```c
#include <stdio.h>
#include <stdlib.h>

#define SATIRLIMIT 81
#define SATIRMIN 40
#define KELIMELIMIT 22
#define ONBELLEK 40
#define DOSYA 80
#define SATIRBOL 22

/* Sabitler ve Anlamlari */
/* --------------------- */
/* SATIRLIMIT : Satir genisligi ust limitinin bir fazlasi.
   SATIRMIN   : Satir genisliginin alt limiti.
   KELIMELIMIT: Kelime uzunlugu limitinin iki fazlasi. Program bu iki fazlalikten birini bosluk
                (ASCII:32) digerini de NULL (ASCII:0) yazmak icin kullanabilir.
   ONBELLEK   : Programin bir satira yerlestirbilinecek kelime sayisini bulurken ele alacagi
                maksimum deger. Ornegin ONBELLEK 40, bir satirda en fazla 40 kelime olabilecegi
                anlamina geliyor. Onbellegin dusuk tutulmasi RAM kazanci saglar. Odevde verilen
                limitler icin bir satir en fazla 80 karakter olabiliyorsa her kelime birer harf-
                ten olusursa bile bir satirda en fazla 40 kelime olabileceginden en dusuk onbel-
                lek degeri 40'tir.
   DOSYA      : Dosyanin bir satirinin en fazla kac karakter icerebilecegini belirten sabit.
   SATIRBOL   : justify ekran ciktisinin kac satirda bir bolunmesi gerektigini belirten sabit.

   Bu programda asagidaki limitler vardir ve degistirilemez:
    * Program sadece ASCII karakterlerini okur.
    * Kelime aralarinda sadece birer bosluk karakteri olmalidir.

   Bu programda odevde verilen su sinirlar yoktur:
    * Metin ASCII tablosunda olan herhangi bir noktalama isaretini icerebilir.
    * Program buyuk ve kucuk harf icerebilir. Ancak arama buyuk/kucuk harf duyarlidir.
    * Pragraf 5000 karakterden cok daha fazla karaktere sahip olabilir. Bu programin limiti asa-
    gidaki gibidir:
                    paragraf_karakter_sayisi * 22 < C#'deki int tipi degiskenin makisimum degeri

   Bu programda asagidaki limitler kolayca degistirilebilir:
    * SATIRLIMIT sabiti degistirilerek bir satirin maksimum uzunlugu ayarlanabilir. Burada dik-
      kat edilmesi gereken ONBELLEK degeri (SATIRLIMIT - 1) / 2'den az OLMAMALIDIR. SATIRLIMIT
      SATIRMIN sabitine esit ya da buyuk olmalidir. SATIRLIMIT sabiti KELIMELIMIT - 2'den buyuk
      olmalidir.
    * SATIRMIN sabiti degistirilerek bir satirin minimum uzunluk limiyi ayarlanabilir. Burada
      dikkat edilmesi gereken SATIRMIN SATIRLIMIT sabitine esit ya da kucuk olmalidir. SATIRMIN
      sabiti KELIMELIMIT - 2'ye esit ya da daha buyuk olmalidir.
    * KELIMELIMIT sabiti degistirilerek metinde olabilecek en uzun kelime limiti artirilabilir.
      KELIMELIMIT sabiti SATIRMIN ve SATIRLIMIT sabitine esit ya da daha kucuk olmalidir.
    * ONBELLEK sabiti degistirilerek programin bir satira yerlestirilebilinecek kelime sayisini
      bulurken ele alacagi maksimum kelime sayisini degistirebilirsiniz. Bu sayinin
      (SATIRLIMIT - 1) / 2 ifadesine esit olmasi beklenir. Fazlasi bosuna RAM harcarken daha az
      olmasi programin cokmesine neden olabilir. Onbellegi arttirmak programi HIZLANDIRMAZ.
    * SATIRLIMIT limitini sifirdan buyuk istediginiz tam sayi yapabilirsiniz.

