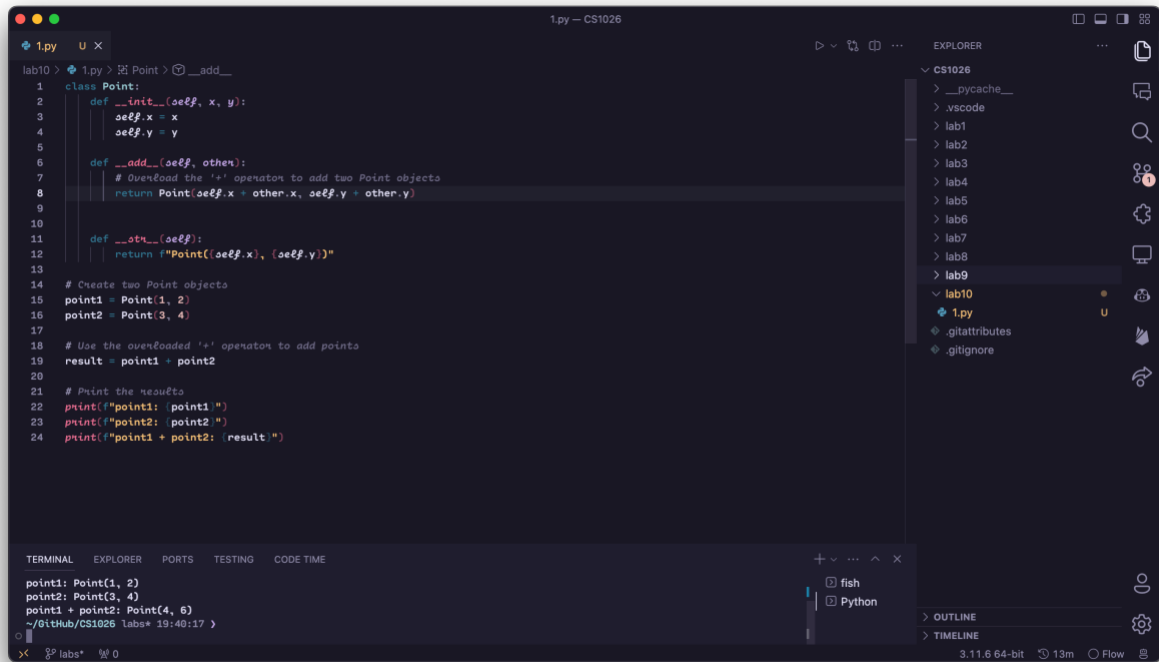


1. iPhone
2. Guido van Rossum
3. Google, microsoft, amazon
4. Python



The screenshot shows a VS Code editor window with a Python file named `1.py`. The code defines a `Point` class with `__init__` and `__add__` methods. It then creates two `Point` objects, `point1` and `point2`, and uses the `+` operator to add them, storing the result in `result`. The code is as follows:

```
1 class Point:
2     def __init__(self, x, y):
3         self.x = x
4         self.y = y
5
6     def __add__(self, other):
7         # Overload the '+' operator to add two Point objects
8         return Point(self.x + other.x, self.y + other.y)
9
10
11     def __str__(self):
12         return f"Point({self.x}, {self.y})"
13
14 # Create two Point objects
15 point1 = Point(1, 2)
16 point2 = Point(3, 4)
17
18 # Use the overloaded '+' operator to add points
19 result = point1 + point2
20
21 # Print the results
22 print(f"point1: {point1}")
23 print(f"point2: {point2}")
24 print(f"point1 + point2: {result}")
```

The terminal at the bottom shows the output of the program:

```
point1: Point(1, 2)
point2: Point(3, 4)
point1 + point2: Point(4, 6)
~/GitHub/CS1026 labs* 19:40:17
```

Program output in terminal at bottom of screenshot

The screenshot shows a VS Code editor window with a Python file named `2.py` open. The code defines a `Fan` class with attributes `OFF` and `ON`, and methods `__init__`, `turn_on`, `turn_off`, and `get_stateValue`. The `__init__` method sets the initial state to `OFF`. The `turn_on` method sets the state to `ON`, and the `turn_off` method sets the state to `OFF`. The `get_stateValue` method returns the current state as a string. The code also creates an instance of the `Fan` class, `fan1`, and prints its state after calling `turn_on` and `turn_off`.

```
1 class Fan:
2     OFF = 0
3     ON = 1
4     _state = OFF
5
6     def __init__(self):
7         # Initial state
8         Fan._state = Fan.OFF
9
10    def turn_on(self):
11        Fan._state = Fan.ON
12
13    def turn_off(self):
14        Fan._state = Fan.OFF
15
16    def get_stateValue(self):
17        state = str(Fan._state)
18        if state == "0":
19            return "OFF"
20        else:
21            return "ON"
22
23    fan1 = Fan()
24    fan1.turn_on()
25    print("Fan is", fan1.get_stateValue())
26    fan1.turn_off()
27    print("Fan is", fan1.get_stateValue())
```

The terminal output at the bottom of the screenshot shows the execution of the program:

```
~/opt/homebrew/bin/python3 /Users/rohin/GitHub/CS1026/Lab10/2.py
Fan is ON
Fan is OFF
~/GitHub/CS1026 Labs* 19:42:08
```

Program output in terminal at bottom of screenshot

1000 base 2 is 10 base 10

11011 base 2 is 27 base 10