Bu föyde yapısal diyagramlar ve davranışsal diyagramlar olmak üzere iki grup vardır. Bu gruplarda 4 soru vardır ve her gruptan 2 soru çözülecektir.

**Davranışsal Diyagramlar**

1. Ekleme sıralamasının (Insertion sort) akış şemasını çiziniz.
2. Aşağıdaki faaliyet diyagramını çiziniz.

**Yemek Teslimatı Sipariş Sistemi:** Yemek siparişi vermenin ana sürecini temsil eder.

**Başlangıç:** Faaliyet diyagramının başlangıç noktası.

**Restoran Seç:** Kullanıcı sipariş vermek için bir restoran seçer.

**Menüye Gözat:** Kullanıcı seçilen restoranın menüsüne göz atar.

**İstenen Öğe Bulundu mu?:** İstenen öğenin menüde bulunup bulunmadığını kontrol etmek için karar noktası.

**Ürünü Sepete Ekle:** İstenen ürün bulunursa, sepete eklenir.

**Alışverişe Devam Edilsin mi?:** Kullanıcının alışverişe devam etmek isteyip istemediğini kontrol etmek için karar noktası.

**Daha Fazla Menüye Göz At:** Kullanıcı alışverişe devam etmek isterse, daha fazla menü öğesine göz atar.

**Sepete Daha Fazla Ürün Ekle:** Kullanıcı sepete daha fazla ürün ekler.

**Ödeme İşlemine Geç:** İstenen tüm ürünler sepete eklendikten sonra kullanıcı ödeme işlemine geçer.

**Siparişi Onayla:** Kullanıcı siparişi onaylar.

**Teslimat Adresi Ver:** Kullanıcı teslimat adresini verir.

**Ödeme Yöntemini Seçin:** Kullanıcı ödeme yöntemini seçer.

**Ödeme Yap: Kullanıcı ödemeyi yapar:** Kullanıcı ödemeyi yapar.

**Sipariş Onayı:** Kullanıcı siparişin onayını alır.

**Bitiş:** Faaliyet diyagramının bitiş noktası.

1. Aşağıdaki kullanım durumu diyagramını çiziniz. (Include ve extend kullanılacak)

**Aktör:** Öğrenci

**Kullanım Durumları:** Kurslara Göz Atın

Kursa Kaydolun

Kurs Materyallerini Görüntüleyin

Ödevleri Tamamlayın

Tartışma Forumlarına Katılın

Bildirim Alın

**Aktör:** Eğitmen

**Kullanım Durumları:** Kurs Oluştur

Kurs Materyallerini Yükle

Ödevlere Not Ver

Tartışma Forumlarını Yönet

**Aktör:** Yönetici

**Kullanım Durumları:** Kullanıcı Hesaplarını Yönetme

Kursları Yönetme

Rapor Oluşturma

1. Aşağıdaki sıra diyagramını çiziniz.

**Aktörler:**

CPU

Bellek

G/Ç Cihazı

CPU belleğe bir istek göndererek okuma işlemini başlatır.

Bellek isteği alır ve yerine getirmek için hazırlanır.

Bellek istenen verileri almak için G/Ç cihazına bir okuma isteği gönderir.

G/Ç cihazı verileri depolama alanından alır.

G/Ç cihazı alınan verileri belleğe geri gönderir.

Bellek verileri alır ve CPU'ya geri göndermek için hazırlanır.

Bellek verileri CPU'ya gönderir.

CPU verileri alır ve hesaplamanın sonraki adımlarına devam eder.

**Yapısal Diyagramlar**

1. Aşağıdaki sınıf diyagramını çiziniz.

|  |  |
| --- | --- |
| User  ---------------------------  - userID: int  - username: string  - email: string  - password: string  ---------------------------  + browseContent(): void  + searchContent(): void  + watchContent(): void  + rateContent(): void  + writeReview(): void  --------------------------- | Profile  ---------------------------  - profileID: int  - name: string  - age: int  - gender: string  - language: string  ---------------------------  + addProfile(): void  + deleteProfile(): void  + updateProfile(): void  --------------------------- |
| Subscription  ---------------------------  - subscriptionID: int  - planName: string  - price: float  ---------------------------  + subscribe(): void  + cancelSubscription(): void  --------------------------- | Movie  ---------------------------  - movieID: int  - title: string  - releaseYear: int  - duration: int  - director: Director  - genres: Genre[]  - actors: Actor[]  - rating: Rating  ---------------------------  + getMovieDetails(): void  + playMovie(): void  --------------------------- |
| TVShow  ---------------------------  - tvShowID: int  - title: string  - releaseYear: int  - seasons: int  - episodesPerSeason: int  - directors: Director[]  - genres: Genre[]  - actors: Actor[]  - rating: Rating  ---------------------------  + getTVShowDetails(): void  + playEpisode(): void  --------------------------- | Genre  ---------------------------  - genreID: int  - name: string  --------------------------- |
| Director  ---------------------------  - directorID: int  - name: string  --------------------------- | Actor  ---------------------------  - actor |

