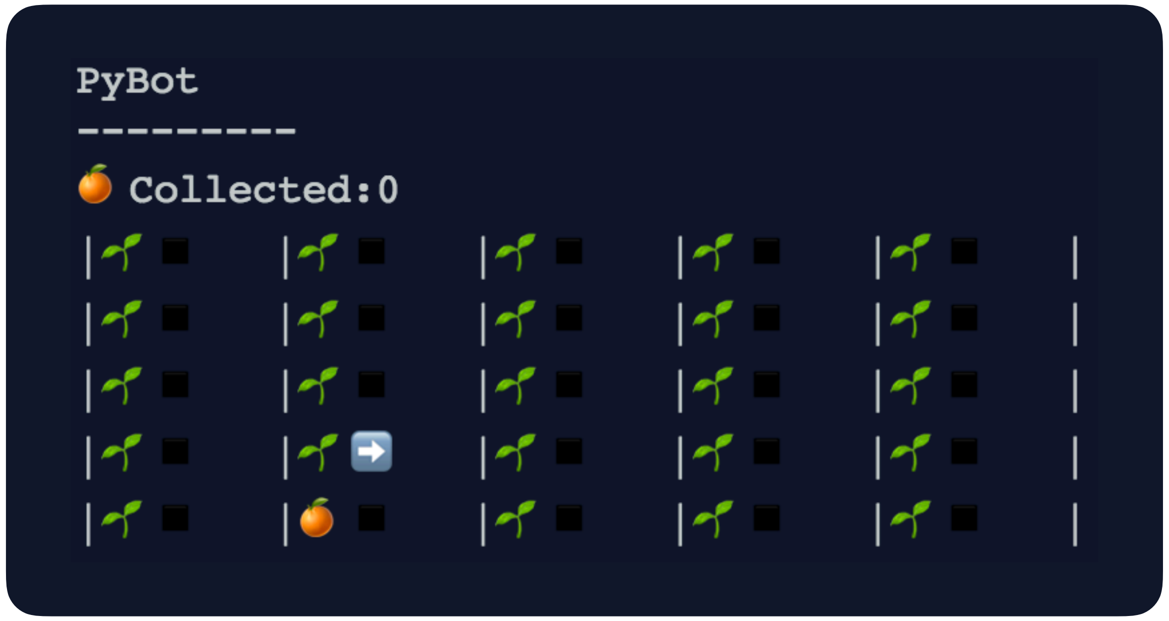
```
PyBot
----------
🍊 Collected:0
|🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |
|🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |
|🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |
|🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |
|🌱 ⬛   |🍊 ➡️   |🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |
```

AFTER

PyBot **Functions**

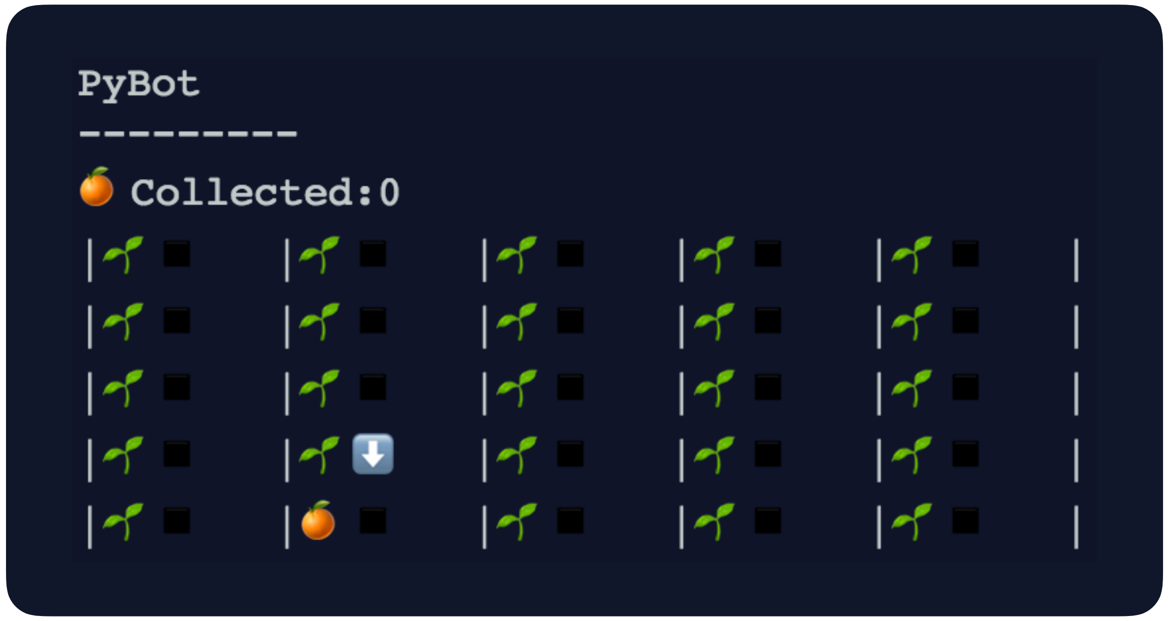
!