*/

void sirala(int *sirala, int *sayi) {

    /* Bu fonksiyon *sirala dizisini siralayarak (B->K) geri gonderir.
       sirala dizisi siralanirken *sirala dizisinin herhangi iki
       elemaninin yerini degistirirken *sayi dizisinde de ayni
       iki elemanin yerini degistirir.

       Örnek Girdi
       -----------
       Sirala: 12 32 56 43 23
       Sayi  : 0  1  2  3  4

       Cikti Dizileri
       --------------
       Sirala: 56 43 32 23 12
       Sayi  : 2  3  1  3  0

       Bu fonksiyona gonderilen diziler ayni sayida elemana sahip
       olmalidirlar. (Eleman sayisi: KELIMELIMIT-2)

       Bu fonksiyon selection sort algoritmasinin biraz etkinlestirilmis
       bir halini kullanir.

       Bu fonksiyonun aynisini daha once Sayisal Loto LAB uygulamasinda ve
       Genel Seçimler ödevinde de kullanmistim.

    */

    int sayac1, sayac2;
    int gecici;

    for (sayac1=0; sayac1<KELIMELIMIT-3;sayac1++) { // Niye KELIMELIMIT-2 değil de KELIMELIMIT-3? Bu bir hata değil, etkinlik.
        for (sayac2=sayac1 + 1; sayac2<KELIMELIMIT-2;sayac2++) {
            if (sirala[sayac1] < sirala[sayac2]) {
                gecici = sirala[sayac1];
                sirala[sayac1] = sirala[sayac2];
                sirala[sayac2] = gecici;

                gecici = sayi[sayac1];
                sayi[sayac1] = sayi[sayac2];
                sayi[sayac2] = gecici;
            }
        }
    }
}

void beklet() {
    /* Bu fonksiyonun tek amaci kullaniciyi bir tusa basana kadar
       bekletmek, ve sonra ekrani silmektir.                    */
```

```
104         printf("\nDevam etmek icin bir tusa basin.");
105         getche();
106         system ("cls");
107     }
108
109     void bosluksayisi(int bosluk_karakter, int bosluk_sayisi, int soldan_mi, int *ciktidizisi) {
110
111         /* Fonskiyonun amacı bir satir için gönderilecek bosluk_karakter kadar
112            boşluk sayısını bosluk_sayisi kadar boşluğa kaçar kaçar yerleştirileceğini
113            gösteren bir ciktidizisi olsuturur. soldan_mi degeri 0 olursa bolsuklari
114            soldan baslayarak 1 olursa sagdan baslayarak dizer.
115
116            Fonksiyon ciktidizisinde 0 degerini gonderebilir. Bu fonksiyona sadece
117            fazlalik bosluklar gelmektedir. Bunlar disinda zaten her kelimenin sonunda
118            bir bosluk oldugu kabul edilmektedir.
119
120            Örnek
121            -----
122            Aşağıdaki metin 20 karakterlik satır genişliğine sığacak. Bu fonksiyonun
123            asagidaki degerler ile cagiriliyor olmasi beklenir:
124
125            12345678901234567890
126            Kartal kalkar dal sarkar dal sarkar kartal kalkar
127
128            Yukarıdaki cümlenin ilk satırı için bu fonksiyona bosluk_karakter girdisi 2'dir.
129
130            Her kelime sonundaki boşluk da dahil kabul edildiğinden;
131            kartal = 7 karakter
132            kalkar = 7 karakter
133            dal    = 4 karakter
134            --------------------
135            1.satırda toplam 18 karakter var.
136            21 - 18 = 2 tane boşluk fazlalığı var. Bunları dağıtmak lazım. (bosluk_karakter=3)
137            (Neden 21? Program her zaman satir sayisinin bir fazlasina gore islem yapmaktadir.)
