2

EGE UNIVERSITY

DERS : ALGORITMA VE PROGRAMLAMA 1
PROJE : 2

İçindekiler.....

Analiz:

Uluslararası Satranç Federasyonu (FIDE) tarafından turnuvanın sonunda.belirlenen İsviçre Sistemi eşleştirme kurallarına benzer kurallara göre her turda maç yapmak için bir program geliştirilmesi istenmektedir. bu program sayesinde başarı sıralaması ve çapraz tabloların görüntülenmesi istenir. öncelikle turnuvaya katılan her oyuncu kendi lisans numarası adını soyadını ve FİDE deki kuvvet puanın ayni şekilde UKD puanını girecek.

Herkesin başlangıç puanı 0 dır. Ve puan ELO UKD ad-soyad ve lisans no gibi belli kriterelere göre sıralanır.

tüm oyuncular başlangıçta sıralandıktan sonra 1den başlıyarak liste halinde her kişiye bir başlangıç numarası verilerek verilir.

oyuncalar İsveçre sisteme göre sıralanır. Bu sistemin sonucunda bütün eşleştirmeler bittikten sonra tabloda belirtilir.

Bir sonrakıi tür için maç sonuçlar 0-5 sayıları ile programa yazılır.

- *0: beraberlik, yani maç sonucu ½ - ½*
- *1: beyaz galip, yani maç sonucu 1 - 0*
- *2: siyah galip, yani maç sonucu 0 - 1*
- *3: siyah maça gelmemiş, yani maç sonucu + - -*
- *4: beyaz maça gelmemiş, yani maç sonucu - - +*
- *5: her iki oyuncu da maça gelmemiş, yani maç sonucu - - -*

Bu puanlara göre turnuva sonındaki puanlar, oyuncuların maçlarda aldıkları puanların toplanmasıyla belirlenir. Eğer turnuva sonunda eşitlik söz konusu olursa o zaman aşağıdaki eşitlik bozma ölçümü devreye girer.

Buchholz-1 alttan (BH-1) Buchholz-2 (BH-2) alttan Sonneborn Berger (SB) Galibiyet Saysı (GS) a göre hesaplanır. Bunun sonucunda Sıra numaraya göre son tablomuzu oluştururuz.

En sonda ise oyuncuların oynadığı maçları çapraz tablolarda başlangıç numaralarına göre sıralı olarak yazıyoruz.

.Tasarım:

KAYNAK KOD

```
from math import floor, ceil, log


def nihaiSonuc(puan_tablosu):
    # puan_tablosu : [ [BSNo, LNo, Ad-Soyad, ELO, UKD, Puan, BH1, BH2, SB, GS], ..]

    puan_tablosu.sort(key = lambda row : row[5:-1], reverse = True)  # Puan sistemlerine gore Descending sort.
    nihai_so = []   # [ [SNo, BSNo, LNo, Ad-Soyad, ELO, UKD, Puan, BH1, BH2, SB, GS], ..]

    print("\n\n\nNihai Sıralama Listesi: ")
    print("SNo BSNo   LNo Ad-Soyad      ELO  UKD Puan  BH-1  BH-2    SB GS")
    print("--- ---- ----- ------------ ---- ---- ---- ----- ----- ----- --")
    for i in range(1, len(puan_tablosu) + 1):
        oyuncu = puan_tablosu[i - 1]
        print("{:3} {:4} {:5} {:12} {:4} {:4} {:4} {:5} {:5} {:5} {:2}".format(i, oyuncu[0], oyuncu[1], oyuncu[2],
                oyuncu[3], oyuncu[4], oyuncu[5], oyuncu[6], oyuncu[7],  oyuncu[8], oyuncu[9]))

        nihai_so.append([i] + oyuncu)

    return nihai_so


def caprazTablo(nihai_netice, veriler, tur_sayisi):
    # nihai_netice : [ [SNo, BSNo, LNo, Ad-Soyad, ELO, UKD, Puan, BH1, BH2, SB, GS], ..]
    # Veriler : (asagidaki koment)
    # dict :  {  BSNo : [LNo, Ad-Soyad, Puan_Toplami, [rakipleri (LNo)], ELO, UKD, Oyuncunun Oynadigi Tum Turlar],  ..}

    print("\n\nÇapraz Tablo: ")
    print("BSNo SNo   LNo Ad-Soyad      ELO  UKD ", end="")
    for i in range(tur_sayisi):
        print(" {0}. Tur ".format(i + 1), end="")
    print("")
    print("---- --- ----- ------------ ---- ----", end='')
    for i in range(tur_sayisi):
        print(" -------", end='')
    print("")
    nihai_sonuc = nihai_netice[:]
    nihai_sonuc.sort(key = lambda row : row[1])     # Burda BSNo'ya gore siraliycaz.

    for bsno in range(1, len(nihai_sonuc) + 1):
        player = nihai_sonuc[bsno - 1]
        print("{:4} {:3} {:5} {:12} {:4} {:4}  ".format(bsno, player[0], player[2], player[3], player[4], player[5]), end='')
        veri = veriler[bsno]
        for t in range(6, len(veri)):
            tur = veri[t]   # [Rakibinin BSNo, Renk, Puan]
            string = str(tur[0]) + " " + tur[1]
            if tur[-1] == 0.5:
                string += " " + chr(189)
            else:
                string += " " + str(tur[-1])
            print(" {:7}".format(string), end="")
        print("")


def tumPuanlariHesapla(veriler):
    # Bu fonk tum verileri alip puanlari toplar, sonra liste seklinde return yapar.
    # Sonra bunun return yaptigi listeden Nihai Sonuc ve Son Sonucu Tutan Capraz Tablo olusturulur.

    # Veriler -> main'deki capraz_tablo'dur:
    # dict :  {  BSNo : [LNo, Ad-Soyad, Puan_Toplami, [rakipleri (LNo)], ELO, UKD, Oyuncunun Oynadigi Tum Turlar],  ..}
    sonuc = []  # [ [BSNo, LNo, Ad-Soyad, ELO, UKD, Puan, BH1, BH2, SB, GS], ..]

    for BSNo in sorted(veriler.keys()):
        _bh1_bh2 = _BH1_BH2_Hesapla(veriler, BSNo)
        sb = _SB_Hesapla(veriler, BSNo)
        gs = _GS_Hesapla(veriler, BSNo)
        player = veriler[BSNo]
        sonuc.append([BSNo, player[0], player[1], player[4], player[5], player[2], _bh1_bh2[0], _bh1_bh2[1], sb, gs])

    return sonuc
```

```
def _BH1_BH2_Hesapla(veriler, BSNo):
    # Veriler -> main'deki capraz_tablo'dur:
    # dict :  {  BSNo : [LNo, Ad-Soyad, Puan_Toplami, [rakipleri (LNo)], ELO, UKD, Oyuncunun Oynadigi Tum Turlar],  ..}

    puanlar = []  # Bunda oyuncunun rakiplerinin punalrini aliyoruz.
    rakipler = veriler[BSNo][3]
    for BSNo in range(1, len(veriler) + 1):
        if veriler[BSNo][0] in rakipler:
            puanlar.append(veriler[BSNo][2])
    puanlar.sort(reverse=True)

    _BH1_ = 0
    if len(puanlar) > 0:
        puanlar.pop(-1)  # En sondaki puan en dusuk puandir. BH1 icin
        for p in puanlar:
            _BH1_ += p

    _BH2_ = 0
    if len(puanlar) > 0:
        puanlar.pop(-1)  # En sondaki puan en dusuk puandir. BH2 icin
        for p in puanlar:
            _BH2_ += p

    return [_BH1_, _BH2_]



def _SB_Hesapla(veriler, BSNo):
    # Veriler -> main'deki capraz_tablo'dur:
    # dict :  {  BSNo : [LNo, Ad-Soyad, Puan_Toplami, [rakipleri (LNo)], ELO, UKD, Oyuncunun Oynadigi Tum Turlar],  ..}
    # Oyuncunun Oynadigi Tum Turlar : Burda her tur icin 3 verisi olan bir List.
    # Listin sekli : [Turdaki_rakibinin_BSNo'su, Renk, Turda_aldigi_puan]

    top_puan = 0
    for i in range(6, len(veriler[BSNo])):
        veri = veriler[BSNo][i]  # i. turun verisi.  [Turdaki_rakibinin_BSNo'su, Renk, Turda_aldigi_puan]
        rakip_bsno = veri[0]
        if rakip_bsno != "-":  # Oyuncu BYE oldogu zaman rakip_bsno "-" olur. Onun icin error vermesin diye if.
            rakip_puani = veriler[rakip_bsno][2]

            if str(veri[-1]) == str(1):  # Burda str alma nedenim veri[-1]'in "-" ve "+" te error engellemek.
                top_puan += rakip_puani
            elif str(veri[-1]) == str(0.5):  # Burda str alma nedenim veri[-1]'in "-" ve "+" te error engellemek.
                top_puan += rakip_puani / 2.0
            elif veri[-1] == "+":
                top_puan += rakip_puani

    return top_puan



def _GS_Hesapla(veriler, BSNo):
    # Veriler -> main'deki capraz_tablo'dur:
    # dict :  {  BSNo : [LNo, Ad-Soyad, Puan_Toplami, [rakipleri (LNo)], ELO, UKD, Oyuncunun Oynadigi Tum Turlar],  ..}
    # Oyuncunun Oynadigi Tum Turlar : Burda her tur icin 3 verisi olan bir List.
    # Listin sekli : [Turdaki_rakibinin_BSNo'su, Renk, Turda_aldigi_puan]
    sayi = 0  # Oynayarak kazandigi ve rakibinin gelmedigi maclarin sayisi.
    for i in range(6, len(veriler[BSNo])):
        veri = veriler[BSNo][i]  # i. turun verisi

        if str(veri[-1]) == str(1):  # Burda str alma nedenim veri[-1]'in "-" ve "+" te error engellemek.
            sayi += 1
        elif veri[-1] == "+":
            sayi += 1
        else:
            pass

    return sayi

# ============================================================================================================================

def yardimci__rakipBul(gerek_veriler):
    # Bu fonksiyon rakipBul'a yardimcidir. Oyunculari eslestirecek.

    # gerekli_veriler :  [ [ LNo, BSNo, Puan_Toplami, [Onceki Rakipleri],  Renk, Renk_Sayisi], ...]
    # Eger oyuncu sayisi tek ise BYEli oyuncu gerekli_veriler'den silinmistir.
    veriler = gerek_veriler[:]
    eslestirme = []     # Bu liste fonkisyonu cagiran (rakipleriBul) fonksiyondaki "eslestirme" gibi olacak.
                        # Oyuncu: [LNo, BSNo, Puan, Renk, Renk_Tekrari(sayi)]
    masa_sayisi = int(len(veriler) / 2)

    for masa_no in range(masa_sayisi - 1):
        oyuncu = veriler[0]     # Oyuncu her seferinde oyuncularin en yuksek puanlisi.
        rakip_bulunmadi = True
        indix = 1               # Basta oyuncu 0'indexteki oldogu icin rakibi 1 indexi veya sonrakiler "olabililir"
        beyaz = siyah = rakip = ''
        while rakip_bulunmadi:
            rakip = veriler[indix]  # Olabilecek rakip
            _3th_priority_flag = True  # 1.1 ve 1.2 olamdiginda, 1.3'e gecme yolu bulamadigim icin bunu kullanicam.
                                       # Cunku 1.2 ve 1.3'un if'teki sartlari benim algomda ayniydi ama icerigi
                                       # degisik isler yapiyordu. 1.2 calistiginda bu False olur ve 1.3 calismaz,
                                       # aksine halde calisir.
```

```
