PYTHON OOP (NESNE YÖNELİMLİ PROGRAMLAMA) ÖDEVİ

main.py

```
from product import Product
from customer import Customer
from cart import Cart
from order import Order

def main():
    # Ürünleri oluştur
    products = [
        Product("Laptop", 15000, 5),
        Product("Telefon", 10000, 15),
        Product("Kulaklık", 500, 150)
    ]

    # Müşteri bilgilerini al
    name = input("Adınızı girin: ")
    email = input("Email adresinizi girin: ")
    customer = Customer(name, email)

    # Sepet oluştur
    cart = Cart()

    while True:
        print("\nÜrün Listesi:")
        for i, product in enumerate(products, start=1):
            print(f"{i}. {product}")

        choice = input("\nSatın almak istediğiniz ürünün numarasını girin (Çıkış için 'q'): ")
        if choice.lower() == 'q':
            break

        try:
            choice = int(choice) - 1
            if 0 <= choice < len(products):
                quantity = int(input(f"Kaç adet {products[choice].name} almak istiyorsunuz? "))
                cart.add_product(products[choice], quantity)
            else:
                print("Geçersiz ürün numarası!")
        except ValueError:
            print("Geçerli bir sayı giriniz.")

    # Sipariş oluştur
    if cart.get_total() > 0:
        order = Order(customer, cart)
        order.confirm_order()
    else:
        print("Sepetiniz boş, sipariş oluşturulamadı.")

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

product.py

```
# product.py
class Product:
    def __init__(self, name, price, stock):
        self.name = name
        self.price = price
        self.stock = stock

    def __str__(self):
        return f"{self.name} - {self.price} TL - Stok: {self.stock}"

    def reduce_stock(self, quantity):
        """Stoktan belirtilen miktarda düşürme işlemi"""
        if self.stock >= quantity:
            self.stock -= quantity
            return True
        else:
            return False
```

cart.py

```
# cart.py
class Cart:
    def __init__(self):
        self.items = {}

    def add_product(self, product, quantity):
        """Ürünü sepete ekleme işlemi"""
        if product.reduce_stock(quantity):
            if product.name in self.items:
                self.items[product.name]['quantity'] += quantity
            else:
                self.items[product.name] = {'product': product, 'quantity': quantity}
            print(f"{quantity} adet {product.name} sepete eklendi.")
        else:
            print("Yetersiz stok!")

    def remove_product(self, product_name):
        """Sepetten ürün kaldırma"""
        if product_name in self.items:
            del self.items[product_name]
            print(f"{product_name} sepetten çıkarıldı.")

    def get_total(self):
        """Sepetin toplam fiyatını hesaplama"""
        return sum(item['product'].price * item['quantity'] for item in self.items.values())

    def display_cart(self):
        """Sepetteki ürünleri listeleme"""
        if not self.items:
            print("Sepetiniz boş.")
        else:
            for item in self.items.values():
                print(f"{item['quantity']} x {item['product'].name} = {item['product'].price * item['quantity']} TL")
            print(f"Toplam Tutar: {self.get_total()} TL")
```

customer.py

```
# customer.py
class Customer:
    def __init__(self, name, email):
        self.name = name
        self.email = email

    def __str__(self):
        return f"Müşteri: {self.name}, Email: {self.email}"
```

order.py

```
# order.py
class Order:
    def __init__(self, customer, cart):
        self.customer = customer
        self.cart = cart
        self.total_price = cart.get_total()

    def confirm_order(self):
        """Sipariş detaylarını görüntüleme"""
        print(f"Sipariş veren: {self.customer.name}")
        self.cart.display_cart()
        print(f"Ödenecek Tutar: {self.total_price} TL")
```

1️⃣ Python'da Bir Sınıf (Class) ve Nesne (Object) Oluşturma

Python'da bir sınıf, `class` anahtar kelimesi kullanılarak tanımlanır. Sınıftan nesne oluşturmak için, sınıf adı çağrılarak bir değişkene atanır.

Kod Örneği:

```
class Araba:
    def __init__(self, marka, model, yil):
        self.marka = marka
        self.model = model
        self.yil = yil

    def bilgi_goster(self):
        print(f"Araba: {self.marka} {self.model}, Yıl: {self.yil}")

# Nesne (Object) oluşturma
araba1 = Araba("Toyota", "Corolla", 2022)
araba1.bilgi_goster()
```

2️⃣ Encapsulation (Kapsülleme) Nedir?

Kapsülleme, sınıf içindeki verileri (değişkenleri) dışarıdan doğrudan erişime kapatıp, sadece belirlenen metotlar aracılığıyla erişim sağlamaktır.

Kod Örneği:

```
class BankaHesabi:
    def __init__(self, hesap_no, bakiye):
        self.hesap_no = hesap_no
        self.__bakiye = bakiye  # Özel değişken (private)

    def para_yatir(self, miktar):
        if miktar > 0:
            self.__bakiye += miktar
            print(f"{miktar} TL yatırıldı. Güncel bakiye: {self.__bakiye} TL")

    def bakiye_goster(self):
        print(f"Güncel bakiye: {self.__bakiye} TL")

# Nesne oluşturma
hesap = BankaHesabi("123456", 5000)
hesap.para_yatir(1000)
hesap.bakiye_goster()

# Dışarıdan erişim denemesi
# print(hesap.__bakiye)  # Hata verir! (Çünkü __bakiye özeldir)
```

3️⃣ Python'da `__init__` Metodunun Görevi

`__init__` metodu, bir sınıftan nesne oluşturulduğunda **otomatik olarak çalışan** yapıcı (constructor) metottur. Nesnenin ilk değerlerini ayarlamak için kullanılır.

Kod Örneği:

```
class Ogrenci:
    def __init__(self, ad, yas):
        self.ad = ad
        self.yas = yas
        print(f"Yeni öğrenci oluşturuldu: {self.ad}, {self.yas} yaşında.")

# Nesne oluşturma
ogrenci1 = Ogrenci("Ahmet", 20)
ogrenci2 = Ogrenci("Zeynep", 22)
```

4️⃣ Inheritance (Kalıtım) Nedir?

Kalıtım, bir sınıfın başka bir sınıftan özelliklerini ve metotlarını **miras almasını** sağlar. Böylece tekrar kod yazmaya gerek kalmaz.

Kod Örneği:

```
# Üst sınıf (Parent Class)
class Hayvan:
    def __init__(self, ad):
        self.ad = ad

    def ses_cikar(self):
        print("Bu hayvan bir ses çıkarır.")

# Alt sınıf (Child Class) - Kalıtım alıyor
class Kedi(Hayvan):
    def ses_cikar(self):
        print(f"{self.ad} miyavlıyor!")

class Kopek(Hayvan):
    def ses_cikar(self):
        print(f"{self.ad} havlıyor!")

# Nesneler oluşturma
kedi = Kedi("Minnak")
kopek = Kopek("Karabaş")

kedi.ses_cikar()  # Minnak miyavlıyor!
kopek.ses_cikar() # Karabaş havlıyor!
```

5️⃣ Polymorphism (Çok Biçimlilik) Nedir?

Polymorphism, aynı metot adının farklı sınıflarda farklı şekillerde kullanılabilmesidir.

Kod Örneği:

```
class Kus:
    def ses_cikar(self):
        print("Kuş cik cik eder.")

class Kedi:
    def ses_cikar(self):
        print("Kedi miyavlar.")

class Kopek:
    def ses_cikar(self):
        print("Köpek havlar.")

# Polymorphism kullanımı
hayvanlar = [Kus(), Kedi(), Kopek()]

for hayvan in hayvanlar:
    hayvan.ses_cikar()
```