**Algoritma ve Programlamaya Giriş Dersi 1. Şube 2. Grup Proje Ödevi Raporu**  
**Öğrencilerin Adı-Soyadı:**

* Harun Reşit Mercan (23181616055)
* Yunus Kavaklı (23181617005)
* Sefer Ağca (23181617004)

**Proje Konusu:**

Bu C programı, bir fabrika giriş çıkış sisteminin yönetimini sağlayan bir uygulamadır. Çalışanların giriş ve çıkış saatleri günlük olarak kaydedilir ve bu veriler üzerinde analizler yapılır. Program, çalışanların geç kalma ve erken çıkma durumlarını belirler, tüm çalışan kayıtlarını görüntüler, yeni kayıt ekler ve kayıt silme işlemlerini gerçekleştirir. Ayrıca bu analizlerin sonuçları ayrı dosyalara kaydedilir.

**GitHub: https://github.com/Yunus-Kavakli/1.sube\_10.Grup\_proje**

Programın temel işlevleri:

1. **Giriş ve Çıkış Verilerinin Kaydedilmesi:** Çalışanlar için sicil numarası, saat, dakika ve durum bilgileri girilerek bu veriler bir dosyada saklanır.
2. **Geç ve Erken Gelenlerin Belirlenmesi:** Günlük kayıtlar üzerinden analiz yapılarak en geç gelen ve en erken çıkan çalışanlar tespit edilir ve ayrı dosyalara kaydedilir.
3. **Tüm Çalışanların Görüntülenmesi:** Günlük giriş-çıkış verileri okunarak ekranda görüntülenir.
4. **Kayıt Silme:** Belirli bir sicil numarasına sahip çalışanın kaydı silinebilir.

**Toplantı Tarihi:** 15/12/2024  
**Katılımcılar:**

* Harun Reşit Mercan
* Yunus Kavaklı
* Sefer Ağca

**Toplantının Gündem Maddeleri:**

1. Proje hedeflerinin ve gereksinimlerinin netleştirildi
2. Kodlama ve tasarım sürecine ilişkin bir yol haritasının belirlendi
3. Görev dağılımının yapıldı
4. Dosya işlemleri ve analiz yöntemleri hakkında bilgi paylaşımı yapıldı
5. Projenin teslim tarihine kadar izlenecek süreç planlandı

**Toplantının İçeriği:**

Toplantının başlangıcında proje konusu detaylı bir şekilde ele alındı. Çalışan giriş çıkış verilerinin nasıl yönetileceği, analizlerin nasıl yapılacağı ve verilerin dosyalara nasıl kaydedileceği üzerine fikir birliğine varıldı.

Katılımcılar, proje kapsamında eksik oldukları konuları belirlediler. Özellikle dosya işlemleri ve struct kullanımı konularında araştırma yapma kararı alındı. Bu eksikliklerin giderilmesi amacıyla gerekli kaynaklar paylaşıldı ve grup üyeleri arasında bilgi alışverişi yapıldı.