            if rakip[0] not in oyuncu[3]:   # Olabilecek rakibin LNo'su oyuncunun rakip listesine olmadigi zaman.

                # Birinci oncelik yani 1.1
                if (rakip[-2] != oyuncu[-2]) or rakip[-2] == '':  # oyuncu[-2] ve rakip[-2] ikisinin rengidir
                    # Oyuncu ve Rakip onceki turlarinin karsit rengini aliyorlar, onun icin
                    # Rakibin sonraki rengi oyuncunun onceki rengine esit olur.
                    # Ama Oyuncunun rengi sonra degistirlidigi icin rakibin rengi oyuncunun simdiki renginie esit olmayandir.
                    rakip_bulunmadi = False
                    if oyuncu[-2] == 'b':
                        # Burda oyuncunun rengi b -> s, rakibin rengi s -> b.
                        siyah = [oyuncu[0], oyuncu[1], oyuncu[2], 's', 1]   # Yeni renk aldigi icin renk sayisi 1.
                        beyaz = [rakip[0], rakip[1], rakip[2], 'b', 1]      # //
                    else:   # Oyuncunun rengi Siyah
                        # Burda oyuncunun rengi s -> b, rakibin rengi b -> s.
                        beyaz = [oyuncu[0], oyuncu[1], oyuncu[2], 'b', 1]   # Yeni renk aldigi icin renk sayisi 1.
                        siyah = [rakip[0], rakip[1], rakip[2], 's', 1]      # //

                # Ikinci oncelik 1.2
                elif (rakip[-2] == oyuncu[-2]) and (rakip[-1] == 1):
                    # Oyuncunun rangi degisecek, rakibinin rengi degismeyecek.
                    # Onun icin rakibin rengi oyuncunun eski rengine esit olacak. (if'te)
                    rakip_bulunmadi = False
                    _3th_priority_flag = False  # Yani 1.2 calisti, 1.3'u calistirma.
                    if oyuncu[-2] == 'b':
                        # Burda oyuncunun rengi b -> s, rakibin rengi degismeyecek.
                        siyah = [oyuncu[0], oyuncu[1], oyuncu[2], 's', 1]  # Yeni renk aldigi icin renk sayisi 1.
                        beyaz = [rakip[0], rakip[1], rakip[2], 'b', 2]     # Ayni renk tekrar oldogu icin 2
                    else:  # Oyuncunun rengi Siyah
                        # Burda oyuncunun rengi s -> b, rakibin rengi degismeyecek.
                        beyaz = [oyuncu[0], oyuncu[1], oyuncu[2], 'b', 1]  # Yeni renk aldigi icin renk sayisi 1.
                        siyah = [rakip[0], rakip[1], rakip[2], 's', 2]     # Ayni renk tekrar oldogu icin 2

                # Ucuncu Oncelik 1.3
                elif _3th_priority_flag and (rakip[-2] == oyuncu[-2]):
                    # Oyuncunun rengi degismez, rakibinin rengi onceki rengin karsiti olur.
                    # Onun icin rakibin rengi, oyuncunun rengiyle esit. (if'te)
                    rakip_bulunmadi = False
                    if oyuncu[-2] == 'b':
                        # Burda oyuncunun rengi degismez, rakibin rengi b -> s.
                        beyaz = [oyuncu[0], oyuncu[1], oyuncu[2], 'b', 2]  # Yeni renk almadigi icin renk sayisi 2.
                        siyah = [rakip[0], rakip[1], rakip[2], 's', 1]  # Yeni renk aldigi icin renk sayisi 1.
                    else:  # Oyuncunun rengi Siyah
                        # Burda oyuncunun rengi degismez, rakibin rengi s -> b.
                        siyah = [oyuncu[0], oyuncu[1], oyuncu[2], 's', 2]  # Yeni renk almadigi icin renk sayisi 2.
                        beyaz = [rakip[0], rakip[1], rakip[2], 'b', 1]  # Yeni renk aldigi icin renk sayisi 1.
                indix += 1

            eslestirme.append([beyaz, siyah])  # Oyuncu ve rakibini listeye ekliyoruz.
            veriler.remove(oyuncu)  # Oyuncuyu ve rakibi listeden siliyoruz.
            veriler.remove(rakip)   # Baska Oyuncu-Rakip ciftlerini bulmayi kolaylastirmak icin.

        oyuncu = veriler[0]     # Her Oyuncu: [LNo, BSNo, Puan, Renk, Renk_Tekrari(sayi)]
        rakip = veriler[1]
        x = y = ''
        if oyuncu[-2] != rakip[-2]:  # Eger oyuncu ve rakibin onceki renkleri bir birine karsit ise, oyuncu rakibinin ve
                                     # rakip oyuncunun rengini alir.
            x = [oyuncu[0], oyuncu[1], oyuncu[2], rakip[-2], 1]  # Yeni renk aldigi icin renk sayisi 1.
            y = [rakip[0], rakip[1], rakip[2], oyuncu[-2], 1]  # Yeni renk aldigi icin renk sayisi 1.
        else:   # Eger oyuncu ve rakibin onceki renkleri ayni ise, yani oyuncunun_rengi = rakibin_rengi.
            if oyuncu[-1] == 2 and rakip[-1] == 1:  # Oyuncu o rrengi 2 kez almis, rakip o rengi 1 kez almis.
               if oyuncu[-2] == "b":
                    x = [oyuncu[0], oyuncu[1], oyuncu[2], "s", 1]  # Yeni renk aldigi icin renk sayisi 1.
                    y = [rakip[0], rakip[1], rakip[2], "b", 2]  # Yeni renk aldmaigi icin renk sayisi 2.
               else:    # oyuncunun rengi Siyah
                   x = [oyuncu[0], oyuncu[1], oyuncu[2], "b", 1]  # Yeni renk aldigi icin renk sayisi 1.
                   y = [rakip[0], rakip[1], rakip[2], "s", 2]  # Yeni renk aldmaigi icin renk sayisi 2.
            else:   # Oyuncu o rrengi 1 kez almis, rakip o rengi 2 kez almis
                if rakip[-2] == "b":
                    x = [oyuncu[0], oyuncu[1], oyuncu[2], "b", 2]  # Yeni renk aldmaigi icin renk sayisi 2.
                    y = [rakip[0], rakip[1], rakip[2], "s", 1]  # Yeni renk aldigi icin renk sayisi 1.
                else:  # rakibin rengi Siyah
                    x = [oyuncu[0], oyuncu[1], oyuncu[2], "s", 2]  # Yeni renk almadigi icin renk sayisi 2.
                    y = [rakip[0], rakip[1], rakip[2], "b", 1]  # Yeni renk aldigi icin renk sayisi 1.

        if x[-1] == "b":  # Eger Oyuncu Beyaz ise
            eslestirme.append([x, y])
        else:   # Eger Rakip Beyaz ise
            eslestirme.append([y, x])


        return eslestirme


    def rakipleriBul(Capraz_Tablo, current_tur, _tek_cift):
        # Oyuncu eslestirmelerini hazirlayip return yapacak.
        # Capraz_Tablo :
        # dict :  {  BSNo : [LNo, Ad-Soyad, Puan_Toplami, [rakipleri (LNo)], ELO, UKD, Oyuncunun Oynadigi Tum Turlar],  ..}

        eslestirme = []     # eslestirme current_tur gibi yapilacak : turun karsilastirmalarini tutuyor
                            # Burda Sag -> Beyaz , Sol -> Siyah tir.
                            # nested List :   [ [ [1.Oyuncu (Beyaz)], [2.Oyuncu (Siyah)] ], .. ]
                            # Oyuncu : [ LNo, BSNo, Puan, Renk, Renk_Tekrari (sayi) ]
        gerekli_veriler = []  # List : Bu eslestirmeleri yapmak icin kullanilacak
                              # [ [ LNo, BSNo, Puan_Toplami, [Onceki Rakipleri],  Renk, Renk_Sayisi], ...]
```

```
                        # Asagida Puan_Toplami'ni ve Onceki Rakiplerini Capraz_Tablo'dan alicaz,
                        # Renk ve Renk_Sayisini curretn_tur'dan.
    if _tek_cift == "Cift":
        sayi = len(current_tur)
    else:   # _tek_cift -> "Tek
        sayi = len(current_tur) - 1

    for No in sorted(Capraz_Tablo.keys()):      # Burda LNo, BSNo, Puan_Toplami, Rakipleri'ni aliyoruz.
        gerekli_veriler.append([Capraz_Tablo[No][0], No, Capraz_Tablo[No][2], Capraz_Tablo[No][3]])

    for i in range(sayi):           # Renk ve Renk_Sayisi gerekli_veriler'e ekliyoruz.
        beyaz = current_tur[i][0]
        siyah = current_tur[i][1]
        # Asagida indexi BSNo'ya gore buluyoruz, BSNo 1'den ama List'te index 0'dan basladigi icin oyuncu BSNo'sunu
