Bu ödev kapsamında BMP görüntü işleme ödevini yaptım. Ödevde C++ programlama dili kullanarak BMP formatındaki bir görüntü dosyasını okuyabilen, bu dosya üzerinde işlem yapabilen ve sonucu yeni bir BMP dosyası olarak kaydedebilen bir program geliştirdim. Yazdığım programın kaynak kodlarını ve bu çalışmayı detaylı şekilde açıkladığım 2–3 sayfalık raporu PDF formatında bu ödev alanına yükledim.

Programda ilk olarak BMP dosyaları hakkında araştırma yaptım ve BMP dosya yapısını inceledim. BMP dosyalarının ilk 54 byte'lık kısmının header bilgilerini içerdiğini, bu bölümde dosyanın boyutları, renk derinliği ve veri yerleşimi gibi bilgilerin bulunduğunu öğrendim. Bu bilgileri kullanarak BMP dosyasını doğru şekilde okuyabilmek için programımı bu yapıya uygun olarak tasarladım.

Kodda `KImage` adında bir sınıf oluşturdum. Bu sınıf BMP dosyasını yükleme, görüntü bilgilerini okuma, piksel verileri üzerinde işlem yapma ve sonucu dosyaya yazma işlemlerinden sorumludur. Sınıf içerisinde BMP dosyasının header bilgilerini tutmak için 54 byte'lık bir `unsigned char` dizisi kullandım. Piksel verilerini tutmak için ise dinamik olarak ayrılan bir `unsigned char` işaretçisi tanımladım. Bunun sebebi, görüntü boyutlarının dosyaya göre değişebilmesidir.

BMP dosyasını okumak için `ifstream` sınıfını binary modda kullandım. Dosyanın başarılı bir şekilde açılıp açılmadığını kontrol ettim. Dosya açılamazsa kullanıcıya hata mesajı verdim ve programı sonlandırdım. Daha sonra dosyanın ilk iki byte'ını kontrol ederek dosyanın gerçekten bir BMP dosyası olup olmadığını denetledim. BMP dosyalarında bu iki byte "B" ve "M" karakterlerinden oluştuğu için bu kontrolü yaptım.

Header bilgilerini okuduktan sonra görüntünün genişlik, yükseklik ve renk derinliği bilgilerini header içinden aldım. Bu değerleri alırken little-endian byte sıralamasına dikkat ettim. Bu amaçla 2 byte ve 4 byte'lık verileri doğru şekilde integer ve short türlerine çeviren yardımcı fonksiyonlar yazdım. Bu fonksiyonlar sayesinde header içindeki ham byte verilerini doğru şekilde yorumladım.

Programda yalnızca 24-bit ve 32-bit BMP dosyalarını destekleyecek şekilde bir kontrol ekledim. Bunun sebebi, bu formatların RGB ve RGBA yapısına sahip olması ve piksel işlemlerinin daha kolay yapılabilmesidir. Farklı bit derinliğine sahip BMP dosyaları için hata mesajı vererek programın devam etmesini engelledim.

Renk derinliği bilgisini aldıktan sonra piksel başına düşen byte sayısını hesapladım. 24-bit BMP dosyalarında her piksel 3 byte, 32-bit BMP dosyalarında ise 4 byte yer kaplamaktadır. BMP dosyalarında her satırın 4 byte hizalamaya sahip olması gerektiği için satır uzunluğunu padding'i de dikkate alarak hesapladım. Bu hesaplama sayesinde piksel verilerini doğru indekslerle okuyabildim.

Görüntüye ait tüm piksel verilerini tutabilmek için dinamik bellek ayırdım. Daha sonra dosyadan okuduğum piksel verilerini bu alana aktardım. Dinamik bellek kullanmamın sebebi, görüntü boyutlarının sabit olmaması ve farklı BMP dosyalarıyla da çalışabilmektir. Program sonunda bu belleği serbest bırakarak bellek sızıntılarının önüne geçtim.

Görüntü işleme aşamasında, görüntünün merkez noktasını hesapladım. Merkez koordinatlarını genişlik ve yükseklik değerlerini kullanarak belirledim. Daha sonra bu

merkezin etrafında kullanıcı tarafından verilen boyutta kare bir alan oluşturdum. Bu alan içerisindeki pikseller üzerinde işlem yaparak görüntünün ortasında siyah bir kare elde ettim.

Piksel işlemleri sırasında BMP dosyalarının BGR renk sıralamasına sahip olduğunu dikkate aldım. Belirlediğim kare alan içerisindeki piksellerin mavi, yeşil ve kırmızı değerlerini sıfırladım. Eğer görüntü 32-bit ise alfa kanalını 255 olarak ayarladım. Bu işlemleri yaparken her piksel için doğru indeksi hesaplayarak doğrudan piksel dizisi üzerinde çalıştım. Böylece görüntünün sadece orta kısmı değiştirilmiş oldu, diğer alanlar korunmuş oldu.

Görüntü işleme işlemi tamamlandıktan sonra, orijinal BMP header bilgilerini ve güncellenmiş piksel verilerini kullanarak yeni bir BMP dosyası oluşturdum. Çıktı dosyasını `output.bmp` adıyla kaydettim. Bu sayede orijinal dosya üzerinde herhangi bir değişiklik yapmadan sonucu ayrı bir dosyada sakladım.

Programın çalışma sürecinde C++ dilinde dosya okuma ve yazma işlemlerini, dinamik bellek yönetimini ve düşük seviyeli byte işlemlerini uygulamalı olarak öğrenmiş oldum. Özellikle BMP dosya yapısını ve padding kavramını daha iyi anladım. Piksel tabanlı işlemlerin nasıl yapıldığını ve görüntü üzerinde matematiksel indeks hesaplamalarının önemini görmüş oldum.

Sonuç olarak bu ödev, C++ programlama dili ile görüntü işleme konusuna giriş niteliğinde faydalı bir çalışma olmuştur.