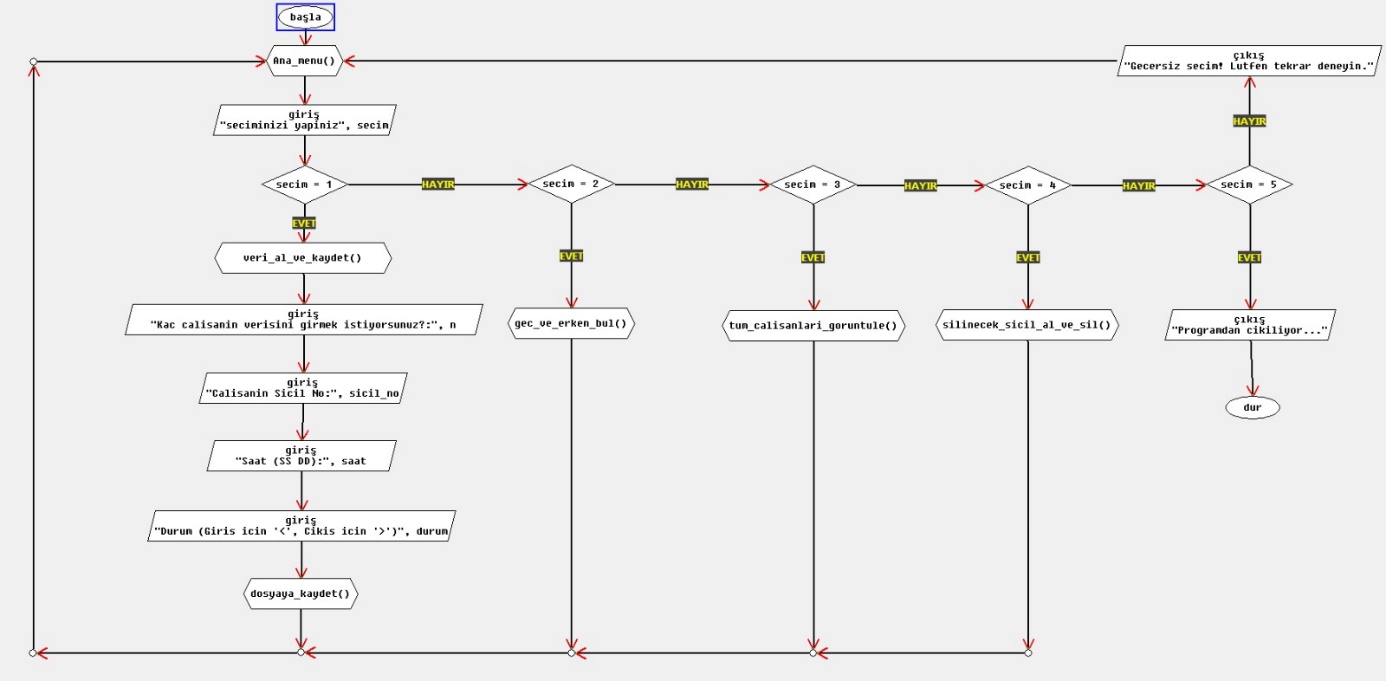
**Görev Dağılımı:**

* **Kodlama:** Harun Reşit Mercan ve Yunus Kavaklı, dosya işlemleri ve algoritma geliştirme üzerinde çalışacak.
* **Rapor Yazımı:** Harun Reşit Mercan ve Sefer Ağca proje raporunu hazırlayacak ve toplantı detaylarını yazacak.
* **Dosya ve Akış Şeması:** Yunus Kavaklı ve Sefer Ağca, programın işleyişini temsil eden akış şemalarını birlikte hazırlayacak.

**Projenin Teslimine Kadar İzlenecek Süreç:**

1. İlk olarak, temel fonksiyonların geliştirilmesi ve dosya işlemlerinin doğrulanması hedeflendi.
2. İkinci aşamada, verilerin işlenmesi ve düzenlenmesi üzerine çalışmalar yapıldı.
3. Son aşamada ise, raporun yazılması ve projenin genel testlerinin gerçekleştirilmesi kararlaştırıldı.

**Sonuç:**  
Toplantıda alınan kararlar doğrultusunda, grup üyeleri proje süreci boyunca birbirlerine destek olma konusunda fikir birliğine vardı. Görev dağılımı yapılarak her bir üyenin sorumlulukları netleştirildi. Projenin başarıyla tamamlanması için düzenli aralıklarla kontrol toplantılarının yapılması kararlaştırıldı.

**Akış Şeması:**

**Program Kodları:**

#include <stdio.h> // Standart girdi/çıktı işlemleri için gerekli kütüphane (printf, scanf gibi)  
#include <string.h> // String (dizi) işlemleri için gerekli kütüphane (strcpy, strcmp gibi)  
#include <stdbool.h> // Boolean (true/false) değerler için gerekli kütüphane  
#include <stdarg.h> // Değişken sayıda argüman alan fonksiyonlar için gerekli kütüphane  
  
#define MAX\_CALISAN 100 // En fazla 100 çalışan kaydedilebilir.  
#define MAX\_SATIR 100 // Bir satırda en fazla 100 karakter olabilir.  
  