138
139            Birinci satırda kelime aralarında toplam 2 boşluk var. (kartal ile kalkar ve kalkar
140            ile dal arasında) (bosluk_sayisi = 2)
141
142            1.satir için boşluklar soldan dağıtılacağından soldan_mi = 0
143
144            Bu fonksiyonun bosluksayisi(2,2,0,ciktidizisi) degeri için aşağıdaki gibi bir çıktı
145            dizisi döndürmesi beklenir:
146
147            1.SATIR İÇİN
148            indis deger
149            ----- -----
150            0     2
151            1     1
152
153            12345678901234567890
154            Kartal+++kalkar++dal
155
```

```
156            Bu dizinin anlami: 1.boslukta 'kartal+' disinda 2 bosluk daha oalcak. 2.boslukta
157            'kaklar+' disinda bir bosluk daha olacak.
158
159            Bunu yapan algoritma aşağıdadır.
160         */
161
162         int indis;
163
164         for (indis=0;indis<ONBELLEK;indis++) {  /* Cikti dizisi sifirlanir. */
165             ciktidizisi[indis] = 0;
166         }
167
168         if (soldan_mi == 0) { /* Hesaplamaya baslama noktasina gore indise bir deger atanir.        */
169             indis = 0;        /* Fazlaliklari dagitmaya soldan (yani 0.bosluktan) basla            */
170         } else {
171             indis = bosluk_sayisi-1; /* Fazlaliklari dagitmaya sagdan (yani bosluk_sayisi-1) basla  */
172         }
173
174         while(bosluk_karakter != 0) { /* Dagitilacak fazlalikler bitene kadar dön */
175             bosluk_karakter--;         /* Fazlaliklardan biri dagitildi. */
176             ciktidizisi[indis]++;      /* indis elemanine fazla bosluklardan birini kakala. */
177
178             /* Bir sonraki dizi elemanina gecme kisminin BASLANGICI */
179             if (soldan_mi == 0) {
180                 indis++;
181                 if (indis == bosluk_sayisi) {
182                     indis = 0;
183                 }
184             } else {
185                 indis--;
186                 if (indis == -1) {
187                     indis = bosluk_sayisi - 1;
188                 }
189             }
190             /* Bir sonraki dizi elemanina gecme kisminin SONU */
191         }
192     }
193
194     int siradakikelime(FILE *fp, char *kelimeciktisi) {
195
196         /* Fonksiyon dosya göstericisinin bulunduğu konumdan sonraki ilk kelimeyi
197            kelimeciktisi dizisi içerisinde döndürür. Fonksiyon çıktı olark kendisi
198            kelimenin uzunluğunu verir. Fonskiyon döndürdüğü diziye ve kelime uzun-
199            luğuna bir de boşluk karakteri (ASCII:32) ekler.
200
201             Örneğin fp ile tanımlı dosyanın içeriği şuysa:
202             'google is the best search engine ' (dosyanin bitiminde de bir bosluk olduguna dikkat ediniz)
203             fonksiyon birinci çalışında (eğer dosya göstericisi 0 ise)
204
205             6 ve dizi olarak da "g" "o" "o" "g" "l" "e" " " ve bundan sonraki tüm
206             dizi elemanlari için NULL (ASCII:0) döndürür.
207
```

```
208           fonksiyon ikinci çalışışında (eğer dosya göstericisinin yeri değiştirilmediyse)
209
210           3 ve dizi elemanı olarak da "i" "s" " " NULL döndürür.
211       */
212
213       int indis;
214       for (indis=0;indis<KELIMELIMIT;indis++) {   /* Önce çıktı dizisi NULL'a sıfırlanır. */
215           kelimeciktisi[indis] = 0;
216       }
217
218       indis = 0;
219
220       do {
221           kelimeciktisi[indis] = fgetc(fp);   /* Sıradaki karakteri oku sonra. */
222           if (kelimeciktisi[indis] == EOF) { /* Dosya sonu geldiyse dükkanı kapatalım. */
223               kelimeciktisi[indis] = 0;      /* Dosya sonu oldugu icin -1 atanmisti, ama NULL olmali. */
224               return -1;
225           }
226           indis++;
227       } while (kelimeciktisi[indis-1]!=32);  /* Boşluk karakterine kadar dön */
228
229       return indis;
230   }
231
232   void justify(FILE *fp,int satiruz_hedef) {
233
234       /* Bu fonksiyon fp'de acilmis dosyanin icerigini satiruz_hedef satir uzunlugunda satirlarda
235          iki yana yaslayarak ekrana yazdirir.
