**Edibe Berra Sarıkaya – İTÜ NOM Yazılım Araştırma Ödevi**

SORU-1: Object-Oriented Programming (OOP) nedir? Programlamada neden önemlidir? OOP’nin kullanım amaçları nelerdir? Detaylıca açıklayınız.

**1) OOP’nin Tanımı**  
Nesne Yönelimli Programlama (OOP), yazılımı sınıflardan üretilen nesneler etrafında örgütleyen bir yaklaşımdır. Nesneler, veri (durum) ile bu veriye çalışan metotları (davranış) birlikte barındırır. Böylece sistem, gerçek dünyadaki kavramların yazılımsal karşılıklarıyla modellenir.

**2) Temel İlkeler**

* **Kapsülleme:** Veriyi ve fonksiyonlarını bir arada tutar; yalnızca gereken kısımlar dışarı açılır (erişim denetimi).
* **Soyutlama:** Gereksiz iç detayları saklayıp yalnızca gerekli arayüzü sunar.
* **Kalıtım:** Bir sınıfın başka bir sınıftan özellik ve davranış devralmasıyla tekrarın azaltılması.
* **Çok Biçimlilik:** Aynı arayüzün farklı sınıflarda farklı biçimde uygulanabilmesi (örn. ciz() her şekilde farklı çalışır).

**3) Neden Önemlidir?**

* **Modülerlik ve bakım kolaylığı:** Kod parçalanır, test/onarım basitleşir.
* **Yeniden kullanılabilirlik:** Ortak davranışlar paylaşılır, tekrar yazım azaltılır.
* **Genişletilebilirlik:** Yeni özellikler mevcut arayüzleri bozmadan eklenebilir (Açık/Kapalı ilkesi).
* **Ekip çalışması ve tutarlılık:** Sözleşmeler (arayüz/soyut sınıf) paralel geliştirmeyi kolaylaştırır.
* **Gerçek dünya modellemesi:** İş kuralları güçlü alanlarda kavramsal eşleşme sağlar.

**4) Kullanım Amaçları (Nerede/Niçin Kullanılır?)**

* **Alan (domain) modelleme:** ERP, CRM, e-ticaret gibi kuralların ve ilişkilerin yoğun olduğu iş uygulamaları.
* **Arayüz ve uygulama geliştirme:** Masaüstü/ mobil/ oyun motorları (widget, actor, component yapıları).
* **Büyük ölçekli kurumsal yazılımlar:** Katmanlı mimari, bağımlılık enjeksiyonu, birim testi kolaylığı.
* **Eklenti/uzatma mimarileri:** Strateji/Komut/Ziyaretçi gibi desenlerle davranışı değiştirme ve genişletme.
* **Kütüphane/çatı (framework) geliştirme:** Arayüz/soyut sınıf sözleşmeleri üzerinden genişletilebilir yapı.

SORU-2: Threading ve Multithreading kavramlarını detaylıca acıklayın. Hangi durumlarda neden kullanılır açıklayınız.

Thread (iş parçacığı), bir program içerisinde çalışan en küçük işlem birimidir ve programın komutlarının ardışık olarak yürütülmesini sağlar. Varsayılan olarak çoğu yazılım tek bir thread ile çalışır ve işlemler sıralı bir şekilde gerçekleştiği için bazı durumlarda programın beklemesine ve yavaş çalışmasına neden olabilir.

Multithreading ise bir program içerisinde aynı anda birden fazla thread’in çalıştırılmasıdır. Bu yöntem sayesinde programın farklı görevleri eşzamanlı olarak yürütmesi mümkün olur. Böylece hem işlemci kaynakları daha verimli kullanılır hem de programın genel çalışma süresi kısalır.

Multithreading özellikle aşağıdaki durumlarda tercih edilir:

* **I/O bekleme işlemleri:** Dosya okuma, veri tabanı bağlantısı veya internetten veri alma gibi beklemeli işlemlerde, ana thread durmak yerine diğer thread’ler işlemlere devam eder.
* **CPU yoğunluklu işlemler:** Matematiksel hesaplamalar, görüntü işleme veya şifreleme gibi çok çekirdekli işlemcilerde paralel yürütülebilen görevlerde performansı ciddi ölçüde arttırır.
* **Arayüz (UI) uygulamaları:** Kullanıcı arayüzünün donmasını engellemek için arka plan işlemleri ayrı thread’lerde çalıştırılır. Bu sayede program hem işlem yapar hem de kullanıcı etkileşimini kesintisiz sürdürür.
* **Gerçek zamanlı sistemler:** Oyun motorları, robotik sistemler, sensör uygulamaları gibi birden fazla görevin eşzamanlı yönetilmesi gereken senaryolarda multithreading kritik öneme sahiptir.

