**DİKKAT EDİLECEK HUSUSLAR**

1. Son ödeviniz dönem projesi grup çalışması şeklinde yürütülecektir. Grup üyeleri en az 2, en çok 3 kişi olacaktır.
2. Proje konuları ders kitabının DOSYALAR Konusu -Konu Sonu Çalışma Sorularından seçilecektir. Grubun sorumlu olduğu proje **Soru numarası=Grup-numaranız%55** şeklinde formülize edilmiştir. Bu yöntemle grubunuza ait proje konunuzu belirleyiniz.

*(Grup numarası tek sayı olan gruplar --- (1015+grup numarası)%55*

*(Örneğin ;5.grup için--- 1020%55=30.soru)*

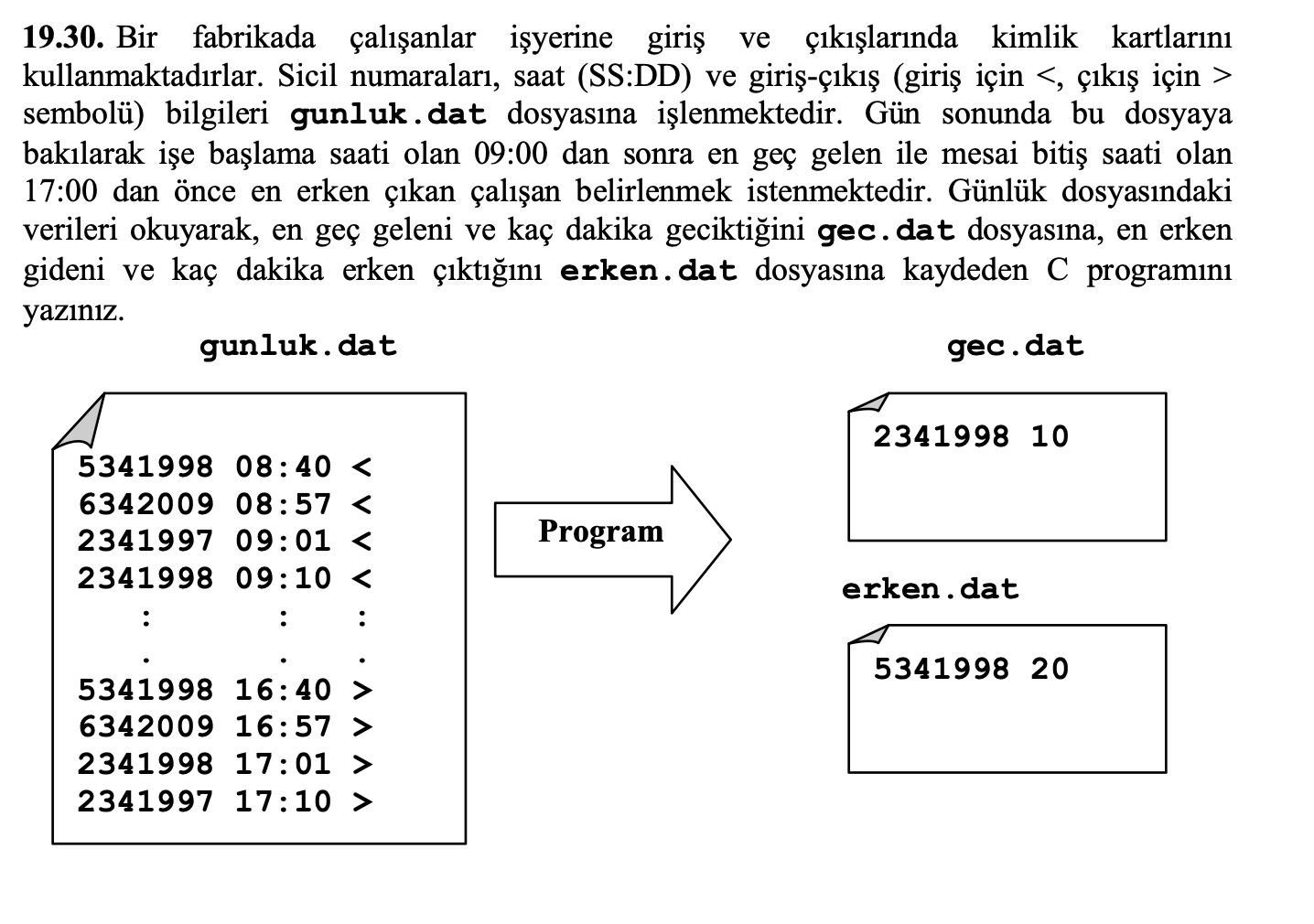
*Grup numarası çift sayı olan gruplar --- (2000+grup numarası)%55*