        # aldiktan sonra onu -1 yapiyoruz taki indexi versin diye.
        # BU KISMI ANLAMANIZ ZOR OLACAK, suandan tahmin edebiliyorum :)
        gerekli_veriler[beyaz[1] - 1].append(beyaz[-2])  # Rengi ekliyoruz -> b/s
        gerekli_veriler[beyaz[1] - 1].append(beyaz[-1])  # Renk Sayisini
        gerekli_veriler[siyah[1] - 1].append(siyah[-2])  # //
        gerekli_veriler[siyah[1] - 1].append(siyah[-1])  # //

    if _tek_cift == "Tek":
        beyaz = current_tur[-1][0]
        gerekli_veriler[beyaz[1] - 1].append(beyaz[-2])  # Rengi ekliyoruz -> b/s
        gerekli_veriler[beyaz[1] - 1].append(beyaz[-1])  # Renk Sayisini

    # Simdi oyunculari Puan_Toplam'larina gore Descending Sort yapicaz.
    gerekli_veriler.sort(key = lambda row : row[2], reverse = True)
    _BYEli_oyuncu = "Yok"
    if _tek_cift == "Tek":
        _BYEli_oyuncu_bulunmadi = True
        son_index = len(gerekli_veriler) - 1
        while _BYEli_oyuncu_bulunmadi:  # En dusuk puanli oyuncudan basliyoruz.
            if "BYE" not in gerekli_veriler[son_index][3]:  # Yani onceden BYE olmus rakibi yoksa
                _BYEli_oyuncu = gerekli_veriler[son_index]
                gerekli_veriler.pop(son_index)  # Simdi BYEli olan oyuncuyu listeden sil.
                _BYEli_oyuncu_bulunmadi = False
            son_index -= 1


    eslestirme = yardimci__rakipBul(gerekli_veriler)
    if _BYEli_oyuncu != "Yok":
        eslestirme.append([_BYEli_oyuncu, ["BYE"]])

    return eslestirme



def puanVer(sayi):
    if sayi == 0:
        return [0.5, 0.5]               # beraberlik (1/2)
    elif sayi == 1:
        return [1, 0]                   # Beyaz galip
    elif sayi == 2:
        return [0, 1]                   # Siyah galip
    elif sayi == 3:
        return ["+", "-"]               # Siyah maca gelmemis
    elif sayi == 4:
        return ["-", "+"]               # Beyaz maca gelmemis
    else:
        return ["-", "-"]               # Her iki oyuncu da maca gelmemis



def turdakiOyunlar(current_tur, Capraz_Tablo, tur_no, _tek_cift):
    # Bu fonksiyon bir turdaki oyunlari ele aliyor (Duzgun soyleyecek bir sey bulamadim).
    # COK ONEMLI KONU : Burda Capraz_Tablo uzerine islem yapilir, kopyasina degil.

    # Capraz_Tablo ->  dict :  {  BSNo : [LNo, Ad-Soyad, Puan_Toplami, [rakipleri (LNo)], ELO, UKD,
                                                            # Oyuncunun Oynadigi Tum Turlar],  ..}
                        # Oyuncunun Oynadigi Tum Turlar : Burda her tur icin 3 verisi olan bir List.
                        # Listin sekli : [Turdaki_rakibinin_BSNo'su, Renk, Turda_aldigi_puan]
    # current_tur -> list : turun karsilastirmalarini tutuyor
                        # Burda Sag -> Beyaz , Sol -> Siyah tir.
                        # nested List :   [ [ [1.Oyuncu (Beyaz)], [2.Oyuncu (Siyah)] ], .. ]
                        # Oyuncu : [ LNo, BSNo, Puan, Renk, Renk_Tekrari (sayi)]

    if _tek_cift == "Cift":          # Burda her turun mac sayisini aliyoruz.
         sayi = len(current_tur)     # Eger oyuncu sayisi cift ise normal .
    else:                   # Eger oyuncu sayisi tek ise en son oyuncunun rakibi BYE oldogu icin o turu atlamis olur.
        sayi = len(current_tur) - 1
    for i in range(sayi):
        puan = girdiAl(str(tur_no) + ". turda " + str(i + 1) +
                       ". masada oynanan macin sonucunu giriniz ({0}-{1}): ", 0, 5, "Tur", "")
        sonuc = puanVer(puan)       # Puan komutu aldiktan sonra her oyuncuya verilecek puan. sonuc =  [beyazin, siyahin]
        beyaz = current_tur[i][0]   # Burda Beyazin verilerini aliyoruz [ LNo, BSNo, Puan, Renk, Renk_Tekrari (sayi)]
        siyah = current_tur[i][1]   # Burda Siyahin  ////

        Capraz_Tablo[beyaz[1]].append([siyah[1], beyaz[3], sonuc[0]])   # Burda macin verilerini kaydediyoruz
        Capraz_Tablo[siyah[1]].append([beyaz[1], siyah[3], sonuc[-1]])  # [rakip_BSNo, Oyuncunun_rengi, macta_aldigi_puan]

        Capraz_Tablo[beyaz[1]][3].append(siyah[0])  # Beyazin rakibinin LNo'sunu rakipler listesine ekliyoruz.
        Capraz_Tablo[siyah[1]][3].append(beyaz[0])  # Siyahin rakibinin LNo'sunu rakipler listesine ekliyoruz.

        if sonuc[0] in [0, 0.5, 1]:  # Beyazin puani
```

```
            Capraz_Tablo[beyaz[1]][2] += sonuc[0]    # turdaki puani oyuncunun toplam puanina ekliyoruz
        elif sonuc[0] == "+":
            Capraz_Tablo[beyaz[1]][2] += 1           # Rakibi gelmedi, 1 puan aliyor.
        if sonuc[1] in [0, 0.5, 1]:  # Siyahin puani
            Capraz_Tablo[siyah[1]][2] += sonuc[1]    # turdaki puani oyuncunun toplam puanina ekliyoruz
        elif sonuc[1] == "+":
            Capraz_Tablo[siyah[1]][2] += 1           # Rakibi gelmedi, 1 puan aliyor.

    # ONEMLI KONU : "BYE" OYUNCUSUNUN KADERI BELIRLENDI Gibi Gibi.
    if _tek_cift == "Tek":
        son_oyuncu = current_tur[-1][0]
        Capraz_Tablo[son_oyuncu[1]][2] += 1     # Rakibi yok (BYE) onun icin tur atlayip 1 puan aliyor.
        Capraz_Tablo[son_oyuncu[1]][3].append("BYE")    # Rakipler listesine "BYE" ekliyoruz.
        Capraz_Tablo[son_oyuncu[1]].append(["-", "-", 1])   # Mac Sonucu.


def baslangicTur(players, renk, _tek_cift):  # ==============    COMPLETE   ===========
    # Turnuvanin 1.turundaki siralamayi yapar.
    # Players -> BSNo'ya gore siralanmis oyuncular.  dict : {1:[ELO, UKD, adSoy, LNo], ...}

    bas_tur = []    # Burda Sag -> Beyaz , Sol -> Siyah tir.
                    # nested List :   [ [ [1.Oyuncu (Beyaz)], [2.Oyuncu (Siyah)] ], .. ]
                    # Oyuncu : [ LNo, BSNo, Puan, Renk, Renk_Tekrari (sayi)]

    if _tek_cift == "Cift":             # Eger oyuncu sayisi cift ise normal oyuncu-rakip islemi.
        sayi = len(players)
    else:   # Oyuncu sayisi tek ise.    # Eger oyuncu sayisi tek ise en alttaki oyuncudan oncekiler normal oyuncu-rakip
        sayi = len(players) - 1         # islemi. Son oyuncu en sonda eklenir, cunku o BYE olur.

    for bsno in range(1, sayi, 2):                      # Burda Oyuncu_1 ve Oyuncu_2'nin sekli : [ELO, UKD, adSoy, LNo]
        if renk == "b":  # Tekler Beyaz, Ciftler Siyah.
            oyuncu_1 = players[bsno]
            beyaz = [oyuncu_1[-1], bsno, 0.0, "b", 1]
            oyuncu_2 = players[bsno + 1]
            siyah = [oyuncu_2[-1], bsno + 1, 0.0, "s", 1]
        else:           # Tekler Siyah, Ciftler Beyaz
            oyuncu_1 = players[bsno + 1]
            beyaz = [oyuncu_1[-1], bsno + 1, 0.0, "b", 1]
            oyuncu_2 = players[bsno]
            siyah = [oyuncu_2[-1], bsno, 0.0, "s", 1]

        bas_tur.append([beyaz, siyah])

    if _tek_cift == "Tek":
        oyuncu_1 = players[len(players)]    # En son oyuncu -- Oyuncu sayisi tek oldogu zaman.
        beyaz = [oyuncu_1[-1], len(players), 0.0, "", 0]   # Karsi Rakip yok; BYE. Renk yok
        siyah = ["BYE"]                                    # Karsi Rakip BYE.
        bas_tur.append([beyaz, siyah])

    return bas_tur

def turPrinter(Current_Tur, _tek_cift):
    # Current_Tur burda Sag -> Beyaz , Sol -> Siyah tir.
    # nested List :   [ [ [1.Oyuncu (Beyaz)], [2.Oyuncu (Siyah)] ], .. ]
    # Oyuncu : [ LNo, BSNo, Puan, Renk, Renk_Tekrari (sayi)]

    print("        Beyazlar        Siyahlar")
    print("MNO BSNo   LNo Puan - Puan   LNo BSNo")
    print("--- ---- ----- ----   ---- ----- ----")

    if _tek_cift == "Cift":             # Eger oyuncu sayisi cift ise normal print yapiliyor.
        sayi = len(Current_Tur)         # Eger tek ise en son oyuncunun rakibi BYE oldogu icin genel print
    else:                               # islemlerinde error engellemek icin once baska oyunculari sonra en son
        sayi = len(Current_Tur) - 1     # oyuncuyu print.

    for MSNo in range(sayi):
        beyaz = Current_Tur[MSNo][0]
        siyah = Current_Tur[MSNo][1]
        print("{:3} {:4} {:5} {:4} - {:4} {:5} {:4}".
              format(MSNo + 1, beyaz[1], beyaz[0], beyaz[2], siyah[2], siyah[0], siyah[1]))

    if _tek_cift == "Tek":
        beyaz = Current_Tur[-1][0]
        siyah = Current_Tur[-1][1]
        print("{:3} {:4} {:5} {:4} - {:4}".format(len(Current_Tur), beyaz[1], beyaz[0], beyaz[2], siyah[-1]))
    print("")


def tabloYap(baslangic_siralama, hangi_kriter):
    # Bu fonk. Capraz Tablo ve Nihai Siralama olusturmak icindir.
    # baslangic_siralam -> dict : {1:[ELO, UKD, adSoy, LNo], ...}
    # hangi_kriter : string -> "BSNo"

    # Eger hangi_kriter == "BSNo" ise dict return yapar.
    if hangi_kriter == "BSNo":  # Capraz Tablo icin
        tablo = {}  # tablo -> dict: {BSNo: [LNo, Ad - Soyad, Puan_Toplami, [rakipleri(LNo)], ELO, UKD],..}
        sirlama = sorted(baslangic_siralama.keys())
        for i in sirlama:
            oyuncu = baslangic_siralama[i]
            tablo[i] = [oyuncu[-1], oyuncu[2], 0.0, [], oyuncu[0], oyuncu[1]]

    return tablo
```

```
def baslangicSiralama(tum_oyuncular = dict):
    # BU Fonksiyon baslangic siralamayi yapip onu print yapar sonra da o siralamay return.
    # tum_oyuncular -> dict : {LNo: [ad_soyad, ELO, UKD], ..}
    # Return yapilan siralam Dictionary.  dict -> {baslangic_sira_no : [ELO, UKD, Ad-Soyad, LNo]}

    players = []
    for k, v in tum_oyuncular.items():          # players = [ [ELO, UKD, Ad-Soyad, LNo], ] olusturmak icin.
        players.append([v[1], v[2], v[0], k])

    # Aşağıdaki 3 lambda satırı ile sort yapıcaz, proje dosyasinda belirtilen Order şedkline göre.
    players.sort(key=lambda row: row[-1])                   # LNo -> Ascending Order.
    players.sort(key=lambda row: row[2])                    # Ad-Soyad -> Turkce alfabesine gore.
    players.sort(key=lambda row: row[0:2], reverse=True)    # ELO, UKD -> Descending Order.

    bas_siralama = dict()   # Dict : {BSNo : [ELO, UKD, Ad-Soyad, LNo], ...}

    print("BSNo   LNo Ad-Soyad      ELO  UKD")
    print("---- ----- ------------- ---- ----")

    # Sort yapilan oyunculari alip onlara BSNo vericez.
    for i in range(len(players)):
        bas_siralama[i + 1] = players[i]
        print("{:4} {:5} {:13} {:4} {:4}".format((i + 1), players[i][-1], players[i][2], players[i][0], players[i][1]))

    return bas_siralama


def girdiAl(komut, sinir1, sinir2, input_type, data):  # data -> Dictionary of players. Used for finding duplicate LNo's
    # BURDA data : Ihtiyaca gore oyuncu dict'i de olabilir, girdi-tipi de. (UKD, ELO, Tur)

    if input_type == "ELO" or input_type == "UKD":                  # This if is for ELO/UKD.
        girdi = int(input(komut.format(str(sinir1), str(sinir2))))
        while not(girdi >= sinir1 or girdi == sinir2):
            girdi = int(input(komut.format(str(sinir1), str(sinir2))))
    elif input_type == "LNo":                               # This is for LNo.
        girdi = int(input(komut))
        while girdi in data.keys():
            girdi = int(input(komut))
    elif input_type == "Tur":                               # Bu Tur Sayisini ve Puan komutunu (0-5) almak icin.
        girdi = int(input(komut.format(sinir1, sinir2)))
        while not (sinir1 <= girdi <= sinir2):
            girdi = int(input(komut.format(sinir1, sinir2)))
    else:                                                   # Renk almak icin.
        girdi = input(komut.format(sinir1, sinir2)).lower()
        while not (girdi == sinir1 or girdi == sinir2):
            girdi = input(komut.format(sinir1, sinir2)).lower()
    return girdi

# ======================================================================================================================

def main():

    # Burda capraz tabloyu buyuk harfle baslamamin nedeni bunun daha anlasilir yapmasi.
    # Yani BASKA DEGISKENLERDEN DAHA ONEMLI OLDOGUNU GOSTERMEK ICIN.
    Capraz_Tablo = []
    # dict :  {  BSNo : [LNo, Ad-Soyad, Puan_Toplami, [rakipleri (LNo)], ELO, UKD, Oyuncunun Oynadigi Tum Turlar],  ..}
                      # Oyuncunun Oynadigi Tum Turlar : Burda her tur icin 3 verisi olan bir List.
                      # Listin sekli : [Turdaki_rakibinin_BSNo'su, Renk, Turda_aldigi_puan]
                      # EN SONUNDA Yani maclar bittikten sonra Capraz Tabloya her oyuncu icin
                      # 5 Eleman daha eklenecek; Top_Puan, BH-1, BH-2, SB, GS

    oyuncular = {}      # Dictionary -> LNo : [adSoyad, ELO, UKD]

    LNo = girdiAl("Oyuncunun lisans numarasini giriniz (bitirmek için 0 ya da negatif giriniz): ", "", "", "LNo", oyuncular)
    while LNo > 0:
        ad_soyad = input("Oyuncunun adini-soyadini giriniz: ").strip()
        ad_soyad = ad_soyad.replace('i', 'İ').replace('ı', 'I').upper()
        ELO = girdiAl("Oyuncunun ELO'sunu giriniz (en az {0}, yoksa {1}): ", 1000, 0, "ELO", "")
        UKD = girdiAl("Oyuncunun UKD'sini giriniz (en az {0}, yoksa {1}): ", 1000, 0, "UKD", "")
        oyuncular[LNo] = [ad_soyad, ELO, UKD]

        # Baska oyuncu Var/Yok kontrol etmek icin kullanilacak.
        print("")
        LNo = girdiAl("Oyuncunun lisans numarasini giriniz (bitirmek için 0 ya da negatif giriniz): ", "", "", "LNo", oyuncular)

    bas_siralama = baslangicSiralama(oyuncular)  # baslangic_siralam -> dict : {1:[ELO, UKD, adSoy, LNo], ...}
    tur_sayisi = girdiAl("Turnuvadaki tur sayisini giriniz ({0}-{1}): ",
                         ceil(log(len(oyuncular), 2)), len(oyuncular) - 1, "Tur", "")  # Burda dogru sonuc vermeye bilir.(girdAl)
    renk = girdiAl("Baslangic siralamasina gore ilk oyuncunun ilk turdaki rengini giriniz ({0}/{1}): ", "b", "s", "Renk", "")

    # ==================================       Girdiler Bitti     -      Turnuva Baslayacak    ========================
    print("\n")
    if len(oyuncular) % 2 == 0:     # Oyuncu sayisi tek mi cift?
        _tek_cift_ = "Cift"
    else:
        _tek_cift_ = "Tek"
    current_tur = baslangicTur(bas_siralama, renk, _tek_cift_)
    Capraz_Tablo = tabloYap(bas_siralama, "BSNo")

    for tur in range(1, tur_sayisi + 1):
        print("\n{0}. Tur Eşleştirme Listesi:".format(tur))
        turPrinter(current_tur, _tek_cift_)
        turdakiOyunlar(current_tur, Capraz_Tablo, tur, _tek_cift_)     # Program Maclara Burdan gecer.
```

```
        current_tur = rakipleriBul(Capraz_Tablo, current_tur, _tek_cift_)  # Burda yeni eslestirmeler, sonra program basina.

    # ================================          Turnuva Bittiiii   -   Simdi Sonuclar      ============================

    puanlar_tablosu = tumPuanlariHesapla(Capraz_Tablo)
    Nihai_Sonuc = nihaiSonuc(puanlar_tablosu)
    caprazTablo(Nihai_Sonuc, Capraz_Tablo, tur_sayisi

main()
```

Programcı kataloğu

Programımızın analiz ve tasarımı : 7 gün

gerçekleştirim ve test aşaması : 4 gün

Raporlama : 4 saat

Kullanılan kütüphane:

```
from math import floor, ceil, log
```

math kütüphanesiden floor cell ve log fonksiyonlarını çağırdık.
Math.floor () fonksiyonu, belirtilen bir değerden küçük veya ona eşit olan en yakın tam sayı değerini döndürmek için kullanılır.
Math.ceil () fonksiyonu bir değerin tavan değerini döndürür, yani o değerden küçük olmayan en küçük tamsayı.
Math.log() fonksiyonu, bir sayının doğal logaritmasını veya sayının logaritmasını tabana döndürür.

Kullanılan fonksiyonlar:

```
def nihaiSonuc(puan_tablosu):
```

nahai sonucumuzu çapraz tablolarla birlikte tablolar halinde yazan, döndüren ve parametre olarak puan tablosunu alan fonksiyon.

```
def caprazTablo(nihai_sonuc, veriler, tur_sayisi):
```

nahai sonucumuzu çapraz tablolarla birlikte tablolar halinde yazan, döndüren ve parametre olarak puan tablosunu alan fonksiyon.

```
def tumPuanlariHesapla(veriler):
```

Bu fonk tum verileri alip puanlari toplar, sonra liste seklinde return yapar.
sonra bunun return yaptigi listeden Nihai Sonuc ve Son Sonucu Tutan Capraz Tablo olusturulur.

```
def _BH1_BH2_Hesapla(veriler, BSNo):
```

bu fonksiyon main içinde olan çapraz tablo değerleri, veriler olarak başlangıç numara ile birlikte parametre olarak alıyor. BH1 ve BH2 yı sonucunu döndürür.

```
def _SB_Hesapla(veriler, BSNo):
```

bu fonksiyon main içinde olan çapraz tablo değerleri, veriler olarak başlangıç numara ile birlikte parametre olarak alıyor. Toplam puanın sonucunu döndürür.

```
def _GS_Hesapla(veriler, BSNo):
```

bu fonksiyon main içinde olan çapraz tablo değerleri, veriler olarak başlangıç numara ile birlikte parametre olarak alıyor. sayıyı döndürür.

```
def yardimci__rakipBul(veriler):
```

bu fonksiyon rakipBul fonksiyona yardımcıdır. Oyuncuları eşleştiriyor. Ve eşleştirme sonucunu ona göre döndürüyor.

```
def rakipleriBul(Capraz_Tablo, current_tur, _tek_cift):
```

bu fonksiyon çapraz tablo, şimdiki tür ve tek çift i parametre olarak alıyor. Oyuncu eşleştirmelerini hazirlayıp döndürüyor.

```
def puanVer(sayi):
```

bu fonksiyon beraberlik , beyazın galip, siyahın galip, gelmeyen siyah ve beyaz ve gelmeyen iki oyuncu durumları için sonucu sayı olarak döndürüyor.

```
def turdakiOyunlar(current_tur, Capraz_Tablo, tur_no, _tek_cift):
```

Bu fonksiyon bir turdaki oyunlari ele aliyor.
COK ONEMLI KONU : Burda Capraz_Tablo uzerine islem yapilir, kopyasina degil.

```
def baslangicTur(players, renk, _tek_cift):
```

bu fonksiyon turnuvanin 1. Turundaki siralamayi yapar.

```
def turPrinter(Current_Tur, _tek_cift):
```

bu fonksiyon türlerim çıktısını veriyor. Sağ beyazlar için , Sol ise Siyahlar içindir.

```
def tabloYap(baslangic_siralama, hangi_kriter):
```
bu fonksiyon çapraz tablo ve nihai siralama oluşturmak içindir.

```
def baslangicSiralama(tum_oyuncular = dict):
```

bu Fonksiyon baslangic siralamayi yapip onu print yapar sonra da o siralamayi return.

```
def girdiAl(komut, sinir1, sinir2, input_type, data):
```

bu fonksiyon dictionary halinde tutulan oyuncuları, data olarak parametre olarak alır. Burada data ihtiyaca göre oyuncu sözcuğu da olabilir. Girdi tipi de.

```
def main():
```

main fonksiyonumuz ana fonksiyonumuzdur. içinde, Burda capraz tabloyu buyuk harfle baslamamin nedeni bunun daha anlasilir yapmasi.
Yani BASKA DEGISKENLERDEN DAHA ONEMLI OLDOGUNU GOSTERMEK ICIN.

Kullancı kataloğu :

Programim 100% çalışmaktadır.

EKRAN ÇIKTISI
program ilk çalıştırdığında girdileri topluca giriyoruz.

```
Oyuncunun lisans numarasini giriniz (bitirmek için 0 ya da negatif giriniz): 555
ALİ KAYA
0
1200
6
Ahmet Ağa
1200
0
70
ömürcan ay
0
1800
88
Ayşe Sarı
1000
1300
123
Ali Kaya
0
1200
222
ırmak su bir
1500
1600
3
Fatma BEYAZ
0
0
4
ilknur akar
1500
1600
99
çetin can ay
1500
1700
0
5
s
```

```
Ali Kaya
0
1200
222
ırmak su bir
1500
1600
3
Fatma BEYAZ
0
0
4
ilknur akar
1500
1600
99
çetin can ay
1500
1700
0
5
s
2
1
2
4
1
1
2
1
2
2
1
0
0
0
0
1
1
1
```

Oluşturulan başlangıç sıralama listesi aşağıdaki
şekilde görüntülenir:

```
2
4
1
1
2
1
2
2
1
0
0
0
0
1
1
1
2
5
Oyuncunun adini-soyadini giriniz: Oyuncunun ELO'sunu giriniz (en az 1000, yoksa 0): Oyuncunun UKD'sini giriniz (en az 1000, yoksa 0):
Oyuncunun lisans numarasini giriniz (bitirmek için 0 ya da negatif giriniz): Oyuncunun adini-soyadini giriniz: Oyuncunun ELO'sunu giriniz (en az 1000, yoksa 0): Oyuncunun UKD'sini giriniz (en az 1000, yoksa 0):
Oyuncunun lisans numarasini giriniz (bitirmek için 0 ya da negatif giriniz): Oyuncunun adini-soyadini giriniz: Oyuncunun ELO'sunu giriniz (en az 1000, yoksa 0): Oyuncunun UKD'sini giriniz (en az 1000, yoksa 0):
Oyuncunun lisans numarasini giriniz (bitirmek için 0 ya da negatif giriniz): Oyuncunun adini-soyadini giriniz: Oyuncunun ELO'sunu giriniz (en az 1000, yoksa 0): Oyuncunun UKD'sini giriniz (en az 1000, yoksa 0):
Oyuncunun lisans numarasini giriniz (bitirmek için 0 ya da negatif giriniz): Oyuncunun adini-soyadini giriniz: Oyuncunun ELO'sunu giriniz (en az 1000, yoksa 0): Oyuncunun UKD'sini giriniz (en az 1000, yoksa 0):
Oyuncunun lisans numarasini giriniz (bitirmek için 0 ya da negatif giriniz): Oyuncunun adini-soyadini giriniz: Oyuncunun ELO'sunu giriniz (en az 1000, yoksa 0): Oyuncunun UKD'sini giriniz (en az 1000, yoksa 0):
Oyuncunun lisans numarasini giriniz (bitirmek için 0 ya da negatif giriniz): Oyuncunun adini-soyadini giriniz: Oyuncunun ELO'sunu giriniz (en az 1000, yoksa 0): Oyuncunun UKD'sini giriniz (en az 1000, yoksa 0):
Oyuncunun lisans numarasini giriniz (bitirmek için 0 ya da negatif giriniz): Oyuncunun adini-soyadini giriniz: Oyuncunun ELO'sunu giriniz (en az 1000, yoksa 0): Oyuncunun UKD'sini giriniz (en az 1000, yoksa 0):
Oyuncunun lisans numarasini giriniz (bitirmek için 0 ya da negatif giriniz): Oyuncunun adini-soyadini giriniz: Oyuncunun ELO'sunu giriniz (en az 1000, yoksa 0): Oyuncunun UKD'sini giriniz (en az 1000, yoksa 0):
Oyuncunun lisans numarasini giriniz (bitirmek için 0 ya da negatif giriniz): BSNo   LNo Ad-Soyad      ELO  UKD
---- ----- ------------ ---- ----
   1    99 ÇETİN CAN AY  1500 1700
   2   222 IRMAK SU BİR  1500 1600
   3     4 İLKNUR AKAR   1500 1600
   4     6 AHMET AĞA     1200    0
   5    88 AYŞE SARI     1000 1300
   6    70 ÖMÜRCAN AY       0 1800
   7   123 ALİ KAYA         0 1200
   8   555 ALİ KAYA         0 1200
   9     3 FATMA BEYAZ      0    0
Turnuvadaki tur sayisini giriniz (4-8): Baslangic siralamasina gore ilk oyuncunun ilk turdaki rengini giriniz (b/s):
```

Turnuvadaki tür sayısı kadar eşleştirme döndürüyor.

```
1. Tur Eşleştirme Listesi:
       Beyazlar        Siyahlar
MNO BSNo   LNo Puan - Puan   LNo BSNo
--- ---- ----- ----   ---- ----- ----
  1    2   222  0.0 -  0.0    99    1
  2    4     6  0.0 -  0.0     4    3
  3    6    70  0.0 -  0.0    88    5
  4    8   555  0.0 -  0.0   123    7
  5    9     3  0.0 - BYE

1. turda 1. masada oynanan macin sonucunu giriniz (0-5): 1. turda 2. masada oynanan macin sonucunu giriniz (0-5): 1. turda 3. masada oynanan macin sonucunu giriniz (0-5): 1. turda 4. masada oynanan macin sonucunu giriniz
 (0-5):
2. Tur Eşleştirme Listesi:
       Beyazlar        Siyahlar
MNO BSNo   LNo Puan - Puan   LNo BSNo
--- ---- ----- ----   ---- ----- ----
  1    1    99  1.0 -  1.0     6    4
  2    5    88  1.0 -  1.0   123    7
  3    9     3  1.0 -  0.0   222    2
  4    6    70  0.0 -  0.0     4    3
  5    8   555  0.0 - BYE

2. turda 1. masada oynanan macin sonucunu giriniz (0-5): 2. turda 2. masada oynanan macin sonucunu giriniz (0-5): 2. turda 3. masada oynanan macin sonucunu giriniz (0-5): 2. turda 4. masada oynanan macin sonucunu giriniz
 (0-5):
3. Tur Eşleştirme Listesi:
       Beyazlar        Siyahlar
MNO BSNo   LNo Puan - Puan   LNo BSNo
--- ---- ----- ----   ---- ----- ----
  1    5    88  2.0 -  2.0    99    1
  2    2   222  1.0 -  1.0     6    4
  3    7   123  1.0 -  1.0    70    6
  4    9     3  1.0 -  1.0   555    8
  5    3     4  0.0 - BYE
```

3-4-5 türdeki eşleştirmeler

```
3. turda 1. masada oynanan macin sonucunu giriniz (0-5): 3. turda 2. masada oynanan macin sonucunu giriniz (0-5): 3. turda 3. masada oynanan macin sonucunu giriniz (0-5): 3. turda 4. masada oynanan macin sonucunu giriniz
 (0-5):
4. Tur Eşleştirme Listesi:
       Beyazlar        Siyahlar
MNO BSNo   LNo Puan - Puan   LNo BSNo
--- ---- ----- ----   ---- ----- ----
  1    1    99  3.0 -  2.0   123    7
  2    4     6  2.0 -  2.0    88    5
  3    2   222  1.0 -  1.5   555    8
  4    3     4  1.0 -  1.5     3    9
  5    6    70  1.0 - BYE

4. turda 1. masada oynanan macin sonucunu giriniz (0-5): 4. turda 2. masada oynanan macin sonucunu giriniz (0-5): 4. turda 3. masada oynanan macin sonucunu giriniz (0-5): 4. turda 4. masada oynanan macin sonucunu giriniz
 (0-5):
5. Tur Eşleştirme Listesi:
       Beyazlar        Siyahlar
MNO BSNo   LNo Puan - Puan   LNo BSNo
--- ---- ----- ----   ---- ----- ----
  1    3     4  2.0 -  3.5    99    1
  2    7   123  2.5 -  2.5     6    4
  3    5    88  2.5 -  2.0   555    8
  4    9     3  1.5 -  2.0    70    6
  5    2   222  1.5 - BYE

5. turda 1. masada oynanan macin sonucunu giriniz (0-5): 5. turda 2. masada oynanan macin sonucunu giriniz (0-5): 5. turda 3. masada oynanan macin sonucunu giriniz (0-5): 5. turda 4. masada oynanan macin sonucunu giriniz
 (0-5):
```

En sonda çapraz tablo ve nihai sonuç ekrana döndürlür.

```
5. turda 1. masada oynanan macin sonucunu giriniz (0-5): 5. turda 2. masada oynanan macin sonucunu giriniz (0-5): 5. turda 3. masada oynanan macin sonucunu giriniz (0-5): 5. turda 4. masada oynanan macin sonucunu giriniz
 (0-5):


Nihai Sıralama Listesi:
SNo BSNo   LNo Ad-Soyad      ELO  UKD Puan  BH-1  BH-2    SB GS
--- ---- ----- ------------ ---- ---- ---- ----- ----- ----- --
  1    1    99 ÇETİN CAN AY 1500 1700  3.5  11.5   9.5  8.25  3
  2    3     4 İLKNUR AKAR  1500 1600  3.5  11.0   9.0  9.25  3
  3    7   123 ALİ KAYA        0 1200  3.5  10.0   8.0   8.5  3
  4    6    70 ÖMÜRCAN AY      0 1800  3.0   8.0   6.0   4.0  3
  5    5    88 AYŞE SARI    1000 1300  2.5  13.5  10.5   7.5  2
  6    4     6 AHMET AĞA    1200    0  2.0  10.5   7.0   3.5  2
  7    2   222 IRMAK SU BİR 1500 1600  2.0  10.0   7.0   2.0  2
  8    8   555 ALİ KAYA        0 1200  2.0   6.0   3.5  2.25  1
  9    9     3 FATMA BEYAZ     0    0  2.0   5.5   3.5  2.75  1


Çapraz Tablo:
BSNo SNo   LNo Ad-Soyad      ELO  UKD  1. Tur  2. Tur  3. Tur  4. Tur  5. Tur    SB  GS
---- --- ----- ------------ ---- ---- ------- ------- ------- ------- ------- ----- --
   1   1    99 ÇETİN CAN AY 1500 1700   2 s 1   4 b 1   5 s 1   3 b ½   7 s 0  8.25  3
   2   7   222 IRMAK SU BİR 1500 1600   1 b 0   9 s 1   3 b 0   6 s 0   - - 1   2.0  2
   3   2     4 İLKNUR AKAR  1500 1600   4 s 0   6 b 1   2 s 1   1 s ½   5 b 1  9.25  3
   4   6     6 AHMET AĞA    1200    0   3 b 1   1 s 0   7 s 0   - - 1   6 b 0   3.5  2
   5   5    88 AYŞE SARI    1000 1300   6 s 1   7 b 1   1 b 0   8 b ½   3 s 0   7.5  2
   6   4    70 ÖMÜRCAN AY      0 1800   5 b 0   3 s 0   - - 1   2 b 1   4 s 1   4.0  3
   7   3   123 ALİ KAYA        0 1200   8 s +   5 s 0   4 b 1   9 s ½   1 b 1   8.5  3
   8   8   555 ALİ KAYA        0 1200   7 b -   - - 1   9 b ½   5 s ½   9 b -  2.25  1
   9   9     3 FATMA BEYAZ     0    0   - - 1   2 b 0   8 s ½   7 b ½   8 s -  2.75  1

Process finished with exit code 0
```