`turn_right()` **PyBot rotates 90 degrees to the right, facing a new direction.**



```
PyBot
---------
🍊 Collected:0
|🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |
|🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |
|🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |
|🌱 ⬛    |🌱 ➡️    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |
|🌱 ⬛    |🍊 ⬛    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |
```

BEFORE



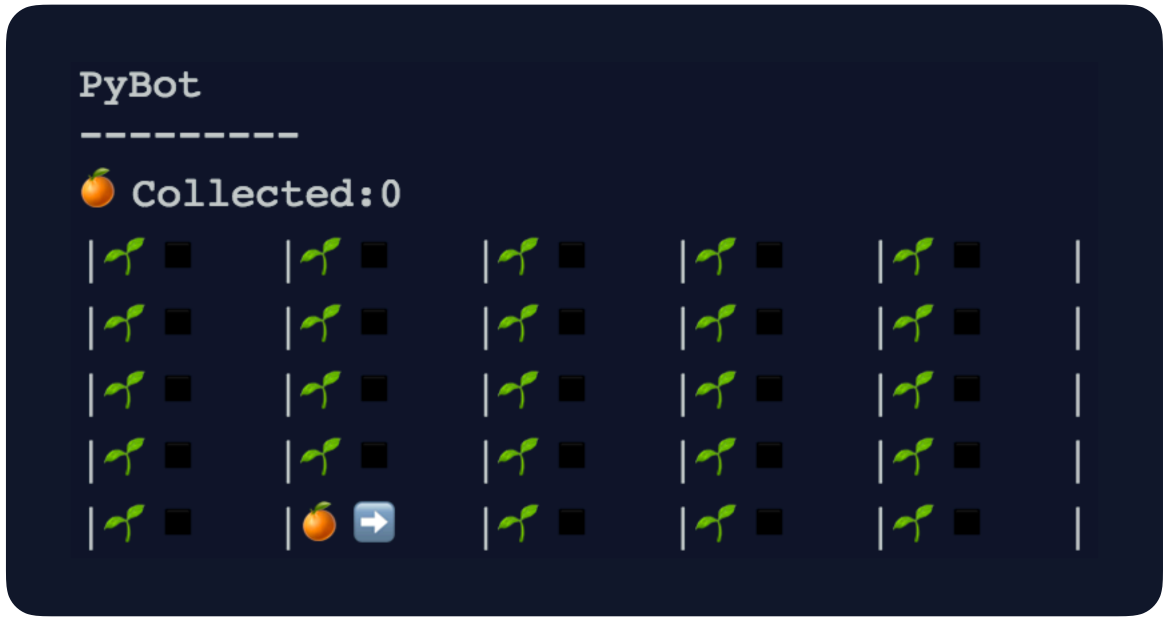
```
PyBot
---------
🍊 Collected:0
|🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |
|🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |
|🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |
|🌱 ⬛    |🌱 ⬇️    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |
|🌱 ⬛    |🍊 ⬛    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |
```

AFTER

PyBot **Functions**

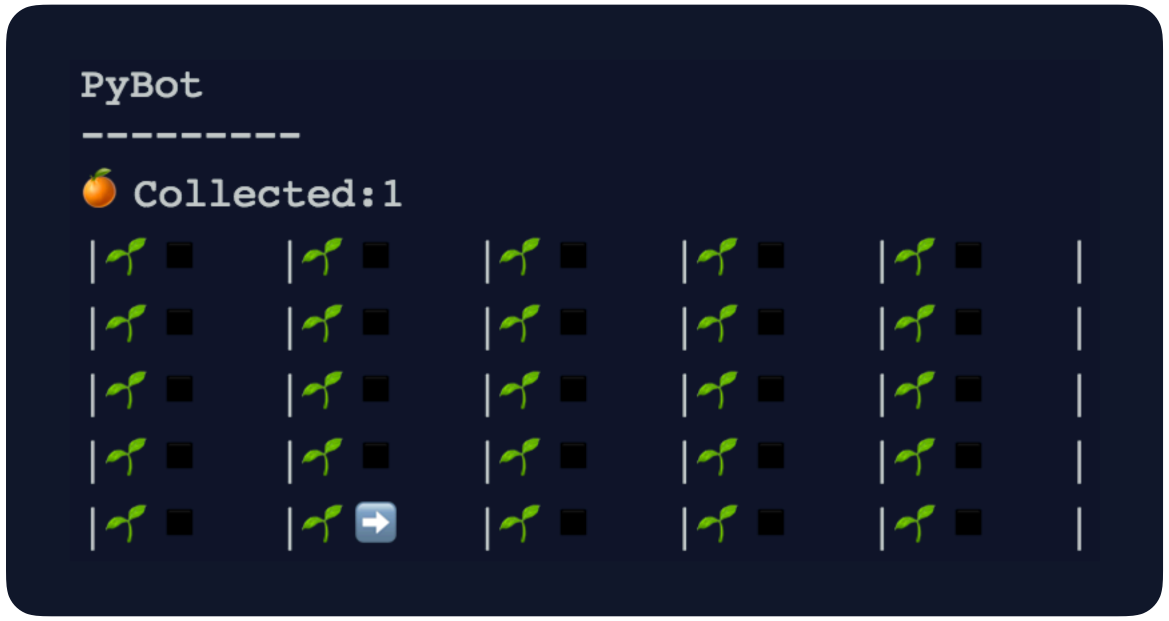
!