*(Örneğin ;8.grup için--- 2008%55=28.soru))*

1. Proje konunuzun belirlenmesinden sonra bir toplantı yaparak aranızda görev dağılımı yapınız.
2. Grup üyeleri olarak en az bir toplantı düzenleyerek, bu toplantılardan en az birini raporlayınız. **Kayıt altına aldığınız herhangi bir toplantının raporu 1 sayfayı geçmemelidir.** Proje ekibindeki katılımcılar projedeki görevlerini açıklamalı ve toplantının tarih ve günüde rapora eklenmelidir.
3. Grup projesi tamamlandıktan sonra ilgili ders saatinde proje sunumu için proje ekibinin **tamamı katılmalıdır.** Sunumda, proje ekibi üyelerinden proje sorusunu, akış şemasını ve C kodlarını kısaca açıklaması istenmektedir. Ekibin görev dağılımı proje raporunda belirtilmelidir.
4. Proje raporunuzun içeriği: ***Proje Konusu, Akış Şeması, C Program kodları, Ekran çıktıları*** olmalıdır.
5. Proje dosyalarınızın github ortamına yüklenmiş olması gerekmektedir. Github linki proje raporuna eklenmelidir.
6. Raporunuzun ismi: ***şubeno***-***grupnumaranız-proje*** olmalıdır. Proje raporunuzu .doc yada .docx formatında **Google Classroom’a** yükleme (upload) yapınız.
7. Son teslim tarihinden sonraki teslimler değerlendirilmeyecektir.
8. Son Teslim Tarihi ve saati : 20/12/2024 Cuma saat 23:59

Algoritma ve Programlamaya Giriş Dersi

Dönem Sonu Projesi



Şube No: 2

Grup Numarası : 10

Grup Üyeleri : Yusuf Kuru, Rıza Maden, Emirhan Fırtına

Github: https://github.com/rizamaden/Giris-Cikis-Kontrol

Toplantı Raporu:

**Toplantı Tarihi ve Günü:** 17 Aralık 2024, Salı  
**Toplantı Süresi:** 21:00 - 23:00  
**Toplantı Yeri:** Çevrimiçi (Zoom)  
**Katılımcılar: Emirhan Fırtına, Rıza Maden, Yusuf Kuru**

#### ****Toplantı Amacı:****

Proje kapsamında üstlenilen görevlerin son durumunu değerlendirmek, geri kalan işleri planlamak ve projenin zamanında tamamlanmasını sağlamak.