// Çalışan verilerini tutmak için kullanılan struct yapısı  
typedef struct {  
 char sicil\_no[20]; // Çalışanın sicil numarası (max 20 karakter)  
 int saat; // Giriş/Çıkış saati  
 int dakika; // Giriş/Çıkış dakikası  
 char durum; // Çalışanın giriş ('<') veya çıkış ('>') durumu  
} Calisan;  
  
// Fonksiyonların prototipleri (yani isimleri ve ne alıp ne döndürdükleri)  
void veri\_al\_ve\_kaydet(); // Kullanıcıdan çalışan verilerini alıp dosyaya kaydeder  
void gec\_ve\_erken\_bul(); // Dosyadaki verilere göre en geç ve en erken gelenleri bulur  
void tum\_calisanlari\_goruntule(); // Tüm çalışan verilerini ekrana yazdırır  
int saat\_dakika\_fark(int saat1, int dakika1, int saat2, int dakika2); // İki saat arasındaki farkı dakika cinsinden hesaplar  
void dosyaya\_yaz(const char \*dosya\_adi, const char \*format, ...); // Verileri belirtilen formatta dosyaya yazar  
void gec\_ve\_erken\_dosyaya\_yaz(const char \*dosya\_adi, const char \*metin, bool bulundu, const char \*sicil, int saat, int dakika, int fark); // Geç/erken verilerini dosyaya yazar  
void calisan\_kaydini\_sil(const char \*silinecek\_sicil); // Belirtilen sicil numarasına sahip çalışanın kaydını siler  
void silinecek\_sicil\_al\_ve\_sil(); // Kullanıcıdan silinecek sicil numarasını alıp silme işlemini yapar  
  
// Ana fonksiyon (programın başladığı yer)  
int main() {  
 int secim; // Kullanıcının menüden seçtiği seçenek  
 while (1) { // Sonsuz döngü (kullanıcı çıkış yapana kadar devam eder)  
 printf("\n\*\*\* Fabrika Giris Sistemi \*\*\*\n");  
 printf("1. Calisan Giris ve Cikis\n");  
 printf("2. Gec ve Erken Gelenler\n");  
 printf("3. Tum Calisanlari Goruntule\n");  
 printf("4. Calisan Kaydini Sil\n");  
 printf("5. Cikis\n");  
 printf("Seciminizi yapin: ");  
 scanf("%d", &secim); // Kullanıcının seçimi okunur  
 while (getchar() != '\n'); // Buffer temizleme (scanf'ten sonraki gereksiz karakterleri temizlemek için)  
 switch (secim) { // Kullanıcının seçimine göre işlem yapılır  
 case 1:  
 veri\_al\_ve\_kaydet(); // Çalışan verilerini alıp kaydetme fonksiyonu çağrılır  
 break;  
 case 2:  
 gec\_ve\_erken\_bul(); // Geç ve erken gelenleri bulma fonksiyonu çağrılır  
 break;  
 case 3:  
 tum\_calisanlari\_goruntule(); // Tüm çalışanları görüntüleme fonksiyonu çağrılır  
 break;  
 case 4:  
 silinecek\_sicil\_al\_ve\_sil(); // Çalışan kaydını silme fonksiyonu çağrılır  
 break;  
 case 5:  
 printf("Programdan cikiliyor...\n");  
 return 0; // Programdan çıkış  
 default:  
 printf("Gecersiz secim! Lutfen tekrar deneyin.\n"); // Geçersiz seçim yapıldığında hata mesajı  
 }  
 }  
}