`pick_fruit()` **PyBot picks the fruit in the current cell.**



```
PyBot
----------
🍊 Collected:0
|🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |
|🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |
|🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |
|🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |
|🌱 ⬛    |🍊 ➡️    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |
```

BEFORE



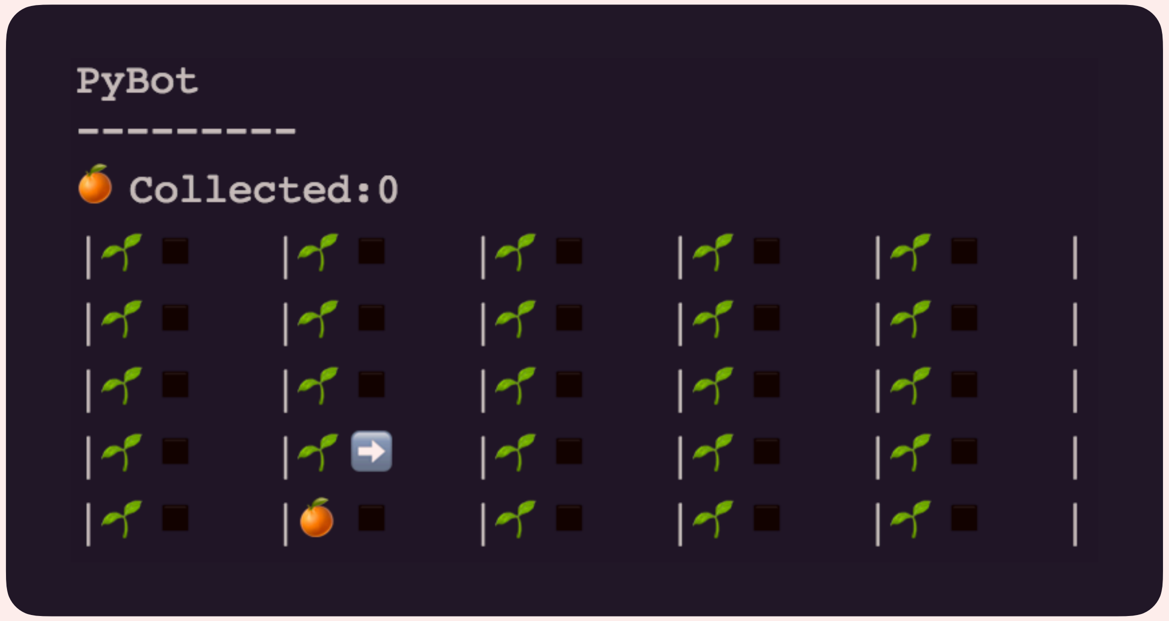
```
PyBot
----------
🍊 Collected:1
|🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |
|🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |
|🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |
|🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |
|🌱 ⬛    |🌱 ➡️    |🌱 ⬛    |🌱 ⬛    |🌱 ⬛    |
```

AFTER

PyBot **Functions**

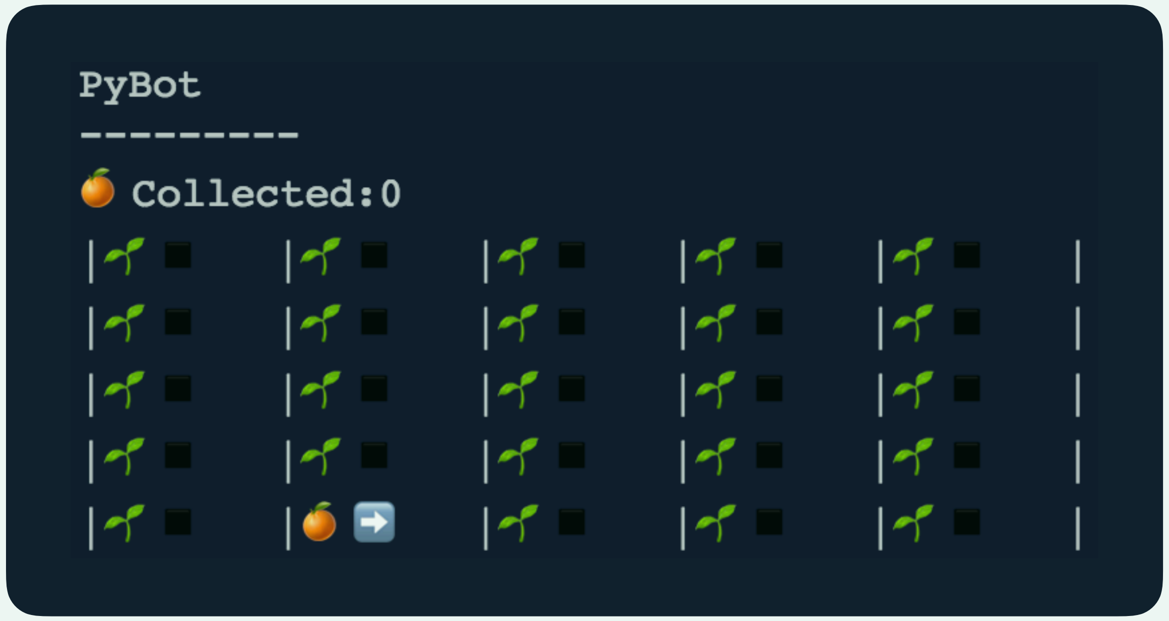
?