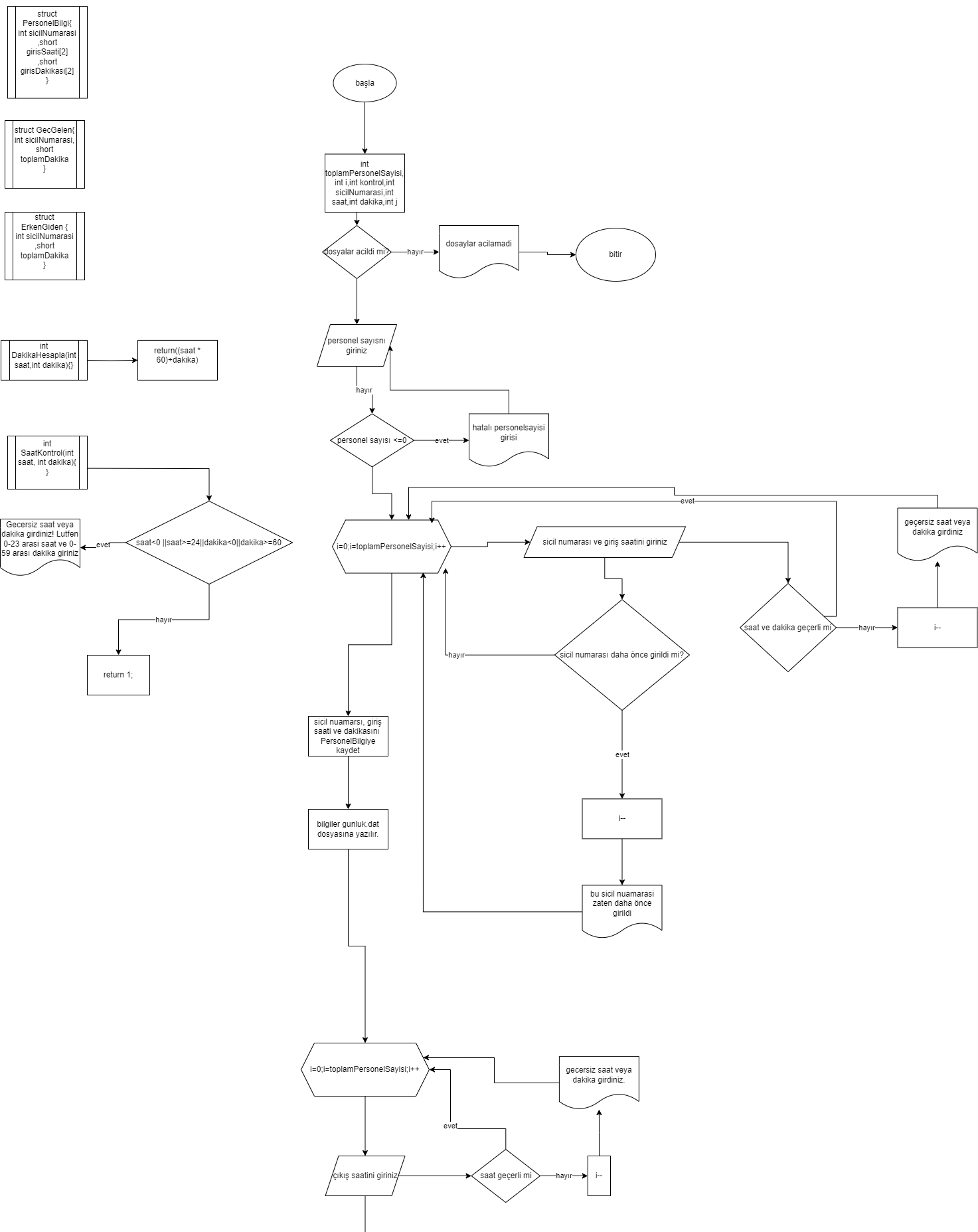
### Görevler:

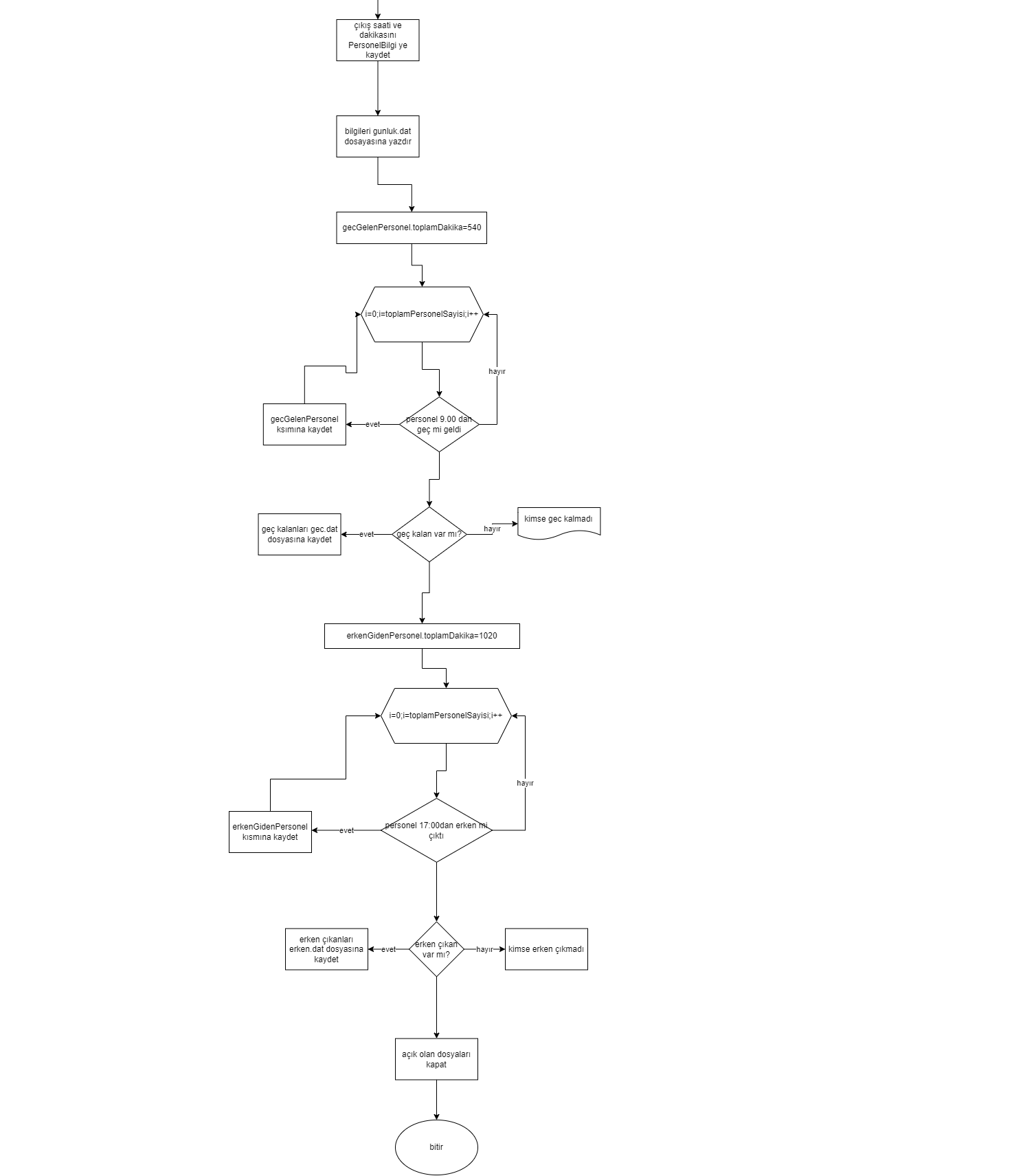
1. **Emirhan Fırtına ve Rıza Maden:**
   * Kod yazımı ve testleri tamamladılar. Kodun çalışmasını sağlayarak proje için gereken tüm işleri bitirdiler.
2. **Yusuf Kuru:**
   * Akış şemasını hazırladı ve ekipten onay aldı. Raporu birleştirme ve teslim etme işlerini planladı.

#### ****Toplantı Notları:****

* Proje ilerleyişinde bir aksaklık bulunmadığı gözlemlendi.
* Kod yazımı ve akış şeması gibi teknik işler başarıyla tamamlandı.
* Raporun son hali 20 Aralık 2024'te teslim edilecek.

Proje Akış Şeması:





Program Kodu:

#include <stdio.h>  
#include <stdlib.h>  
  
 struct personelBilgi { // İşe gelen personellerin bilgilerini tutan struct  
 int sicilNumarasi; // Sicil numarasi tutan degisken  
 int saat[2]; // Saati tutan degisken 0 numaralı indis giris saati 1 numaralı indis cikis saati  
 int dakika[2]; // Dakika tutan degisken 0 numaralı indis giris dakikasi 1 numarali indis cikis dakikasi  
 };  
  
 struct GecGelen { // Gec gelen personelin bilgilerini tutan struct  
 int sicilNumarasi; // Gec gelen personelin sicil numarasini tutan degisken  
 int gecikmeDakikasi; // Gec gelen personelin saat 9:00'dan itibaren kac dakika geciktigini tutan degisken  
 };  
  
 struct ErkenGiden { // Erken gelen personelin bilgilerini tutan struct  
 int sicilNumarasi; // Erken gelen personelin sicil numarasini tutan degisken  
 int erkenCikmaDakikasi; // Erken gelen personelin 17:00'dan kac dakika once gittigini tutan degisken  
 };  
  
 int saatDakikayaCevir(int saat, int dakika) { // Bu projede yapilacak karsilastirmalar icin ihtiyacimiz olan dakika verisini bu fonksiyon ile elde ederiz  
 return saat \* 60 + dakika;  
 }  
  