**Fonksiyonlar:**

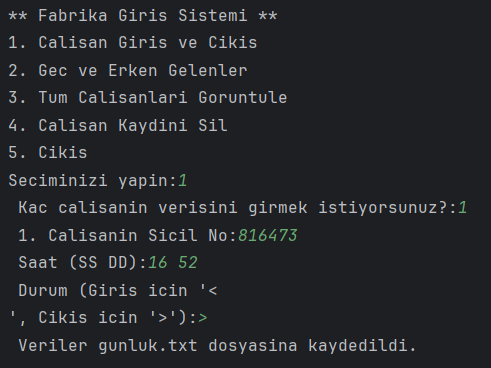
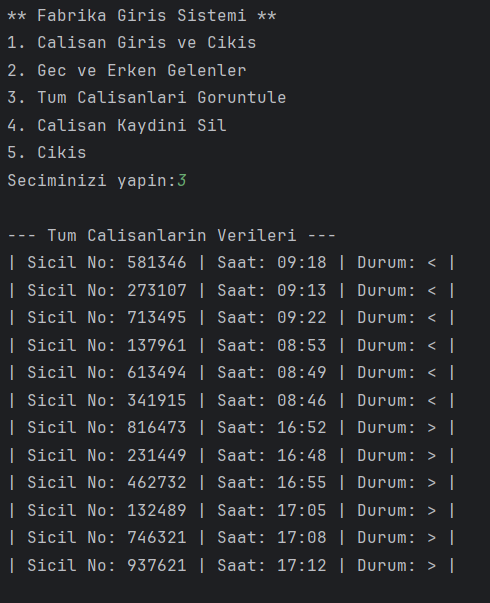
// Kullanıcıdan çalışan verilerini alıp "gunluk.txt" dosyasına kaydeder  
void veri\_al\_ve\_kaydet() {  
 int n;  
 printf("Kac calisanin verisini girmek istiyorsunuz?: ");  
 scanf("%d", &n); // Kaç çalışan verisi girileceği kullanıcıdan alınır  
 while (getchar() != '\n'); // Buffer temizleme  
 Calisan calisanlar[MAX\_CALISAN]; // Çalışan bilgilerini tutacak dizi  
 for (int i = 0; i < n; i++) { // Belirtilen sayıda çalışan için döngü  
 char sicil\_input[MAX\_SATIR];  
 printf("%d. Calisanin Sicil No: ", i + 1);  
 fgets(sicil\_input, sizeof(sicil\_input), stdin); // Sicil numarası kullanıcıdan alınır  
 if(sscanf(sicil\_input, "%s", calisanlar[i].sicil\_no) != 1){ // Sicil numarası düzgün alınmazsa hata mesajı  
 printf("Hatali sicil numarasi girisi!\n");  
 i--; // Döngüde bir geri giderek tekrar giriş yapmasını sağlar  
 continue;  
 }  
  
 printf("Saat (SS DD): ");  
 if (scanf("%d %d", &calisanlar[i].saat, &calisanlar[i].dakika) != 2) { // Saat ve dakika bilgileri alınır  
 printf("Hatali saat girisi!\n");  
 while (getchar() != '\n'); // Hatalı giriş durumunda buffer temizlenir  
 i--;  
 continue;  
 }  
 while (getchar() != '\n'); // Buffer temizleme  
  
 printf("Durum (Giris icin '<', Cikis icin '>'): ");  
 if(scanf(" %c", &calisanlar[i].durum) != 1){ // Durum bilgisi alınır  
 printf("Hatali durum girisi!\n");  
 while (getchar() != '\n'); // Hatalı giriş durumunda buffer temizlenir  
 i--;  
 continue;  
 }  
 while (getchar() != '\n'); // Buffer temizleme  
  
 // Çalışan bilgileri "gunluk.txt" dosyasına yazılır  
 dosyaya\_yaz("gunluk.txt", "%s %02d:%02d %c\n", calisanlar[i].sicil\_no, calisanlar[i].saat, calisanlar[i].dakika, calisanlar[i].durum);  
 }  
  
 printf("Veriler gunluk.txt dosyasina kaydedildi.\n");  
}

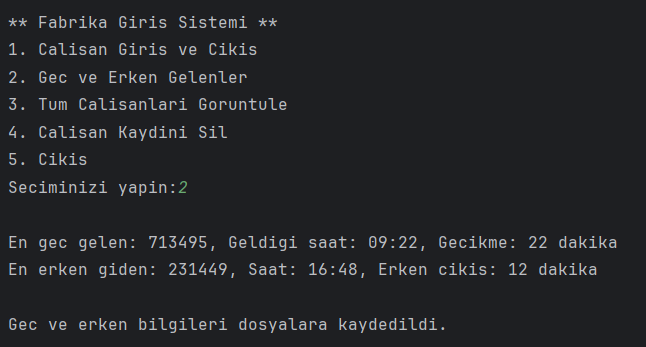
// "gunluk.txt" dosyasındaki verileri okur ve en geç/en erken gelenleri bulur  
void gec\_ve\_erken\_bul() {  
 FILE \*dosya = fopen("gunluk.txt", "r"); // Dosya okuma modunda açılır  
 if (dosya == NULL) {  
 printf("gunluk.txt dosyasi bulunamadi!\n"); // Dosya açılamazsa hata mesajı  
 return;  
 }  
  
