

Plataformas e Serviços X-Ops - 2024/25

1. Analise os seguintes trechos de código em Python e identifique eventuais *code smells* existentes.
2. Com base nas respostas do ponto 1., reestruture e otimize o código.

A)

```
def process_data(data):  
    # This method does too much  
    result = []  
    for item in data:  
        item['processed'] = True  
        if 'value' in item:  
            item['value'] *= 2  
        if 'name' in item:  
            item['name'] = item['name'].upper()  
        item['timestamp'] = datetime.now()  
        result.append(item)  
    save_to_database(result)  
    send_notification("Processing complete")  
    return result
```

B)

```
class Rectangle:  
    def __init__(self, width, height):  
        self.width = width  
        self.height = height  
  
    def area(self):  
        return self.width * self.height  
  
    def perimeter(self):  
        return 2 * (self.width + self.height)
```

C)

```
def connect_to_server(ip, port, username, password):  
    print(f"Connecting to {ip}:{port} with user {username}")  
    # Connection logic here
```

D)

```
def CalculateArea():  
    pass  
  
def calc_perimeter():  
    pass  
  
def CalcVolume():  
    pass
```

E)

```
class GodClass:  
    def __init__(self):  
        self.users = []  
        self.orders = []  
        self.products = []  
  
    def add_user(self, user):  
        self.users.append(user)  
  
    def add_order(self, order):  
        self.orders.append(order)  
  
    def add_product(self, product):  
        self.products.append(product)  
  
    def get_users(self):  
        return self.users  
  
    def get_orders(self):  
        return self.orders
```

F)

```
class Order:  
    def __init__(self, customer, items):  
        self.customer = customer  
        self.items = items  
  
    def print_customer_address(self):  
        print(f"Customer address: {self.customer.street}, {self.customer.city}")  
  
class Customer:  
    def __init__(self, name, street, city):  
        self.name = name  
        self.street = street  
        self.city = city
```

G)

```
def get_animal_sound(animal):  
    if animal == "Dog":  
        return "Bark"  
    elif animal == "Cat":  
        return "Meow"  
    elif animal == "Cow":  
        return "Moo"  
    else:  
        return "Unknown sound"
```

H)

```
class Order:
    def __init__(self, customer):
        self.customer = customer

    def get_customer_address(self):
        return self.customer.get_address()

class Customer:
    def __init__(self, address):
        self.address = address

    def get_address(self):
        return self.address
```

I)

```
class Library:
    def __init__(self, books):
        self.books = books

    def find_book_author(self, title):
        return self.books.get(title).get_author().get_name().get_initials()

class Book:
    def __init__(self, title, author):
        self.title = title
        self.author = author

    def get_author(self):
        return self.author

class Author:
    def __init__(self, name):
        self.name = name

    def get_name(self):
        return self.name

class Name:
    def __init__(self, initials):
        self.initials = initials

    def get_initials(self):
        return self.initials
```