  
 int saatkontrol(int x,int y) { // Bu projede yapilacak karsilastirmalar icin ihtiyacimiz olan dakika verisini bu fonksiyon ile elde ederiz  
 if(x>23 || x<0 || y>59 || y<0) { // Gecersiz saat girilirse fonksiyon 1 eğer gecerli saat girilirse 0 dondurur  
 return 1;  
 }  
 else {  
 return 0;  
 }  
 }  
  
 int main() {  
 FILE \*gunlukdosya, \*erkendosya, \*gecdosya; //Dosya isaretcilerinin tanimlanmasi  
 int toplamPersonel; //Kac kisinin ise gelecegini tutan degisken  
  
 printf("Bugun ise gelecek toplam personel sayisini giriniz: ");  
  
  
 do { // Gecersiz personel sayisi girisini engelleyen dongu  
 scanf("%d", &toplamPersonel);  
 if (toplamPersonel<=0) printf("Hatali personelsayisi girisi\n");  
  
 }while (toplamPersonel <= 0);  
  
  
 struct personelBilgi personel[toplamPersonel]; // Tum personellerin bilgisini tutan yapi  
  
 gunlukdosya = fopen("gunluk.dat", "w"); // Girilen sicil numaralari saatler bu dosyaya yazilacaktir.  
  
  
 if (gunlukdosya == NULL) { //Dosya acilamadiysa kullanıcıya bildiri veren ve programi sonlandiran kosul  
 printf("gunluk.dat dosyasi acilamadi.\n");  
 return 1;  
 }  
  
 for (int i = 0; i < toplamPersonel; i++) { // Kullanicidan ilk olarak Sicil No,Giris saatlerini aldiran ve dosyaya yazdiran dongu  
  
 int kontrol=0; // Ayni sicil numarasi girilmesin diye olusturlan degisken  
  
 printf("%d. personel icin sicil numarasi ve giris saati (SS:DD) girin: ", i + 1);  
 scanf("%d %d:%d", &personel[i].sicilNumarasi,&personel[i].saat[0],&personel[i].dakika[0]); //Kullanicidan verilerin alinmasi;  
  
  
 for (int j = 0; j < i; j++) { //Eger sicil numarasi daha onceden girildiyse kontrol degiskenini 1'e atayan dongu  
 if (personel[j].sicilNumarasi == personel[i].sicilNumarasi) {  
 kontrol=1;  
 break;  
 }  
 }  
  
  
 if (kontrol) { //Kontrol degeri 1 ise verilerin tekrar girilmesi istenecektir  
 printf("Bu sicil numarasi zaten daha once girildi.Lutfen tekrar giris yapiniz\n");  
 i--;  
 continue;  
 }  
 if (saatkontrol(personel[i].saat[0],personel[i].dakika[0])) { //Saat gecersiz bir sekilde girildiyse verilerin tekrar girilmesi istenecektir  
 printf("Hatali saat girisi.Lutfen tekrar giriniz\n");  
 i--;  
 continue;  
 }  
 fprintf(gunlukdosya, "%d %02d:%02d <\n", personel[i].sicilNumarasi,personel[i].saat[0],personel[i].dakika[0]); //Girilen degerler kontrol dongulerinden gecebildiyse  
 //gunlukdosya'ya yazdirilir  
 }  
  
 for (int i = 0; i < toplamPersonel; i++) { // Kullanicidan onceden alinan Sicil nolara gore gore cikis saatlerini aldiran dongu  
 printf("%d sicil numarali personelin cikis saatini giriniz\n",personel[i].sicilNumarasi);  
 scanf("%d:%d", &personel[i].saat[1],&personel[i].dakika[1]); //Kullanicidan cikis verilerinin aldirilmasi  
  
  
 if (saatkontrol(personel[i].saat[1],personel[i].dakika[1])) { //Saat gecersiz bir sekilde girildiyse verilerin tekrar girilmesi istenecektir  
 printf("Hatali saat girisi.Lutfen tekrar giriniz\n");  
 i--;  
 continue;  
 }  
 fprintf(gunlukdosya, "%d %02d:%02d >\n",personel[i].sicilNumarasi,personel[i].saat[1],personel[i].dakika[1]); //Girilen degerler kontrol dongusunden gecebildiyse  
 //gunlukdosya'ya yazdirilir  
 }  
  