236
237          fp ile acilmis dosya asagidaki ozelliklere sahip olmalidir:
238          * Satir sonu karakteri (ASCII:10) icermemelidir.
239          * Dosya bitiminde bosluk (ASCII:32) karakteri olmalidir.
240          Diger limitler Sabitler ve Anlamlari kismindaki aciklamalarda bulunabilir. Dosyanin odevde
241          verilenden yukaridaki ozelliklere sahip bir dosyaya donusturulmesi main() fonksiyonunda
242          yapilmistir.
243
244       */
245
246       /* Degişken tanimlamalari BASLANGICI */
247       int kelime_uzunlugu[ONBELLEK];
248       int gosterici_konum_son = 0;
249       int gosterici_konum_ilk = 0;
250       char kelime[KELIMELIMIT];
251       int bosluk_karakter;
252       int boslukdizisi[ONBELLEK];
253       int sayac;
254       int sayac2;
255       int kac_kelime_sigacak;
256       int satirno=0;
257       int cik = 0;
258       int satiruz;
259       /* Degisken tanimlamalari SONU */
```

```
260
261       fseek(fp,0,0); /* Öncelikle dosyanın başına gidelim, belki değilizdir. */
262
263       while (cik != 1) {  /* Dosya sonu gelene kadar donulecek */
264
265           /* Her satir için gerekli değişkenleri sıfırlama BAŞLANGICI */
266           for (sayac=0;sayac<ONBELLEK;sayac++) {      /* Önbelleğe alınmadan önce, önbellek dizisi NULL'a ayarlanır. */
267               kelime_uzunlugu[sayac] = 0;
268           }
269
270           satiruz = 0;
271           bosluk_karakter = 0;
272           kac_kelime_sigacak = 0;
273           /* Her satir için gerekli değişkenleri sıfırlama SONU */
274
275           for (sayac=0;sayac<ONBELLEK;sayac++) {  /* Bir satırı doldurmaya yeterli kelime önbelleğe alınana kadar
    dönülür. Bu esanada diziye kelime uzunlukları kaydedilir (sonlarındaki boşluk dahil)*/
276
277               kelime_uzunlugu[sayac] = siradakikelime(fp,kelime);  /* Bir sonraki kelime okunur. */
278
279               if (kelime_uzunlugu[sayac] == -1) {  /* Dosya sonu gelmiş */
280                   cik = 1;
281                   break;
282               }
283
284               satiruz += kelime_uzunlugu[sayac];
285               if (satiruz_hedef < satiruz) {   /*Yeterli kelimeyi bulduk */
286                //   kac_kelime_sigacak = sayac;
287                   break;
288               }
289               kac_kelime_sigacak++; /* Satida kac kelime olacagini tutan degiskeni bir artiralim. */
290           }
291
292           if (cik!=1) {  /* Dosya sonu degilse */
293
294               gosterici_konum_son = ftell(fp) - kelime_uzunlugu[sayac]; /* Bir sonraki kelime nerde başlıyor? (dosya
    isaretçisini kaydedelim) */
295
296               for (sayac=0;sayac<kac_kelime_sigacak;sayac++) {          /* Kelimeler kac karakter tutuyor?   */
297                   bosluk_karakter += kelime_uzunlugu[sayac];
298               }
299               bosluk_karakter = satiruz_hedef - bosluk_karakter;        /* Kac fazladan bosluk kaldi? */
300
301               bosluksayisi(bosluk_karakter,kac_kelime_sigacak - 1,satirno % 2,boslukdizisi); /* Bosluklar kacar
    dagiliyor? */
302
303           }
304
305           // printf("%2d    ",satirno+1);    /* SATIR NUMARALARINI GOSTERMEK ICIN BU COMMENTI KALDIRIN */
306
307           fseek(fp,gosterici_konum_ilk,0); /* Dosya ilk konuma kaysin, cunku bu sefer ekrana yazdirmak icin tekrar
    okuyacagiz. */
```

```
308
309             for (sayac=0;sayac<=((79-satiruz_hedef)/2);sayac++) { /* Ortalayalim. */
310                 printf (" ");