**Sonuç olarak**, multithreading kullanımı; performans artışı, daha verimli kaynak kullanımı ve kullanıcı deneyiminin iyileştirilmesi amacıyla tercih edilir. Doğru planlanan thread yapıları programı hızlandırırken, yanlış kullanım senkronizasyon sorunlarına yol açabileceğinden dikkatli tasarım gerektirir.

SORU-3: Görüntü işlemede optimizasyon yöntemleri nelerdir? Görüntü aktarımında gecikmeyi azaltmak için hangi yöntemler kullanılır? Detaylı bir şekilde açıklayınız.

**1) Görüntü işlemede optimizasyon yöntemleri:**  
Görüntü işleme uygulamalarında amaç işlemleri **daha hızlı**, **daha az kaynak kullanarak** ve **gerçek zamana yakın** şekilde yapmaktır. Bu nedenle çeşitli optimizasyon yöntemleri kullanılır:

* Gereksiz bölgeleri işlememek için **ROI (Region of Interest)** kullanma
* İşlemleri paralel çalıştırmak için **çok çekirdekli işlemci veya GPU kullanma**
* **Daha basit filtreler ve algoritmalar** tercih etme (örneğin karmaşık konvolüsyon yerine ayrılabilir filtre)
* **Vektörizasyon ve SIMD** komutları ile aynı anda birden fazla pikseli işleme
* **Daha düşük çözünürlükte önce işleyip ardından geliştirme** (çoklu çözünürlük yaklaşımı)
* Derin öğrenme kullanılıyorsa modeli hızlandırmak için **kuantizasyon veya budama (pruning)**

Bu yöntemler, görüntü işlemenin hızını artırır ve işlem maliyetini azaltır.

**2) Görüntü Aktarımında Gecikmeyi Azaltma Yöntemleri:**  
Görüntü aktarımında amaç, görüntünün **karşı tarafa mümkün olan en kısa sürede ulaşmasıdır**. Yani “yayın donmasın, görüntü geç gelmesin”. Bunun için:

* **Daha hızlı protokoller** kullanılır (örnek: **UDP, RTP, WebRTC**)  
  *Bu protokoller veri kaybolsa bile tekrar beklemez, bu yüzden hızlıdır.*
* **Düşük gecikmeli codec ayarları** kullanılır (örnek: *H.264 low-latency*)  
  *Codec = videoyu sıkıştıran yöntem; doğru ayar gecikmeyi azaltır.*
* **Çözünürlük ve FPS düşürülebilir** (örnek: 1080p yerine 720p, 60 fps yerine 30 fps)  
  *Daha az veri = daha hızlı iletim.*
* **Donanım hızlandırmalı kodlama** kullanılır (örnek: *GPU, NVENC, QuickSync*)  
  *İşlemciye yük binmez, kodlama daha hızlı olur.*
* Sadece gerekli kısmın gönderilmesi (**ROI gönderimi**)  
  *Bütün görüntü yerine sadece önemli bölge yollanır → veri azalır → hız artar.*

SORU-4: Yapay zekâ modeli eğitiminde önemli kriterler nelerdir? Model eğitirken nelere dikkat edilmelidir? Detaylıca açıklayınız.

Bir yapay zekâ modelinin başarılı olabilmesi için, eğitim sürecinde hem verinin hem de model ayarlarının doğru seçilmesi gerekir. Model eğitirken dikkat edilmesi gereken başlıca noktalar şunlardır:

* **Veri Kalitesi:** Kullanılan veriler doğru, temiz ve yeterli olmalıdır. Hatalı veya eksik veriler modelin yanlış öğrenmesine neden olur.
* **Veri Dengesi ve Çeşitliliği:** Veri seti farklı durumları içermeli ve sınıflar arasında dengesizlik olmamalıdır. Böylece model genelleme yapmayı öğrenir.
* **Doğru Veri Bölme:** Veriler eğitim, doğrulama ve test olarak ayrılmalı; test verisi eğitimde kullanılmamalıdır.
* **Uygun Model Seçimi:** Problemin türüne uygun model mimarisi kullanılmalı, model ne çok basit ne de aşırı karmaşık olmalıdır.
* **Eğitim Ayarları:** Learning rate, epoch ve batch size gibi parametreler uygun seviyede seçilmelidir. Ayrıca doğru loss fonksiyonu ve değerlendirme metriği kullanılmalıdır.
* **Overfitting’i Önleme:** Dropout, regularization veya veri artırma (augmentation) gibi teknikler aşırı ezberlemeyi engeller.
* **Eğitim Sürecini İzleme:** Eğitim boyunca doğrulama sonuçları takip edilmeli ve gerekirse early stopping ile eğitim doğru noktada durdurulmalıdır.