`has_fruit()` **Returns True if PyBot's current cell has an orange.**



```
PyBot
----------
🍊 Collected:0
|🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |
|🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |
|🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |
|🌱 ⬛   |🌱 ➡️   |🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |
|🌱 ⬛   |🍊 ⬛   |🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |
```

FALSE



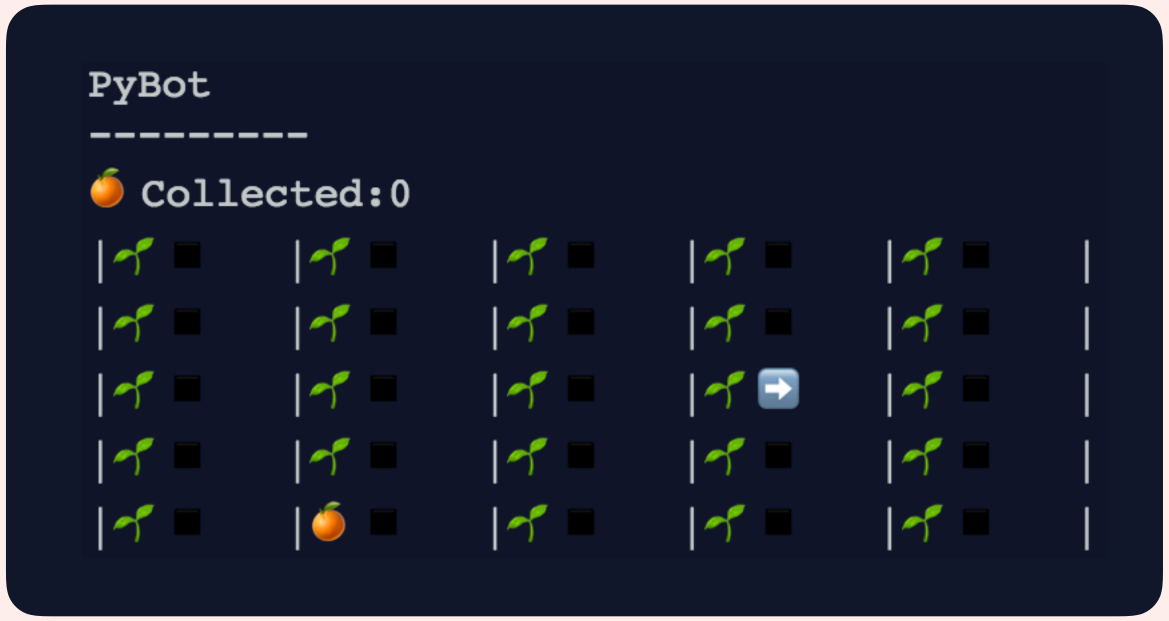
```
PyBot
----------
🍊 Collected:0
|🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |
|🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |
|🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |
|🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |
|🌱 ⬛   |🍊 ➡️   |🌱 ⬛   |🌱 ⬛   |🌱 ⬛   |
```

TRUE

PyBot **Functions**

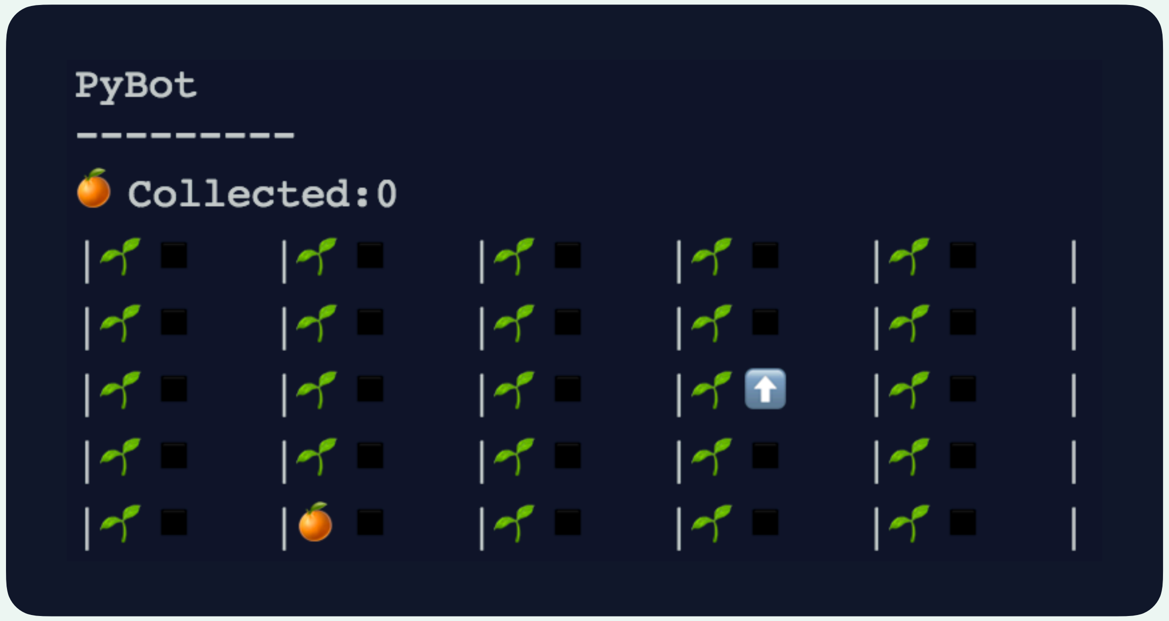
?