311             }
312
313             for (sayac=0;sayac<kac_kelime_sigacak;sayac++) { /* Ekrana yazilacak kelime kadar don */
314                 siradakikelime(fp,kelime); /* Dosya konumundan kelimeyi al ve yaz (sonda bir bosluk dahil)*/
315                 printf ("%s",kelime);
316
317                 if (cik != 1)  { /* eger dosya sonu degilse daha once dizide bulunan degerler kadar bosluk ekle */
318                     for(sayac2 = 0;sayac2<boslukdizisi[sayac];sayac2++) {
319                         printf(" ");
320                     }
321                 }
322             }
323
324             satirno++;    /* satir numarasini arttir */
325             printf("\n"); /* satir atla              */
326
327             if (satirno == SATIRBOL) { /* eger kesilemsi gereken satir nosuna geldiysek */
328                 if(siradakikelime(fp,kelime)!= -1) { /* eger bu satirdan sonra da satirlar varsa (dosya sonundan onceki
    son kelimeyi ekrana yazdimadiysak) */
329                     satirno = 0;
330                     beklet();
331                 }
332             }
333
334             gosterici_konum_ilk = gosterici_konum_son; /* ana donguye son kalinan yerden devam edilecek */
335             fseek(fp,gosterici_konum_ilk,0); /*orayi neresiyse atla oraya */
336         }
337
338         beklet(); /* son satiri da yazinca kullanici bir tusa basana kadar bekle */
339     }
340
341     void istatistik(FILE *fp) {
342
343         /* Bu fonksiyon X karakterli Y tane kelime gectiginin bir istatistigini
344            ekrana yazar.
345         */
346
347         /* desigken tanimlamalari BASLANGICI */
348         int kelime_no[KELIMELIMIT-2];
349         int kelime_kactane[KELIMELIMIT-2] = {0};
350         char kelime[KELIMELIMIT];
351         int sayac;
352         int uzunluk;
353         /* desigken tanimlamalari SONU */
354
355         /* ilk deger atama BASLANGICI */
356         fseek(fp,0,0); /* Once dosyayi basa alalim. */
357         for (sayac=0;sayac<KELIMELIMIT-2;sayac++) {
358             kelime_no[sayac] = sayac + 1;
```

```
359         }
360         /* ilk deger atama SONU */
361
362         /* kac karakterli kelimeden kac tane oldugunu diziye alma BASLANGICI */
363         uzunluk = siradakikelime(fp,kelime);
364
365         while (uzunluk != -1) {  /* dosya bitene kadar dön */
366             kelime_kactane[uzunluk - 2]++; /* siradakikelime fonksiyonu kelime uzunlugunun bir fazlasini döndürür.
    (bosluk dahil olduğu için) Dizi de 0'dan başlar. Bundan dolayı indisten 2 çıkarıyoruz. */
367             uzunluk = siradakikelime(fp,kelime);
368         }
369         /* kac karakterli kelimeden kac tane oldugunu diziye alma SONU */
370
371         sirala(kelime_kactane,kelime_no); /* elde ettiğimiz diziyi kelime_kactane'ye gore sirala */
372
373         printf("*** Kelime uzunlugu istatistikleri ***\n");
374         printf("Harf Sayisi  Kez\n");
375         printf("-----------  ---\n");
376         for (sayac=0;sayac<KELIMELIMIT-2;sayac++) {
377             if (kelime_kactane[sayac] == 0) break;  // Eğer 0'lara geldiysek cik artik, daha sonra yeniden sayi gelmez,
    siraladik.
378             printf("%11d  %d\n",kelime_no[sayac],kelime_kactane[sayac]);
379         }
380
381         beklet();
382     }
383
384     void ara(FILE *fp, char *aranacak_kelime) {
385
386         /* fp ile gonderilen dosyada aranacak_kelime dizisindeki kelimeyi arar
387            ve buldugu sonuclara ekrana yazdirir.