1. Aşağıdaki paket diyagramını çiziniz.

**Ana Paket:** Veri Yapıları

**Alt Paket:** Arrays

**Özellikleri:**

Static Array

Dynamic Array

Multidimensional Array

**Alt Paket:** Lists

**Özellikleri:**

Linked List

DoublyLinkedList

**Alt Paket:** Stacks

**Özellikleri:**

Stack

LinkedStack

**Alt Paket:** Queues

**Özellikleri:**

Queue

LinkedQueue

**Alt Paket:** Trees

**Özellikleri:**

BinaryTree

BinarySearchTree

AVLTree

**Alt Paket:** Graphs

**Özellikleri:**

Graph

DirectedGraph

WeightedGraph

**Alt Paket:** Heaps

**Özellikleri:**

MinHeap

MaxHeap

1. Aşağıdaki Varlık İlişki Diyagramını çiziniz.

User

user\_id (Primary Key)

username

email

bio

location

join\_date

Repository

repo\_id (Primary Key)

repo\_name

description

creation\_date

user\_id (Foreign Key referencing User)

Issue

issue\_id (Primary Key)

title

description

creation\_date

user\_id (Foreign Key referencing User)

repo\_id (Foreign Key referencing Repository)

Pull Request

pr\_id (Primary Key)

title

description

creation\_date

user\_id (Foreign Key referencing User)

repo\_id (Foreign Key referencing Repository)

Commit

commit\_id (Primary Key)

message

timestamp

user\_id (Foreign Key referencing User)

repo\_id (Foreign Key referencing Repository)

Bir Kullanıcı birçok depoya sahip olabilir, ancak bir depo yalnızca bir kullanıcıya aittir.

Bir Kullanıcı birçok sorun oluşturabilir, ancak bir sorun yalnızca bir kullanıcı tarafından oluşturulur.

Bir Depo birçok soruna sahip olabilir, ancak bir sorun yalnızca bir depoya aittir.

Bir Kullanıcı birçok çekme isteği oluşturabilir, ancak bir çekme isteği yalnızca bir kullanıcı tarafından oluşturulur.

Bir Depo birçok çekme isteğine sahip olabilir, ancak bir çekme isteği yalnızca bir depoya aittir.

Bir Kullanıcı birçok commit yapabilir, ancak bir commit yalnızca bir kullanıcı tarafından yapılır.

Bir Depo birçok commit'e sahip olabilir, ancak bir commit yalnızca bir depoya aittir.

1. Aşağıdaki durum makinesi diyagramını çiziniz.

Getirme: CPU bellekten bir sonraki komutu alır.

Kod Çözme: Getirilen komut, gerçekleştirilecek işlemi belirlemek için deşifre edilir. Yürüt: Kodu çözülen komut yürütülür, bu da aritmetik, mantıksal veya kontrol aktarımı gibi işlemleri içerebilir.

Bellek Erişimi: Komut belleğe erişim gerektiriyorsa (örn. yükleme/depolama işlemleri), bu durum bunu gerçekleştirir.

Geri Yaz: Gerekirse, yürütülen komutun sonuçları kayıtlara veya belleğe geri yazılır.

**Geçişler:**

Getirme -> Kod Çözme: Geçiş komutun getirilmesinden sonra gerçekleşir.

Kod Çözme -> Yürütme: Geçiş, komutun kodunu çözdükten sonra gerçekleşir.

Yürüt -> Bellek Erişimi: Geçiş, bellek erişimi içeren bir komut yürütüldükten sonra gerçekleşir.

Bellek Erişimi -> Geri Yazma: Gerekirse bellek erişimi tamamlandıktan sonra geçiş gerçekleşir.

Geri Yaz -> Getir: Geçiş, sonuçları geri yazdıktan sonra gerçekleşir ve bir sonraki komut döngüsüne başlamak için getirme durumuna geri döner.