 char satir[MAX\_SATIR]; // Dosyadan okunan satırlar için dizi  
 char en\_gec\_sicil[20], en\_erken\_sicil[20]; // En geç ve en erken gelenlerin sicil numaraları  
 int en\_gec\_saat = 9, en\_gec\_dakika = 0, gecikme = 0; // En geç gelme saati ve dakika  
 int en\_erken\_saat = 17, en\_erken\_dakika = 0, erken\_cikis = 0; // En erken çıkış saati ve dakika  
 bool gecikme\_durumu = false, erken\_bulundu = false; // Geç kalma durumu ve erken çıkış durumu  
  
 while (fgets(satir, sizeof(satir), dosya) != NULL) { // Dosyanın sonuna kadar satır satır okunur  
 char sicil\_no[20];  
 int saat, dakika;  
 char durum;  
 satir[strcspn(satir, "\n")] = 0; // Satır sonundaki newline karakteri temizlenir  
 if (sscanf(satir, "%s %d:%d %c", sicil\_no, &saat, &dakika, &durum) == 4) { // Satırdaki bilgiler okunur  
 if (durum == '<') { // Giriş durumu kontrol edilir  
 int fark = saat\_dakika\_fark(9, 0, saat, dakika); // Giriş saati ile 9:00 arasındaki fark bulunur  
 if (fark > gecikme) { // En geç geleni belirler  
 gecikme = fark;  
 strcpy(en\_gec\_sicil, sicil\_no);  
 en\_gec\_saat = saat;  
 en\_gec\_dakika = dakika;  
 gecikme\_durumu = true; // Geç kalma durumu true yapılır  
 }  
 } else if (durum == '>') { // Çıkış durumu kontrol edilir  
 int fark = saat\_dakika\_fark(saat, dakika, 17, 0); // Çıkış saati ile 17:00 arasındaki fark bulunur  
 if (fark > erken\_cikis) { // En erken çıkanı belirler  
 erken\_cikis = fark;  
 strcpy(en\_erken\_sicil, sicil\_no);  
 en\_erken\_saat = saat;  
 en\_erken\_dakika = dakika;  
 erken\_bulundu = true; // Erken çıkış durumu true yapılır  
 }  
 }  
 } else {  
 printf("Hatali satir formati: %s\n", satir); // Hatalı satır formatı varsa yazdırır  
 }  
 }  
 fclose(dosya); // Dosya kapatılır  
  
 // Geç kalan bilgileri "gec.txt" dosyasına yazılır  
 gec\_ve\_erken\_dosyaya\_yaz("gec.txt", "\nEn gec gelen: %s, Geldigi saat: %02d:%02d, Gecikme: %d dakika\n", gecikme\_durumu, en\_gec\_sicil, en\_gec\_saat, en\_gec\_dakika, gecikme);  
  
 // Erken çıkan bilgileri "erken.txt" dosyasına yazılır  
 gec\_ve\_erken\_dosyaya\_yaz("erken.txt", "\nEn erken giden: %s, Saat: %02d:%02d, Erken cikis: %d dakika\n", erken\_bulundu, en\_erken\_sicil, en\_erken\_saat, en\_erken\_dakika, erken\_cikis);  
  
 printf("\nGec ve erken bilgileri dosyalara kaydedildi.\n");  
}

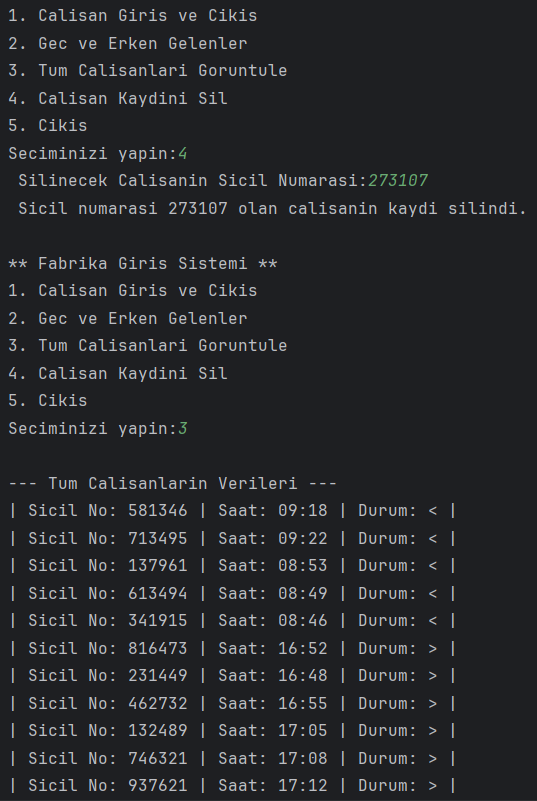
// İki saat arasındaki farkı dakika cinsinden hesaplar  
int saat\_dakika\_fark(int saat1, int dakika1, int saat2, int dakika2) {  
 return (saat2 \* 60 + dakika2) - (saat1 \* 60 + dakika1); // Saatleri ve dakikaları dakika cinsine çevirerek farkı bulur  
}  
  
