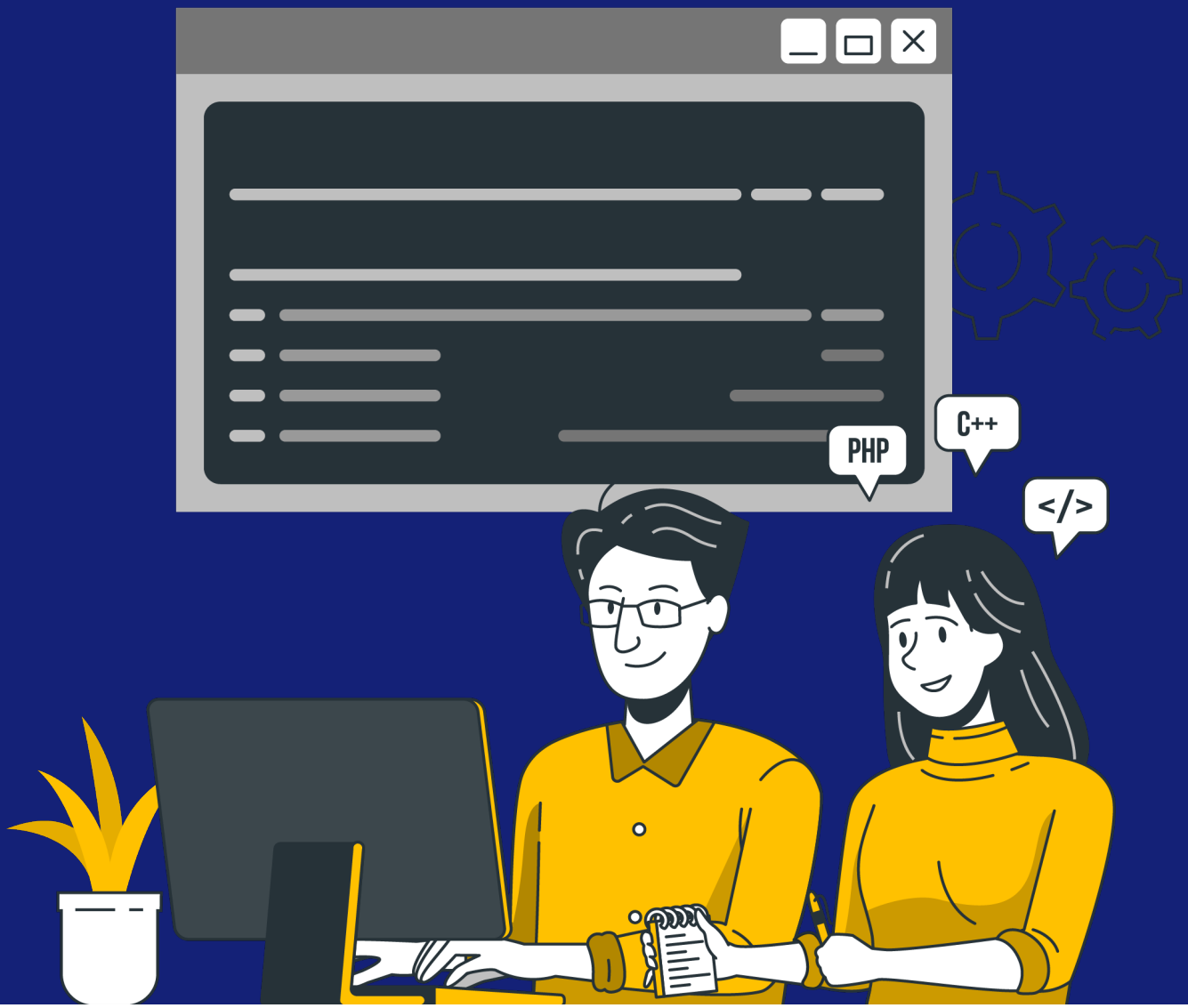
PW SKILLS

Lesson Plan

Dunder Methods



Dunder methods, also known as magic methods or special methods, in Python are special reserved methods that are surrounded by double underscores (i.e., __method__). These methods allow you to define how instances of your classes behave when they are used with built-in Python functions or operators. Understanding dunder methods is crucial for creating custom objects that behave like built-in types or implementing operator overloading in Python. Here's a detailed explanation of some commonly used dunder methods:

1. **__init__(self, ...)**: This method is called when an instance of the class is initialized. It is used to initialize instance variables and perform any setup required for the object.

```
class MyClass:
    def __init__(self, x):
        self.x = x

obj = MyClass(5)
```

2. **__str__(self):** This method is called when the str() function is used on an instance of the class. It should return a string representation of the object.

```
class MyClass:
    def __init__(self, x):
        self.x = x

    def __str__(self):
        return f'MyClass instance with x = {self.x}'

obj = MyClass(5)
print(str(obj))
```

Output:

```
MyClass instance with x = 5
```

3. **__repr__(self):** This method is called when the repr() function is used on an instance of the class. It should return an unambiguous string representation of the object, which can be used to recreate the object.

```
class MyClass:
    def __init__(self, x):
        self.x = x

    def __repr__(self):
        return f'MyClass({self.x})'

obj = MyClass(5)
print(repr(obj))  # Output: MyClass(5)
```

4. **__add__(self, other):** This method is called when the + operator is used with instances of the class. It should return the result of addition.

```
class Point:
    def __init__(self, x, y):
        self.x = x
        self.y = y

    def __add__(self, other):
        return Point(self.x + other.x, self.y + other.y)

p1 = Point(1, 2)
p2 = Point(3, 4)
p3 = p1 + p2
print(p3.x, p3.y)
```

Output:

```
4 6
```

5. **__eq__(self, other):** This method is called when the == operator is used with instances of the class. It should return True if the objects are considered equal, False otherwise.

```
class Point:
    def __init__(self, x, y):
        self.x = x
        self.y = y

    def __eq__(self, other):
        return self.x == other.x and self.y == other.y

p1 = Point(1, 2)
p2 = Point(1, 2)
print(p1 == p2)
```

Output:

```
True
```

These are just a few examples of dunder methods in Python. There are many more dunder methods available for various purposes, such as arithmetic operations, container behavior, context management, and more.

PW SKILLS

THANK YOU !