

Name: Aj Kean Edoria

Programming Constructs and Paradigms

This notebook explains different programming paradigms with examples.

Procedural Programming

```
In [1]: # Procedural Programming (PP)
print("Procedural Programming Example")
def add_numbers(a, b):
    return a + b
print("Sum:", add_numbers(5, 3))
```

Procedural Programming Example
Sum: 8

Problem Exercise: Write a function to calculate the factorial of a given number using procedural programming.

```
In [2]: def factorial(n):
        result = 1
        for i in range(1, n + 1):
            result *= i
        return result
print("Factorial:", factorial(5))
```

Factorial: 120

Functional Programming

```
In [3]: # Functional Programming (FP)
print("Functional Programming Example")
add = lambda x, y: x + y
print("Sum:", add(5, 3))
```

Functional Programming Example
Sum: 8

Problem Exercise: Implement a function that returns a list of squares of numbers using functional programming.

```
In [4]: numbers = [1, 2, 3, 4, 5]
squares = list(map(lambda x: x ** 2, numbers))
print("Squares:", squares)
```

Squares: [1, 4, 9, 16, 25]

Object-Oriented Programming (OOP)

```
In [5]: # Object-Oriented Programming (OOP)
print("Object-Oriented Programming Example")
class Calculator:
    def add(self, x, y):
        return x + y
calc = Calculator()
print("Sum:", calc.add(5, 3))
```

Object-Oriented Programming Example

Sum: 8

Problem Exercise: Create a class `Rectangle` with methods to calculate area and perimeter.

```
In [6]: class Rectangle:
    def __init__(self, length, width):
        self.length = length
        self.width = width

    def area(self):
        return self.length * self.width

    def perimeter(self):
        return 2 * (self.length + self.width)
rect = Rectangle(5, 3)
print("Area:", rect.area())
print("Perimeter:", rect.perimeter())
```

Area: 15

Perimeter: 16

Conclusion

Each paradigm has unique strengths. Combining them often leads to better software design.