// Tüm çalışanların verilerini "gunluk.txt" dosyasından okuyup ekrana yazdırır  
void tum\_calisanlari\_goruntule() {  
 FILE \*dosya = fopen("gunluk.txt", "r"); // Dosya okuma modunda açılır  
 if (dosya == NULL) {  
 printf("gunluk.txt dosyasi bulunamadi!\n"); // Dosya açılamazsa hata mesajı  
 return;  
 }  
  
 printf("\n--- Tum Calisanlarin Verileri ---\n");  
  
 char sicil\_no[20];  
 int saat, dakika;  
 char durum;  
  
 // Dosyadaki her satırı okuyup ekrana yazdırır  
 while (fscanf(dosya, "%s %d:%d %c", sicil\_no, &saat, &dakika, &durum) != EOF) {  
 printf("| Sicil No: %s | Saat: %02d:%02d | Durum: %c |\n", sicil\_no, saat, dakika, durum);  
 }  
  
 fclose(dosya); // Dosya kapatılır  
 printf("\n");  
}  
  
// Değişken sayıda argüman alarak belirtilen dosyaya yazar  
void dosyaya\_yaz(const char \*dosya\_adi, const char \*format, ...) {  
 FILE \*dosya = fopen(dosya\_adi, "a"); // Dosya ekleme modunda açılır  
 if (dosya == NULL) {  
 printf("%s dosyasi olusturulamadi!\n", dosya\_adi); // Dosya açılamazsa hata mesajı  
 return;  
 }  
  
 va\_list args;  
 va\_start(args, format); // Değişken argümanları al  
 vfprintf(dosya, format, args); // Dosyaya yaz  
 va\_end(args); // Değişken argüman listesini temizle  
  
 fclose(dosya); // Dosya kapatılır  
}  
  
// Geç ve erken bilgileri ilgili dosyaya yazar ve ekrana yazdırır  
void gec\_ve\_erken\_dosyaya\_yaz(const char \*dosya\_adi, const char \*metin, bool bulundu, const char \*sicil, int saat, int dakika, int fark) {  
 FILE \*dosya = fopen(dosya\_adi, "w"); // Dosya yazma modunda açılır  
 if (dosya == NULL) {  
 printf("%s dosyasi olusturulamadi!\n", dosya\_adi); // Dosya açılamazsa hata mesajı  
 return;  
 }  
 if (bulundu) { // Geç veya erken gelen bulunduysa  
 fprintf(dosya, metin, sicil, saat, dakika, fark); // Dosyaya yaz  
 printf(metin, sicil, saat, dakika, fark); // Ekrana yaz  
 } else { // Geç veya erken gelen bulunamadıysa  
 fprintf(dosya, "%s bulunamadi\n", dosya\_adi == "gec.txt" ? "Gec kalan" : "Erken cikan"); // Dosyaya yaz  
 printf("%s bulunamadi\n", dosya\_adi == "gec.txt" ? "Gec kalan" : "Erken cikan"); // Ekrana yaz  
 }  
 fclose(dosya); // Dosya kapatılır  
}