 fclose(gunlukdosya); //gunlukdosya kapatilir  
 printf("Günlük bilgiler gunluk.dat dosyasına yazıldı.\n");  
  
  
 gunlukdosya = fopen("gunluk.dat", "r"); // gunlukdosya erkendosya ve gecdosya ya veri yazdırma amaciyla  
 // okuma modunda acilir  
 if (gunlukdosya == NULL) {  
 printf("gunluk.dat dosyası acilamadi.\n");  
 return 0;  
 }  
  
 gecdosya = fopen("gec.dat", "w"); // gec gelenlerin kaydedileceği dosyayı yazma modunda actik  
 erkendosya = fopen("erken.dat", "w"); // erken gidenlerin kaydedileceği dosyayı yazma modunda actik  
 if (gecdosya == NULL || erkendosya == NULL) { // olası hatalar icin mesaj  
 printf("Sonuc dosyalari acilamadi.\n");  
 fclose(gunlukdosya);  
 return 0;  
 }  
  
  
 struct GecGelen gecGelenPersonel ; // gec gelenlerin kaydedileceği struct  
 struct ErkenGiden erkenGidenPersonel; // erken gidenlerin kaydedileceği struct  
  
 int sicilNumarasi, saat, dakika; // karsilastirma islemleri icin kullanilicak degiskenlerin tanimlanmasi  
 char islem;  
  
 gecGelenPersonel.gecikmeDakikasi=0; // gec gelen icin gecikme dakikasini tutan degisken  
 erkenGidenPersonel.erkenCikmaDakikasi=0; // erken giden icin erken gitme dakikasini tutan degisken  
  
 while (fscanf(gunlukdosya, "%d %d:%d %c", &sicilNumarasi, &saat, &dakika, &islem) != EOF) { // gunluk.dat dosyasinin satir satir okunudugu ve verilerin  
 int zamanDakika = saatDakikayaCevir(saat, dakika); // ilgili degiskene kaydedilgini dongu  
  
 if (islem == '<') { // Giriş işlemi karsilastirmaları  
 if (zamanDakika > 540 && (zamanDakika-540) > gecGelenPersonel.gecikmeDakikasi) { // dongunun o anına kadarki maksimum gecikeni kaydeder  
 gecGelenPersonel.sicilNumarasi = sicilNumarasi;  
 gecGelenPersonel.gecikmeDakikasi = zamanDakika - 540;  
 }  
 } else if (islem == '>') { // Çıkış işlemi karsılastırmaları  
 if (zamanDakika < 1020 && (1020 - zamanDakika) > erkenGidenPersonel.erkenCikmaDakikasi) { // dongunun o anına kadarki en erken cikani kaydeder  
 erkenGidenPersonel.sicilNumarasi = sicilNumarasi;  
 erkenGidenPersonel.erkenCikmaDakikasi = 1020 - zamanDakika;  
 }  
 }  
 }  
  
 // Gecikme ve erken çıkma bilgilerini durumlara göre dosyalara yaz  
 if (gecGelenPersonel.gecikmeDakikasi == 0) { // kimsenin gec kalmadigi durum  
 fprintf(gecdosya,"Kimse gec kalmadi\n");  
 }  
 else{  
 fprintf(gecdosya, "%d %d\n", gecGelenPersonel.sicilNumarasi, gecGelenPersonel.gecikmeDakikasi);  
 }  
  
 if (erkenGidenPersonel.erkenCikmaDakikasi == 0) { // kimsenin erken cikmadigi durum  
 fprintf(erkendosya,"Kimse erken cikmadi\n");  
 }  
 else{  
 fprintf(erkendosya, "%d %d \n", erkenGidenPersonel.sicilNumarasi, erkenGidenPersonel.erkenCikmaDakikasi);  
 }  
  
 printf("Sonuclar gec.dat ve erken.dat dosyalarina yazildi.\n");  
  
 fclose(gunlukdosya); // dosya kapatma islemleri  
 fclose(gecdosya);  
 fclose(erkendosya);  
  
 return 0;  
 }

Ekran Çıktıları:

