Ad:Abdul Jalal
Soyad:Lawal
Öğrenci No:200061082

Tek Yönlü Bağlı Liste Kullanılarak Telefon Rehberi Uygulamasi

❖ Uygulama on tane fonksiyondan yazildi

- Main() Fonksiyonu()
- Menu Seçime() Fonksiyonu()
- Düğüm Oluşturma Fonksiyonu()
- Bilgi Alma Fonksiyonu()
- Rehber Yazdirma Fonksiyonu()
- Rehber Arama Fonksiyonu()
- Kişi Güncelleme Fonksiyonu()
- Rehber Ekleme Fonksiyonu()
- Rehber Silme Fonksiyonu()
- Rehber Sıralama Fonksiyonu()

❖ Menu Seçime Fonksiyonu

- Fonksiyonun amaci rehberi kullanacak kişiyi doğru yönlendirmek görevine üstlenir
- Fonksiyon herhangi parametre almiyor
- İnt tipinde değişken donduruyor
- Dondurduğu değerle kişinin hangi işlem yapicağı belirlenir

```
int menu_secimi() {
    int secim;
    printf("\n\t\t\t\t   MENU\n\n");    //
    printf("\t\t\t ------------------------ \n");//
    printf("\t\t\t| 1. Yeni Kisi Ekle       |\n");//
    printf("\t\t\t| 2. Kisi Guncelle        |\n");//
    printf("\t\t\t| 3. Rehberi Goruntule    |\n");//
```

```
    printf("\t\t\t| 4. Kisi Sil              |\n");//
    printf("\t\t\t| 5. Kisi Arama yap        |\n");//
    printf("\t\t\t| 6. Cikis                 |\n");//
    printf("\t\t\t ------------------------ \n");//
    printf("\t\t\t Tercihinizi Giriniz: ");
    scanf_s("%d", &secim);
    return secim;
}
```

❖ Düğüm Oluşturma Fonksiyonu()

- Struct ile oluşturulan düğümün veri tipleri parametre olarak alir
- bliste veri tipinde yeni doğum işaraçeşine bliste tipine göre yer tasvir edilir
- doğumun bir sonraki adresine NULL ile eşitlenir bu şekilde boş bir dugum oluşturulur
- dönüş değerin de oluşturulan yeni düğümün adresidir yani düğümün başi

```
bliste *dugumolustur(char ad[60],char soy_ad[60],char
tel_no[12],char email[12]){
    bliste *yenidugum = (bliste*)malloc(sizeof(bliste));
    strcpy(yenidugum -> ad, ad);
    strcpy(yenidugum -> soy_ad, soy_ad);
    strcpy( yenidugum -> tel_no, tel_no);
    strcpy( yenidugum -> email, email);
    yenidugum -> next = NULL;
    return yenidugum;
}
```

❖ Bilgi Alma Fonksiyonu()

- Fonksiyonun amaci rehber eklemek için klavye yoluyla alinan bilgileri oluşturan boş düğünün içine doldurulur

```
void bilgi_alma(){
printf("Kaydetmek istediniz rehberin adini giriniz: ");
 scanf("%s",ad);
 printf("Kaydetmek istediniz rehberin Soyadini giriniz: ");
 scanf("%s", soy_ad);
 printf("Kaydetmek istediniz rehberin Telefone giriniz: ");
 scanf("%s", tel_no);
 printf("Kaydetmek istediniz rehberin Email addresini giriniz:" );
```

```
    scanf("%s", email);
}
```

❖ Rehber Yazdirma Fonksiyonu()

- Rehbere kaydedilen kişileri görüntülemek amacıyla tasarlandı
- Global değişkeni olarak tanımlanan head işaretçisini gezerek düğümdeki diğer rehberlerde ulaşıp bu işlemi gerçekleştirir,gezinirken doğumun bir sonraki adresi NULL olmamasi şartıyla rehberi gezer ki zaten NULL ise son düğüm olduğu anlamına gelir ve gezinmeyi sonlandirir

```
void rehber_yazdir(){
    rehber_siralama();
    bliste *temp = head;
    while(temp != NULL){
        printf("%s\t%s\t%s\t%s\n",temp -> ad,temp -> soy_ad,temp ->
tel_no,temp -> email);
        temp = temp -> next;
    }
    printf("\n\n");
}
```

❖ Rehber Arama Fonksiyonu()

- Fonksiyonun amaci rehberde daha önce kaydedilmiş olan rehberi aramak
- Klavye yoluyla alınan kişiyi rehbere aratarak gerçekleştirir
- Armayi rehberin boş olmama şartıyla gerçekleştirilir
- Klavye yoluyla girilen kişi ad yada soyadini rehberi aranan kişiyi eşleşene kadar gezinme işlemi gerçekleştirilir eğer girilen kişi rehberde eşleşirse kişinin bilgileri ekrana yazdırıp programi sonlandirilir aksi

takdirde kişinin rehberde kayıtlı olmadığı bilgisini görüntülenir .

```
void rehber_arama(){
    printf("Rehberde aradiniz kisinin ismini giriniz");
    scanf("%s", aranan);
    bliste *temp = head;
    char isim[20],soyisim[20];
    while(temp != NULL){
        if(!strcmp(temp -> ad , aranan) || !strcmp (temp ->
soy_ad,aranan )){
            strcpy(temp -> ad , isim);
            strcpy(temp -> soy_ad , soyisim);
            printf("Aranilan %s %s Rehber bulundu \n", isim,soyisim);
            durum = 1;
            break;
        }
        temp = temp -> next;
    }
    if(durum == 0){
        printf("Aranilan %s isim listede yok \n", aranan);
    }
}
```

❖ Kişi Güncelleme Fonksiyonu()

- Fonksiyonun amaci rehberde kayıtlı kişilerin bilgileri değiştirmek
- Fonksiyonun çalışma mantığı arama fonksiyonuna benzer,öncelikle rehberi gezip güncellenmek istenilen kişi ile eşlenip kullanıcıdan güncel bilgileri alip kişi bilgilerini güncellenir ve yazdirma fonksiyonu çağırarak rehberin güncel halini ekrana yazdırır aksi takdirde kişinin rehberde olmadığının bilgisi ekranda görüntülenir.

```
void kisi_guncele(){
    if (head == NULL){
            printf("Rehberde Kimse Kayitli olmadiği için Günceleme
işlemi yapilmaz\n");
    } else if (head != NULL){
        char guncel_ad[60], guncel_soy_ad[60], guncel_tel_no[12],
guncel_email[12];
```

```
        printf("Rehberde güncelemek istediniz kisinin ismini yada soy
ismini giriniz");
        scanf("%s", aranan);
        bliste *temp = head;
        char isim[60],soyisim[60];
        while(temp != NULL){
            if(!strcmp(temp -> ad , aranan) || !strcmp (temp ->
soy_ad,aranan )){
                strcpy(temp -> ad , isim);
                strcpy(temp -> soy_ad , soyisim);
                printf("\n");
                printf("Güncelemek istediniz  istediniz  kişinin güncel
adini giriniz: \n");
                scanf("%s",guncel_ad);
                printf("Güncelemek istediniz  istediniz  kişinin güncel
soy adini giriniz: \n");
                scanf("%s",guncel_soy_ad);
                printf("Güncelemek istediniz  istediniz  kişinin güncel
telefone numarasi giriniz: \n");
                scanf("%s",guncel_tel_no);
                printf("Güncelemek istediniz  istediniz  kişinin güncel
email addresini giriniz: \n");
                scanf("%s",guncel_email);
                strcpy(temp -> ad, guncel_ad);
                strcpy(temp -> soy_ad, guncel_soy_ad);
                strcpy(temp -> tel_no, guncel_tel_no);
                strcpy(temp -> email, guncel_email);
                rehber_yazdir();
                durum = 1;
                break;
            }
            temp = temp -> next;
        }
        if(durum == 0){
            printf("Aranilan %s isim listede yok \n", aranan);
        }
    }
}
```

- ❖ Rehber Ekleme Fonksiyonu()
  - Fonksiyonun amaci yeni kişiyi rehbere eklemek

- Öncelikle eklenecek olan yeni kişinin bilgileri Bilgi Alma Fonksiyonu() çağırarak alınır ve düğümün başına ekleyip Sıralama fonksiyonu () çağırarak rehberi alfabetik sirasina göre tüm rehberi siralanir.

```
void rehber_ekle(){
    void bilgi_alma();
    bliste *eklenecek = dugumolustur( ad, soy_ad,tel_no,email);
    eklenecek -> next = head ;
    head = eklenecek;
    rehber_siralama();
}
```

❖ Rehber Silme Fonksiyonu()

- Fonksiyonun amaci rehberde kayıtlı kişiyi silmek
- Öncelikle rehberden silinmek istenilen kişi bilgileri alinir
- Klavyeden girilen kişinin isminin önce düğümün başında olup olmadigini kontrol edilir eğer baş ise baş işaretçisinin adresini kaybetmemek için geçiçi değişken içine koyup yeni baş değişkeni bir sonraki düğümü olarak eşlenip geçiçi değişkeni sifirlanir ve kişinin başarılı bir şekilde listeden silindiğinin bilgisini verilrir
- Eğer girilen kişi duğumun başında değil ise duğumun bir sonraki adresi NULL olmadığı şartıyla düğümü gezip kişi rehberde kayıtlı ise silinir aksi takdirde rehberde öyle bir kişinin kayitli olmadığının bilgisini verilir.

```
void rehber_sil(){
    bliste *temp = head;
    bliste *temp1 = NULL;
    char silinecek_kisi[20];
    printf("Rehberde silmek istediniz kisinin ismini
giriniz: ");
    scanf("%s", silinecek_kisi);
    int position = 0, durum = 0;
    if (!strcmp(head -> ad , silinecek_kisi)){
        temp = head;
        head = head -> next;
        free(temp);
        durum = 1;
```

```
        printf(" %s isimli Rehberiniz Rehberden silindi \n",
silinecek_kisi);
    } else {
        while(temp -> next != NULL ){
            if(!strcmp(temp -> next  -> ad , silinecek_kisi)
|| !strcmp (temp -> next  ->  soy_ad,silinecek_kisi )){
                temp1 = temp -> next;
                temp -> next = temp -> next -> next;
                free(temp1);
                durum = 1;
                printf(" %s isimli Rehberiniz Rehberden
silindi \n", silinecek_kisi);
                break;
            }
            temp = temp -> next;
        }
        if(durum == 0){
            printf("Aranilan %s isim listede yok \n",
silinecek_kisi);
        }
    }
}
```

❖ Rehber Sıralama Fonksiyonu()

- Fonksiyonun amaci rehberi alfabetik bir şekilde sıralanmasıdır
- Rehberde en az iki tane kişi olmak şartıyla rehberi gezilir
- Birinci kişinin bilgileri ile bütün rehberi kiyaslanir bu şekilde küçükten büyüğe rehberdeki kişiler siralanir
- Sıralama işlemi kişinin isminin harfleri kiyaslayarak gerçekleştirildiği için isimlerden alfabe sayisinin az olan ismin boyutuyla sıralama işlemi gerçekleştirilir
- Rehberde ayni ismi olan kişilerin ihtimalini düşünerek soyadları ile sıralama işlemi gerçekleşir

```
void rehber_siralama(){
    bliste *temp = head;
    bliste *temp1 = NULL;
    char Stemp[30];
```

```
    int a,b,size,test;

    while(temp != NULL){
        temp1 = temp -> next;
        while (temp1 != NULL) {
            test = 0;
            size = strlen(temp -> ad) > strlen(temp1 -> ad) ?
strlen(temp1 -> ad) : strlen(temp -> ad);
            for(int i = 0; i < size; i++){
                a = temp -> ad[i];
                b = temp1 -> ad[i];
                if(a > b){
                    strcpy(Stemp , temp -> ad);
                    strcpy(temp -> ad , temp1 -> ad);
                    strcpy(temp1 -> ad , Stemp);

                    strcpy(Stemp , temp -> soy_ad);
                    strcpy(temp -> soy_ad , temp1 -> soy_ad);
                    strcpy(temp1 -> soy_ad , Stemp);

                    strcpy(Stemp , temp -> tel_no);
                    strcpy(temp -> tel_no , temp1 -> tel_no);
                    strcpy(temp1 -> tel_no , Stemp);

                    strcpy(Stemp , temp -> email);
                    strcpy(temp -> email , temp1 -> email); // aaaz
aaa
                    strcpy(temp1 -> email , Stemp);
                    test = 1;
                    break;
                } else if (a < b){
                    break;
                }

            }

            if(!test){
                if(strlen(temp -> ad) > strlen(temp1 -> ad)){
                    strcpy(Stemp , temp -> ad);
                    strcpy(temp -> ad , temp1 -> ad);
                    strcpy(temp1 -> ad , Stemp);

                    strcpy(Stemp , temp -> soy_ad);
```

```
                    strcpy(temp -> soy_ad , temp1 -> soy_ad);
                    strcpy(temp1 -> soy_ad , Stemp);

                    strcpy(Stemp , temp -> tel_no);
                    strcpy(temp -> tel_no , temp1 -> tel_no);
                    strcpy(temp1 -> tel_no , Stemp);

                    strcpy(Stemp , temp -> email);
                    strcpy(temp -> email , temp1 -> email); // aaaz
aaa
                    strcpy(temp1 -> email , Stemp);
                }
            }


            if(!strcmp(temp -> ad,  temp1 -> ad)){  //Rehberde ayni
isimi varsa eğer soyadina göre siralama yapar
                size = strlen(temp -> soy_ad) > strlen(temp1 -> soy_ad)
? strlen(temp1 -> soy_ad) : strlen(temp -> soy_ad);
                for(int i = 0; i < size; i++){
                    a = temp -> soy_ad[i];
                    b = temp1 -> soy_ad[i];

                    if(a > b){
                        strcpy(Stemp , temp -> ad);
                        strcpy(temp -> ad , temp1 -> ad);
                        strcpy(temp1 -> ad , Stemp);

                        strcpy(Stemp , temp -> soy_ad);
                        strcpy(temp -> soy_ad , temp1 -> soy_ad);
                        strcpy(temp1 -> soy_ad , Stemp);

                        strcpy(Stemp , temp -> tel_no);
                        strcpy(temp -> tel_no , temp1 -> tel_no);
                        strcpy(temp1 -> tel_no , Stemp);

                        strcpy(Stemp , temp -> email);
                        strcpy(temp -> email , temp1 -> email); // aaaz
aaa
                        strcpy(temp1 -> email , Stemp);
                        test = 1;
                        break;
                    } else if (a < b){
```

```
                        break;
                    }

                }
                if(!test){
                    if(strlen(temp -> soy_ad) > strlen(temp1 ->
soy_ad)){
                        strcpy(Stemp , temp -> ad);
                        strcpy(temp -> ad , temp1 -> ad);
                        strcpy(temp1 -> ad , Stemp);

                        strcpy(Stemp , temp -> soy_ad);
                        strcpy(temp -> soy_ad , temp1 -> soy_ad);
                        strcpy(temp1 -> soy_ad , Stemp);

                        strcpy(Stemp , temp -> tel_no);
                        strcpy(temp -> tel_no , temp1 -> tel_no);
                        strcpy(temp1 -> tel_no , Stemp);

                        strcpy(Stemp , temp -> email);
                        strcpy(temp -> email , temp1 -> email); // aaaz
aaa
                        strcpy(temp1 -> email , Stemp);

                    }
                }
            }

            temp1 = temp1 -> next;
        }
        temp =  temp -> next;
    }
}
```

- ❖ Main() Fonksiyonu()
  - Main fonksiyonunda yukarda tanimladim fonksiyonlari çağırarak menu kisminda işlemleri gerçek kullanıcının isteğine göre gerçekleşir

- Aynı zamanda kullanıcının girdiği bilgileri belli hata kontrolü gerçekleştiriliyor

```
int main(){
    system("color 1f");//Background rengini ve font rengini değiştirir.
    int secim, devam = 1;
    menu:
    secim = menu_secimi();
    if (secim == 1 || secim == 2 || secim == 3 || secim == 4 || secim
== 5 || secim == 6){
        while(secim != 6){
            switch (secim){ //Kullanıcıdan hangi işlemi istediği
arauldu
                case 1:
                    if(head == NULL){
                        bilgi_al:
                        bilgi_alma();
                        head =  dugumolustur( ad, soy_ad,tel_no,email);
                        printf("Rehbernize %s isimli kişi
kaydedildi\n",ad);
                    } else {
                        rehber_ekle();
                    }
                    secim = menu_secimi();
                    break;
                case 2:
                    kisi_guncele();
                    secim = menu_secimi();
                    break;
                case 3:
                    if (head == NULL){
                        int hata_kontrolu;
                        printf("Rehberde kimse Kayitli Değil\n");
                        printf("Rehberinize kişi eklemek için (1)
tiklayabilirsiniz\n\n");
                    } else {
                        rehber_yazdir();
                    }
                    secim = menu_secimi();
                    break;
                case 4:
                    if (head == NULL){
                        printf("Rehberinizde Kayitli Kisi Olmadiği için
Silme İşlemi Yapilmaz\n");
```

```
                        printf("Rehberinize kişi eklemek için (1)
tiklayabilirsiniz\n\n");
                    } else {
                        rehber_sil();
                    }
                    secim = menu_secimi();
                    break;
                default:
                    if (head == NULL){
                        printf("Rehberinizde Kayitli Kisi Olmadiği için
Arama İşlemi Yapilmaz\n");
                    } else {
                        rehber_arama();
                    }
                    secim = menu_secimi();
                    break;
            }
        }
    } else {
        printf("Yanlis secim yaptiniz!Tekrar Deneyiniz\n");
        goto menu;
    }

}
```