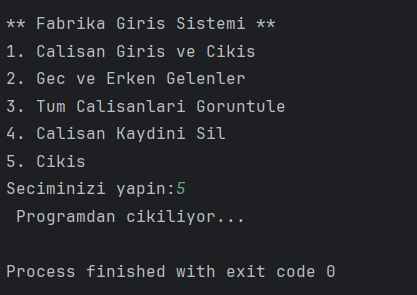
// Kullanıcıdan silinecek sicil numarasını alıp silme işlemini başlatır  
void silinecek\_sicil\_al\_ve\_sil() {  
 char silinecek\_sicil[20];  
 printf("Silinecek Calisanin Sicil Numarasi: ");  
 fgets(silinecek\_sicil, sizeof(silinecek\_sicil), stdin); // Sicil numarası kullanıcıdan alınır  
 silinecek\_sicil[strcspn(silinecek\_sicil, "\n")] = 0; // Satır sonundaki newline karakteri temizlenir  
 calisan\_kaydini\_sil(silinecek\_sicil); // Silme fonksiyonuna sicil numarası gönderilir  
}  
// Belirtilen sicil numarasına sahip çalışanın kaydını "gunluk.txt" dosyasından siler  
void calisan\_kaydini\_sil(const char \*silinecek\_sicil) {  
 FILE \*dosya = fopen("gunluk.txt", "r"); // Dosya okuma modunda açılır  
 if (dosya == NULL) {  
 printf("gunluk.txt dosyasi bulunamadi!\n"); // Dosya açılamazsa hata mesajı  
 return;  
 }  
 FILE \*gecici\_dosya = fopen("gecici.txt", "w"); // Geçici dosya yazma modunda açılır  
 if (gecici\_dosya == NULL) {  
 printf("gecici.txt dosyasi olusturulamadi!\n"); // Geçici dosya açılamazsa hata mesajı  
 fclose(dosya);  
 return;  
 }  
 char sicil\_no[20];  
 int saat, dakika;  
 char durum;  
 bool kayit\_silindi = false; // Kayıt silindi mi bilgisini tutar  
  
 // Dosyanın sonuna kadar satır satır okunur  
 while (fscanf(dosya, "%s %d:%d %c", sicil\_no, &saat, &dakika, &durum) != EOF) {  
 if (strcmp(sicil\_no, silinecek\_sicil) != 0) { // Okunan sicil numarası silinecek numaraya eşit değilse  
 fprintf(gecici\_dosya, "%s %02d:%02d %c\n", sicil\_no, saat, dakika, durum); // Geçici dosyaya yaz  
 } else { // Silinecek sicil numarası bulunduysa  
 kayit\_silindi = true; // Kayıt silindi olarak işaretlenir  
 }  
 }  
  
 fclose(dosya); // Dosya kapatılır  
 fclose(gecici\_dosya); // Geçici dosya kapatılır  
  
 if (remove("gunluk.txt") != 0) { // Orijinal dosya silinmeye çalışılır  
 printf("gunluk.txt dosyasi silinemedi!\n"); // Dosya silinemezse hata mesajı  
 remove("gecici.txt"); // Geçici dosya silinir  
 return;  
 }  
  
 if (rename("gecici.txt", "gunluk.txt") != 0) { // Geçici dosya orijinal isimle yeniden adlandırılmaya çalışılır  
 printf("gecici.txt dosyasi yeniden adlandirilamadi!\n"); // Dosya yeniden adlandırılamazsa hata mesajı  
 remove("gecici.txt"); // Geçici dosya silinir  
 return;  
 }  
  
 if(kayit\_silindi){  
 printf("Sicil numarasi %s olan calisanin kaydi silindi.\n",silinecek\_sicil); // Kayıt silindi mesajı  
 }  
 else{  
 printf("Sicil numarasi %s olan calisan kaydi bulunamadi.\n",silinecek\_sicil); // Kayıt bulunamadı mesajı  
 }  
}

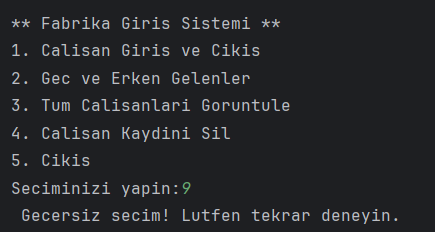
**Ekran Çıktıları:**











**Sonuç:**  
Çalışanların günlük giriş-çıkış verilerini yönetmek ve analiz etmek için gerekli tüm işlevleri başarıyla gerçekleştirdik. Kullanıcılardan aldığımız girdilere dayanarak geç ve erken gelen çalışanları belirleyip dosyalar halinde sakladık. Ayrıca, kullanıcı dostu bir arayüz sunarak çalışan yönetimini kolaylaştırdık.