388
389            fp ile acilmis dosya asagidaki ozelliklere sahip olmalidir:
390            * Satir sonu karakteri (ASCII:10) icermemelidir.
391            * Dosya bitiminde bosluk (ASCII:32) karakteri olmalidir.
392            Diger limitler Sabitler ve Anlamlari kismindaki aciklamalarda bulunabilir. Dosyanin odevde
393            verilenden yukaridaki ozelliklere sahip bir dosyaya donusturulmesi main() fonksiyonunda
394            yapilmistir.
395
396            aranacak_kelime dizisine sigacak maksimum karakter KELIMELIMITI-1'dir. Bu deger programa
397            girilebilinecek makisimum uzunluktaki kelimenin bir fazlasidir ve son karakter her zaman
398            NULL (ASCII:0)'dir.
399         */
400
401         /* ara fonksiyonu degisken tanimlamalari BASLANGICI */
402
403         char harf;
404         int gosterici_konumu;
405         int gosterici_temp;
406         int denetlenen_harf;
407         int bulamadim=0;
408         int dosyabitti=0;
```

```
409         char kelime[KELIMELIMIT];
410         int bosluk=0;
411         int kactane=0;
412         /* ara fonksiyonu degisken tanimlamalari SONU */
413
414         fseek(fp,0,0); /* Dosyayi basa alalim */
415
416         /* ekran giris BASLANGICI */
417         printf("*** Paragrafta Kelime Arama Sonuclari ***\n\n");
418         printf("Aranan kelime: %s\n",aranacak_kelime);
419         printf("       Kelime          Konum\n");
420         printf("--------------------  -----\n");
421         /* ekran giris SONU */
422
423         /* Asagidaki do while yapisinin icindeki arama algoritmasi genel olarak soyle islemektedir:
424
425            Dosyanin basindan baslayarak dosya harf harf okunur. Okunan ilk harf aranan kelimenin ilk
426            harfiyle karsilastirilir. Eger harfler tutuyorsa ikinci harfler, tutuyorsa ucuncu harfler...
427            seklinde aranan kelime dizisinde NULL karakterine varilana kadar karsilastirilir.
428            Eger hepsi tututyorsa bir kelime bulmusuzdur, ekrana yazilir. Ilk tutmayan harfte dosyadan
429            ikinci harf okunarak yukaridaki algoritma tekrarlanir.
430
431            Asagida algoritmanin nasil calistigina dair birkac ornek var:
432
433            METIN: umutb umudo ARANAN: mudo
434            u = m degil
435            m = m evet o halde;
436              u = u evet o halde; (kolay anlasilmasi icin girintili, asagidaki algoritmada bu bir alt algoritma degildir)
437              t = d degil, o zaman ilk harfi kontrol ettigim karakterden bir sonrasina donup devam edeyim. :(
438              DIKKAT: BURADA ayrica b = o degil karsilastirmasi YAPILMAZ. UYUSMAYAN ILK YERDE KONTROL BIRAKILIR.
    (etkinlik meselesi)
439           u = m degil
440           t = m degil
441           b = m degil
442             = m degil (yeni kelime baslangici oldugu icin dosya gostericisi konumu bellege kaydedilir)
443           u = m degil
444           m = m evet o halde;
445             u = u evet o halde;
446             d = d evet o halde;
447             o = o evet VE aranan kelime saglandigina gore bu bir SONUC'tur. Bu sonuc son bosluk (ASCII:32) karakterinden
448                 sonra baslayan kelimedir. Gerekli islemler yapilarak ekrana yazdirilir.
449
450            Asagidaki do while dongusu sozel olarak boyle anlatilan algoritmadir.
451         */
452
453         do {
454
455             denetlenen_harf = 0;
456             gosterici_konumu = ftell(fp);
457
458             while (aranacak_kelime[denetlenen_harf] != 0) {
459                 harf = fgetc(fp);
```

```
460                 if (harf == EOF) {
461                     dosyabitti = 1;
462                     break;
463                 }
464
465                 if (harf==32) {
466                     bosluk = ftell(fp);
467                 }
468
469                 if (harf != aranacak_kelime[denetlenen_harf]) {
470                     bulamadim = 1;
471                     break;
472                 }
473
474                 denetlenen_harf++;
475
476                 if(aranacak_kelime[denetlenen_harf] == 0) {
477                     kactane++;
478                     gosterici_temp = ftell(fp);
479                     fseek(fp,bosluk,0); /* kelimeyi okumak icin son bosluk karakterine don */
480                     siradakikelime(fp,kelime); /* kelimeyi oku */
481                     fseek(fp,gosterici_temp,0); /* sonra kelimeyi okumadan once dosya gostericisini nereden aldiysan
    oraya koy */
482                     printf("%-20s  %d\n",kelime,gosterici_konumu + 1);
483                 }
484
485             }
486
487             if (bulamadim == 1) {
488                 bulamadim = 0;
489                 fseek(fp,gosterici_konumu + 1,0);
490             }
491
492         } while (dosyabitti == 0);  /* Boşluk karakterine kadar dön */
493
494
495         if(kactane == 0) {
496             printf("(eslesme bulunamadi)\n");
497         } else {
498             printf("\nToplam tekrar sayisi: %d",kactane);
499         }
500
501         beklet();
502     }
503
504     int main() {
505
506         /* Acilis yazilari BASLANGICI */
507         printf("Umut BENZER\n");
508         printf("05-06-7670\n");
509         printf("Ege Universitesi Bilgisayar Muhendisligi 1. Sinif\n");
510         printf("http://www.ubenzer.com\n");
```

```
511         printf("Basit Bir Kelime Islemci 1.0\n\n");
512         /* Acilis yazilari SONU */
513
514         /* Dosyayı bir nedenden dolayı açamama durumunda hata mesajı verme de dahil olmak üzere dosya açma işlemi
    BASLANGICI */
515         FILE *fp;
516
517         if((fp=fopen("metin.dat","r")) == NULL) {           /* metin.dat: not alacagimiz yaznin oldugu asil dosya    */
518             printf ("Hata: metin.dat dosyasi acilamadi.");
519             return 0;
520         }
521
522         FILE *fp2;
523         if((fp2=fopen("gecici.tmp","w")) == NULL) {         /* gecici.tmp: metin.dat dosyasinin satir sonu
    karakteri          */
524             printf ("Hata: gecici dosya yaratilamadi.");    /* (ASCII:10) kaldirilmis ve en sonuna bosluk (ASCII:32)
    halidir. */
525             return 0;                                      /* Programim bu dosya uzerinde islem
    yapacak.                    */
526         }
527         /* Dosyayı bir nedenden dolayı açamama durumunda hata mesajı verme de dahil olmak üzere dosya açma işlemi SONU */
528
529         /* main fonksiyonu degisken tanimlamalari BASLANGICI */
530         int sag_serbest = -1;
531         char secenek;
532         char satiruz;
533         char metin[DOSYA];
534         char aranacak[KELIMELIMIT-1];
535         int sayac;
536         /* main fonksiyonu degisken tanimlamalari SONU */
537
538         /* metin.dat dosyasindakiler gecici.tmp dosyasina (ASCII:10) karakterini (ASCII:32)'ye çevirerek ve sona bir
    bosluk ekleyerek yaz */
539         while (!feof(fp)) {
540             fgets(metin, DOSYA, fp);
541             for (secenek=0;secenek<DOSYA;secenek++) {
542                 if (metin[secenek] == 10) {
543                     metin[secenek] = 32;
544                 }
545             }
546
547            fputs(metin,fp2);
548         }
549
550         fputc(32,fp2); /* en sona bir bosluk ekle */
551
552         fclose(fp);
553         fclose(fp2);
554         /* dosyalari kapat ve gecici.tmp'i okumak icin yeniden ac */
555
556          /* Dosyayı bir nedenden dolayı açamama durumunda hata mesajı verme de dahil olmak üzere dosya açma işlemi
    BASLANGICI */
```

```
557         if((fp=fopen("gecici.tmp","r")) == NULL) {
558             printf ("Hata: gecici dosya okunamadi.");
559             return 0;
560         }
561         /* Dosyayı bir nedenden dolayı açamama durumunda hata mesajı verme de dahil olmak üzere dosya açma işlemi SONU */
562
563         /* kullanici cikmak istemedigi surece bu dongude kal */
564         while (sag_serbest==-1) {
565
566             /* menu BASLANGICI */
567             printf("*** Menu *** (acilan dosya: metin.dat)\n\n");
568             printf("1. Paragrafi iki yana yaslayarak goruntuleme \n");
569             printf("2. Paragrafta kelime arama \n");
570             printf("3. Kelime uzunlugu istatistikleri \n");
571             printf("4. Cikis \n\n");
572
573             printf("Lutfen istediginiz islemin numarasini giriniz. ");
574             /*menu BITIMI */
575
576             secenek=getche();
577             system ("cls");
578
579             switch(secenek) {  /* char fonksiyonu girilen karakterin ASCII kodunu verir.          */
580                                /* Onun icin secenek degiskenini switche sayi olarak sokabiliyorum.*/
581                 case 49: //1
582                     /* Degisken tipi uyusmazligi haric hata ayiklamali satir uzunlugu alma kismi BASLANGICI */
583                     satiruz = 0;
584                     system ("cls");
585                     printf("*** Paragrafi Iki Yana Yaslayarak Goruntuleme ***\n\n");
586
587
588                     while (satiruz<SATIRMIN || satiruz > SATIRLIMIT-1) {
589                         printf("Satir uzulugu kac olsun? (%d",SATIRMIN);
590                         printf( " ile %d ",SATIRLIMIT-1);
591                         printf("arasi) \n");
592                         scanf("%d",&satiruz);
593                     }
594                     /* Degisken tipi uyusmazligi haric hata ayiklamali satir uzunlugu alma kismi SONU */
595                     system ("cls");
596                     justify(fp,satiruz + 1); /* Program tum kelimeleri sonunda bir bosluk var kabul ettigi icin
    aldigimiz satir uzunlugunun bir fazlasini gonderiyoruz. */
597                     sag_serbest = -1;
598                     break;
599                 case 50: //2
600                     /* Diziyi NULL'a ayarla */
601                     for (sayac=0;sayac<KELIMELIMIT-1;sayac++) {
602                         aranacak[sayac] = 0;
603                     }
604
605                     system ("cls");
606                     printf("*** Paragrafta Kelime Arama ***\n\n");
607
```

```
608                     printf("(En fazla %d karakterlik sadece kucuk harf iceren TEK kelime olmalidir.)\n",KELIMELIMIT-2);
609                     printf("Aranacak kelime: ");
610                     scanf("%s",aranacak);
611                     system ("cls");
612                     ara(fp,aranacak);
613                     system ("cls");
614                     sag_serbest = -1;
615                     break;
616                 case 51: //3
617                     system ("cls");
618                     istatistik(fp);
619                     system ("cls");
620                     sag_serbest = -1;
621                     break;
622                 case 52: //4
623                     /* E e H h ve disinda girdi kabul etmeyen cikmak istiyor musunuz sorusu BASLANGICI */
624                     do {
625                         printf ("\nCidden cikmak istiyor musunuz? (E/H) ");
626                         secenek = getche();
627                         if (secenek == 'E' || secenek == 'e') {
628                             sag_serbest = 0;
629                             secenek = 'H';
630                         }
631                     } while (secenek != 'H' && secenek != 'h');
632                     /* E e H h ve disinda girdi kabul etmeyen cikmak istiyor musunuz sorusu SONU */
633                     system ("cls");
634                     break;
635                 default:
636                     /* Elleri yanlis tusa basan kullanicilar icin yazilan bolum BASLANGICI */
637                     printf("Lutfen 1'den 4'e kadar bir secim yapin.");
638                     beklet();
639                     /* Elleri yanlis tusa basan kullanicilar icin yazilan bolum SONU */
640                     break;
641             }
642         }
643 
644         printf("O halde hoscakalin. :)\n\n\n");
645         return 0;
646     }
647 
```