**Sonuç olarak**, yapay zekâ eğitiminde amaç, sadece veriyi ezberleyen değil, yeni verilerde de doğru sonuç üretebilen bir model geliştirmektir. Bu nedenle veri kalitesi, model seçimi ve eğitim ayarları dikkatli planlanmalıdır.

SORU-5: MAVLINK protoklü nedir? Mavproxy nedir? UDP – TCP portları arasındaki fark nedir port çakışması nedir ve nasıl çözülür? Detaylıca açıklayınız

**MAVLink**, insansız hava araçlarında (İHA/UAV) kullanılan hafif ve hızlı bir haberleşme protokolüdür. Yer istasyonu ile drone arasında telemetri, konum, sensör ve komut bilgilerinin düzenli bir formatla iletilmesini sağlar. Açık kaynak yapıda olduğu için ArduPilot ve PX4 gibi sistemlerde standart olarak kullanılır.

**MAVProxy**, MAVLink mesajlarını okuyup işleyebilen komut satırı tabanlı bir yer istasyonu yazılımıdır. Bu program sayesinde telemetri izlenebilir, uçuş modları değiştirilebilir ve drone’a komut gönderilebilir. Ayrıca aynı veriyi birden fazla cihaza dağıtma özelliği sayesinde router gibi de kullanılabilir.

**UDP ve TCP farkı:**

* **TCP**, veri gönderirken güvenliği ve eksiksiz iletimi garanti eden bir protokoldür. Kargo gibi çalışır: Paket ulaşmazsa tekrar gönderir. Bu nedenle güvenilirdir ancak daha yavaş ve gecikmesi fazladır.
* **UDP** ise çok hızlı çalışan ama kontrol mekanizması olmayan bir protokoldür. “Gönder gitsin” mantığıyla çalışır ve paket kaybolsa bile beklemez. Bu yüzden gecikme çok düşüktür. Gerçek zamanlı sistemlerde (drone telemetrisi, canlı yayın, oyunlar) UDP tercih edilir.

**Port çakışması**, aynı port numarasını aynı anda iki farklı programın kullanmaya çalışmasıdır. Böyle bir durumda bağlantı kurulamaz ve genellikle “port is already in use” hatası görülür. Çözmek için:

* Aynı portu kullanan uygulamalardan biri kapatılır veya
* İkinci uygulama için farklı bir port numarası seçilir (örneğin 14550 yerine 14551 kullanmak).

**Sonuç olarak**, MAVLink İHA iletişimini sağlayan protokol, MAVProxy ise bu protokolle haberleşmeyi yöneten yazılımdır. UDP düşük gecikmesi nedeniyle drone uygulamalarında tercih edilir ve port çakışması durumunda farklı port seçilerek sorun kolayca çözülür.

SORU-6: Socket haberleşmesi nedir? Socket haberleşmesinde çıkabilecek olası problemler nelerdir ve nasıl çözülebilir? Detaylıca açıklayınız.

Socket haberleşmesi, iki cihazın ağ üzerinden birbirine **bağlanarak veri gönderip almasını** sağlayan iletişim yöntemidir. Bir taraf *sunucu (server)* olarak belirli bir **portu dinler**, diğer taraf *istemci (client)* o porta bağlanır ve böylece veri alışverişi başlar.

Socket haberleşmesi sırasında bazı problemler yaşanabilir:

* **Bağlantı kurulamaması:** Yanlış IP, yanlış port veya güvenlik duvarı nedeniyle olabilir.  
  **Çözüm:** IP ve port bilgisi kontrol edilir, firewall engeli kaldırılır.
* **Port çakışması:** Aynı portu aynı anda iki program kullanmaya çalıştığında oluşur.  
  **Çözüm:** Programlardan biri kapatılır veya farklı bir port numarası seçilir.
* **Bağlantının kopması (timeout):** İnternet kesintisi veya zayıf ağdan kaynaklanır.  
  **Çözüm:** Timeout kontrolü yapılır ve otomatik yeniden bağlanma özelliği eklenir.
* **Veri kaybı (özellikle UDP’de):** Ağ yoğunluğu nedeniyle paketler kaybolabilir.  
  **Çözüm:** Gerekirse TCP kullanılır veya paketlere numara eklenerek eksikler yeniden gönderilir.

Sonuç olarak, socket haberleşmesinde temel amaç cihazların ağ üzerinden iletişim kurmasını sağlamaktır. Olası sorunlar doğru port ayarı, bağlantı kontrolü ve yeniden deneme mekanizmaları kullanılarak kolayca çözülebilir.