`is_facing_north()` Returns **True** if PyBot is facing north.



```
PyBot
---------
🍊 Collected:0
|🌱 ⬛  |🌱 ⬛  |🌱 ⬛  |🌱 ⬛  |🌱 ⬛  |
|🌱 ⬛  |🌱 ⬛  |🌱 ⬛  |🌱 ⬛  |🌱 ⬛  |
|🌱 ⬛  |🌱 ⬛  |🌱 ⬛  |🌱 ➡️  |🌱 ⬛  |
|🌱 ⬛  |🌱 ⬛  |🌱 ⬛  |🌱 ⬛  |🌱 ⬛  |
|🌱 ⬛  |🍊 ⬛  |🌱 ⬛  |🌱 ⬛  |🌱 ⬛  |
```

FALSE



```
PyBot
---------
🍊 Collected:0
|🌱 ⬛  |🌱 ⬛  |🌱 ⬛  |🌱 ⬛  |🌱 ⬛  |
|🌱 ⬛  |🌱 ⬛  |🌱 ⬛  |🌱 ⬛  |🌱 ⬛  |
|🌱 ⬛  |🌱 ⬛  |🌱 ⬛  |🌱 ⬆️  |🌱 ⬛  |
|🌱 ⬛  |🌱 ⬛  |🌱 ⬛  |🌱 ⬛  |🌱 ⬛  |
|🌱 ⬛  |🍊 ⬛  |🌱 ⬛  |🌱 ⬛  |🌱 ⬛  |
```

TRUE

PyBot **Functions**

Example: Harvest a Fruit

PyBot **Functions**

✓ Print Name

Harvest Your First Fruit

Turn Left

What are **functions?**

```
move()
turn_right()
pick_fruit()
has_fruit()
is_facing_north()
is_facing_east()
is_facing_south()
is_facing_west()
```

These are **functions**.

Definition

Function - *Code that is grouped together and packaged under a name, so it can be* ***executed*** *in one line.*

Definition

Execute - *To make the computer do something.*

Function **Structure**

The "**def**" keyword

```
def this_is_a_function():
    """
    This is an example function for the class notes.
    """
    if not front_is_blocked():
        move()
    turn_right()
    turn_right()
    move()
    move()
```

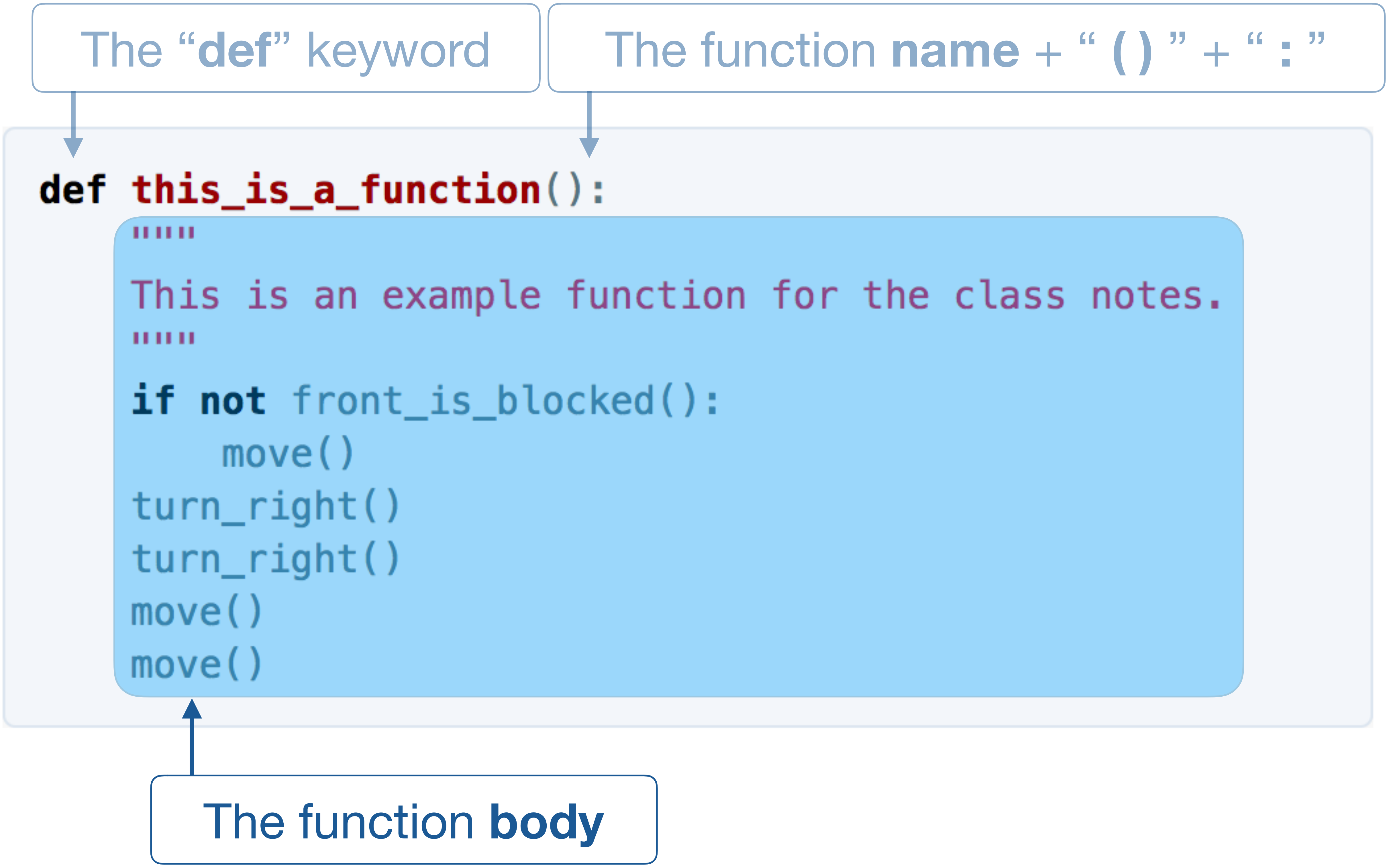
Function **Structure**

The "**def**" keyword

The function **name** + " **( )** " + " **:** "

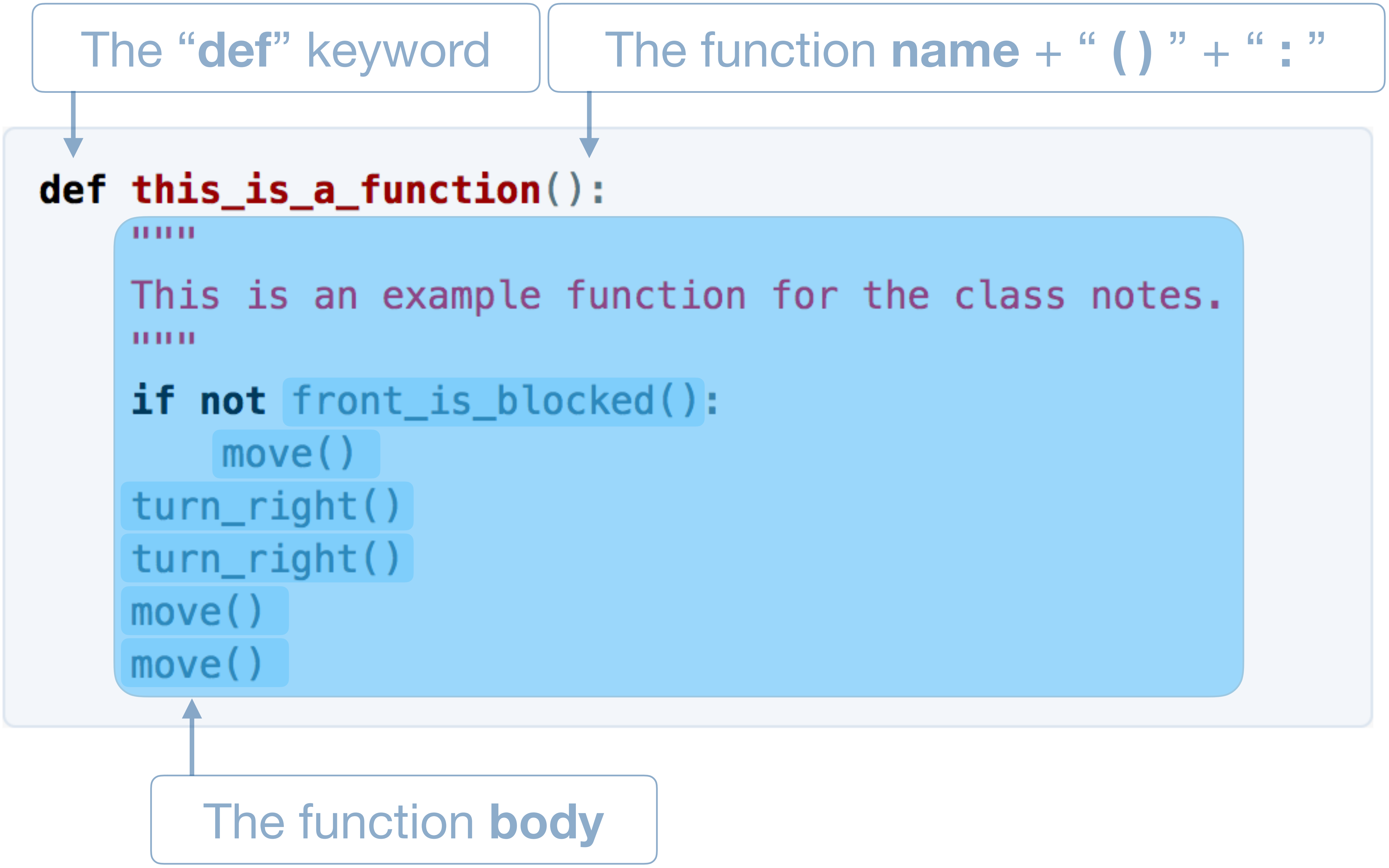
```
def this_is_a_function():
    """
    This is an example function for the class notes.
    """
    if not front_is_blocked():
        move()
    turn_right()
    turn_right()
    move()
    move()
```

Function **Structure**



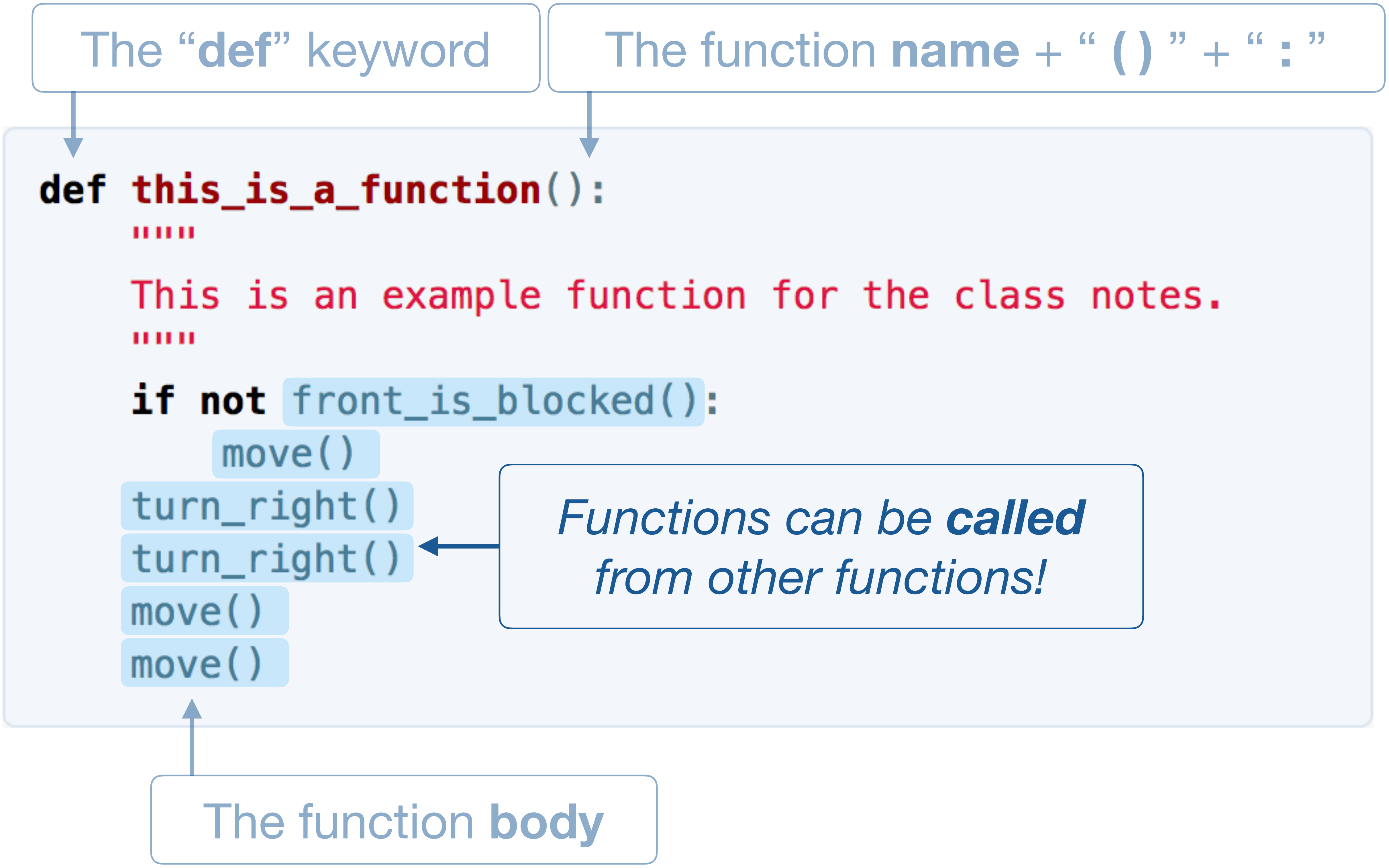
```
def this_is_a_function():
    """
    This is an example function for the class notes.
    """
    if not front_is_blocked():
        move()
    turn_right()
    turn_right()
    move()
    move()
```

Function **Structure**



```
def this_is_a_function():
    """
    This is an example function for the class notes.
    """
    if not front_is_blocked():
        move()
    turn_right()
    turn_right()
    move()
    move()
```

Function **Structure**


The "**def**" keyword
The function **name** + " **()** " + " **:** "
*Functions can be **called** from other functions!*
The function **body**

```
def this_is_a_function():
    """
    This is an example function for the class notes.
    """
    if not front_is_blocked():
        move()
    turn_right()
    turn_right()
    move()
    move()
```

Definition

Call - *To execute the code packaged within a function.*

Function **Structure**

| **"called"** `this_is_a_function()`

is the same thing as…

| **"executed"**

```
if not front_is_blocked():
    move()
turn_right()
turn_right()
move()
move()
```

Function **Implementation**

We **implement** functions in order to **decompose** our code.

Definition

Implement - To ~write~ code! The word for a specific instance of written code is “*implementation*.”

Definition

Decompose - To break apart code into small, reusable pieces.

Function **Implementation**

```python
def main():
    pick_fruit()
    move()
    pick_fruit()
    move()
    pick_fruit()
    move()
    pick_fruit()
    move()
    turn_right()
    turn_right()
    turn_right()
    pick_fruit()
    move()
    pick_fruit()
    move()
    pick_fruit()
    move()
    pick_fruit()
    move()
    turn_right()
    turn_right()
    turn_right()

if __name__ == '__main__':
    main()
```

```python
def turn_left():
    turn_right()
    turn_right()
    turn_right()

def pick_and_move():
    pick_fruit()
    move()

def pick_fruit_across():
    pick_and_move()
    pick_and_move()
    pick_and_move()
    pick_and_move()

def main():
    pick_fruit_across()
    turn_left()
    pick_fruit_across()
    turn_left()

if __name__ == '__main__':
    main()
```

Function **Implementation**

```
def main():
    pick_fruit()
    move()
    pick_fruit()
    move()
    pick_fruit()
    move()
    pick_fruit()
    move()
    turn_right()
    turn_right()
    turn_right()
    pick_fruit()
    move()
    pick_fruit()
    move()
    pick_fruit()
    move()
    pick_fruit()
    move()
    turn_right()
    turn_right()
    turn_right()

if __name__ == '__main__':
    main()
```

```
def turn_left():
    turn_right()
    turn_right()
    turn_right()

def pick_and_move():
    pick_fruit()
    move()

def pick_fruit_across():
    pick_and_move()
    pick_and_move()
    pick_and_move()
    pick_and_move()

def main():
    pick_fruit_across()
    turn_left()
    pick_fruit_across()
    turn_left()

if __name__ == '__main__':
    main()
```

These code segments do the *same thing*.

Function **Implementation**

```python
def main():
    pick_fruit()
    move()
    pick_fruit()
    move()
    pick_fruit()
    move()
    pick_fruit()
    move()
    turn_right()
    turn_right()
    turn_right()
    pick_fruit()
    move()
    pick_fruit()
    move()
    pick_fruit()
    move()
    pick_fruit()
    move()
    turn_right()
    turn_right()
    turn_right()

if __name__ == '__main__':
    main()
```

```python
def turn_left():
    turn_right()
    turn_right()
    turn_right()

def pick_and_move():
    pick_fruit()
    move()

def pick_fruit_across():
    pick_and_move()
    pick_and_move()
    pick_and_move()
    pick_and_move()

def main():
    pick_fruit_across()
    turn_left()
    pick_fruit_across()
    turn_left()

if __name__ == '__main__':
    main()
```

These code segments do the *same thing*.

The one on the right is well *decomposed*.

Function **Implementation**

```python
def main():
    pick_fruit()
    move()
    pick_fruit()
    move()
    pick_fruit()
    move()
    pick_fruit()
    move()
    turn_right()
    turn_right()
    turn_right()
    pick_fruit()
    move()
    pick_fruit()
    move()
    pick_fruit()
    move()
    pick_fruit()
    move()
    turn_right()
    turn_right()
    turn_right()

if __name__ == '__main__':
    main()
```

```python
def turn_left():
    turn_right()
    turn_right()
    turn_right()

def pick_and_move():
    pick_fruit()
    move()

def pick_fruit_across():
    pick_and_move()
    pick_and_move()
    pick_and_move()
    pick_and_move()

def main():
    pick_fruit_across()
    turn_left()
    pick_fruit_across()
    turn_left()

if __name__ == '__main__':
    main()
```

These code segments do the *same thing*.

The one on the right is well *decomposed*.

It is not only *shorter*, but also *easier to read*.

Function **Implementation**

```
def main():
    pick_fruit()
    move()
    pick_fruit()
    move()
    pick_fruit()
    move()
    pick_fruit()
    move()
    turn_right()
    turn_right()
    turn_right()
    pick_fruit()
    move()
    pick_fruit()
    move()
    pick_fruit()
    move()
    pick_fruit()
    move()
    turn_right()
    turn_right()
    turn_right()

if __name__ == '__main__':
    main()
```

```
def turn_left():
    turn_right()
    turn_right()
    turn_right()

def pick_and_move():
    pick_fruit()
    move()

def pick_fruit_across():
    pick_and_move()
    pick_and_move()
    pick_and_move()
    pick_and_move()

def main():
    pick_fruit_across()
    turn_left()
    pick_fruit_across()
    turn_left()

if __name__ == '__main__':
    main()
```

These code segments do the *same thing*.

The one on the right is well *decomposed*.

It is not only *shorter*, but also *easier to read*.

It is also easier to *fix*.

Function **Implementation**

```
def main():
    pick_fruit()
    move()
    pick_fruit()
    move()
    pick_fruit()
    move()
    pick_fruit()
    move()
    turn_right()
    turn_right()
    turn_right()
    pick_fruit()
    move()
    pick_fruit()
    move()
    pick_fruit()
    move()
    pick_fruit()
    move()
    turn_right()
    turn_right()
    turn_right()

if __name__ == '__main__':
    main()
```

```
def turn_left():
    turn_right()
    turn_right()
    turn_right()

def pick_and_move():
    pick_fruit()
    move()

def pick_fruit_across():
    pick_and_move()
    pick_and_move()
    pick_and_move()
    pick_and_move()

def main():
    pick_fruit_across()
    turn_left()
    pick_fruit_across()
    turn_left()

if __name__ == '__main__':
    main()
```

These code segments do the *same thing*.

The one on the right is well *decomposed*.

It is not only *shorter*, but also *easier to read*.

It is also easier to *fix*.

Function **Implementation**

What if we wanted to change `pick_fruit()` to `pick_vegetable()` ?

Function **Implementation**

8 lines :(

```
def main():
    pick_fruit()
    move()
    pick_fruit()
    move()
    pick_fruit()
    move()
    pick_fruit()
    move()
    turn_right()
    turn_right()
    turn_right()
    pick_fruit()
    move()
    pick_fruit()
    move()
    pick_fruit()
    move()
    pick_fruit()
    move()
    turn_right()
    turn_right()
    turn_right()

if __name__ == '__main__':
    main()
```

```
def turn_left():
    turn_right()
    turn_right()
    turn_right()

def pick_and_move():
    pick_fruit()
    move()

def pick_fruit_across():
    pick_and_move()
    pick_and_move()
    pick_and_move()
    pick_and_move()

def main():
    pick_fruit_across()
    turn_left()
    pick_fruit_across()
    turn_left()

if __name__ == '__main__':
    main()
```

1 line :D

Function **Structure**

Example: Pick and Move Function

Function **Structure**

✓ **Print Name**

✓ **Harvest Your First Fruit**

Turn Left

Function **Implementation**

Let's start working on this week's project!

Project: Introducing You

Recap

repl.it = Where we will be coding.

PyBot = Your new best friend. Learn her set of commands!

Functions are little packages of code.

Implement functions to *decompose* and *make your life easier.*