İşletim Sistemleri Dönem Projesi

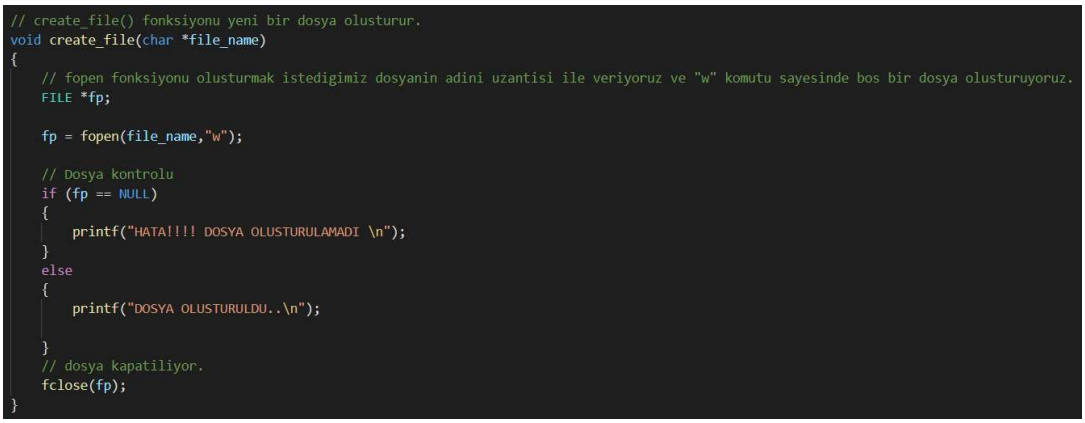
Nihai Rapor

Bu projede, C programlama dilinde bir dosya yöneticisi uygulaması tasarladık. Tasarladığımız dosya yöneticisi uygulaması içinde C programlama derleyicisi olan herhangi bir işletim sisteminde çalışabilmektedir.

TASARIM ANLATIM

1) create_file()

Girdi olarak dosya adı alır ve bunu pass by reference sayesinde adresini alarak işlem yapar. "FILE *" veri yapısı oluşturarak, fopen() fonksiyonu sayesinde girdi olarak aldığımız veriyi kullanarak dosyamızı "w" modunda açıyoruz bu sayede dosyayı oluşturmuş oluyoruz. Sonrasında ise kontrollerimizi yaparak fonksiyonu sonlandırıyoruz. fclose() ile dosyayı kapatıyoruz.



```c
// create_file() fonksiyonu yeni bir dosya olusturur.
void create_file(char *file_name)
{
    // fopen fonksiyonu olusturmak istedigimiz dosyanin adini uzantisi ile veriyoruz ve "w" komutu sayesinde bos bir dosya olusturuyoruz.
    FILE *fp;

    fp = fopen(file_name,"w");

    // Dosya kontrolu
    if (fp == NULL)
    {
        printf("HATA!!! DOSYA OLUSTURULAMADI \n");
    }
    else
    {
        printf("DOSYA OLUSTURULDU..\n");

    }
    // dosya kapatiliyor.
    fclose(fp);
}
```

2) delete_file()

delete_file() girdi olarak char dosya adı alır ve bunu pass by reference sayesinde adresini alarak işlem yapar. remove() fonksiyonuna dosya adını vererek dosyayı silme işlemini gerçekleştiririz ve sonrasında kontrolleri yaparız ve fonksiyonu sonlandırırız.

```
// delete_file() fonksiyonu var olan dosyayi siler.
void delete_file(char *file_name)
{
    // remove() fonksiyonu kullanarak silmek istedigimiz dosyanin adini uzantisi ile birlikte giriyoruz ve dosyamiz siliniyor.
    if(remove(file_name)== 0 )
    {
        printf("DOSYA BASARI ILE SILINDI..\n");
    }
    else
    {
        printf("HATA!!!! DOSYA SILINEMEDI !\n");
    }

}
```

3) rename_file()

rename_file() girdi olarak "dosya adı" ve "yeni dosya adı" alır ve bunu pass by reference sayesinde adresini alarak işlem yapar.İşlem kısmında rename() fonksiyonu sayesinde dosyanın adı ve yeni dosya adı'nı vererek dosyanın adını değiştirir ve sonrasında kontroller yapılarak fonksiyon sonlandırılır.

```
// rename_file() fonksiyonu var olan dosyanin ismini degistirir.
void rename_file(char *file_name, char *file_rename)
{
    // rename() fonksiyonu ile adini değiştirmek istediğimiz dosyanin adini ve yeni adini arguman olarak vererek dosyanin adini kolayca degistiriyoruz.
    int re = rename(file_name, file_rename);

    // rename kontrolu
    if(re == 0)
    {
        printf("DOSYA ADI BASARI ILE DEGISTIRILDI..\n");

    }
    else
    {
        printf("HATA!!!! DOSYA ADI DEGISTIRILEMEDI.\n");
    }


}
```

4) file_copying()

file_copying() girdi olarak "dosya adı" ve "kopyalanacak dosya adı" alır ve bunu pass by reference sayesinde adresini alarak işlem yapar.İşlem kısmında "FILE" veri yapısı kullanılarak dosyalar "r" ve "w" modunda açılarak dosyaların açılıp açılmadığı kontrol edilir ve sonrasında dosya kopyalama while döngüsü sayesinde gerçekleşir ve dosyalar kapatılıp fonksiyon sonlandırılır.

```
// file_copying() fonksiyonu dosya kopyalama islemini gerceklestirir.
void file_copying(char *file_name, char *copy_name )
{
    char ch;
    FILE *source, *target;

    // fopen ile dosyayi okuma modunda aciyoruz.
    source = fopen(file_name, "r");

    // dosya acilmazsa ekrana yazdirilacak mesaj.
    if (source == NULL) {
        printf("HATALI DOSYA ADI...\n");
        exit(EXIT_FAILURE);
    }

    //fopen ile yeni kopya dosyasini yazma modunda aciyoruz içinde değer varsa silinsin ya da dosya yoksa olusturulsun diye.
    target = fopen(copy_name, "w");

    // dosya kontrolu
    if (target == NULL) {
        fclose(source);
        printf("HATAA...\n");
        exit(EXIT_FAILURE);
    }

    //fgetc ile dosyanin sonuna kadar giderek alinan tek tek degerleri diger dosyaya ekliyoruz.
    while ((ch = fgetc(source)) != EOF)
        fputc(ch, target);

    printf("DOSYA KOPYALAMA BAŞARILI...\n");

    //dosyalar kapatiliyor...
    fclose(source);
    fclose(target);
}
```

5) move_file()

move_file() fonksiyonuna taşımak istediğimiz dosyanın bulunduğu dizini (dosyanın ismi ile birlikte) ve taşınmasını istediğimiz dizini vererek çağırıyoruz. rename() fonksiyonu ile dosyanın yerini değiştiriyoruz. Kontrolleri yapıyoruz ve fonksiyon bitiyor.

```c
// move_file() fonksiyonu dosya tasima islemi gerceklestirir.
void move_file(char *oldDirectory, char *newDirectory)
{
    // rename fonksiyonu ile dosyayi baska bir dizine tasima
    int re = rename(oldDirectory, newDirectory);

    // rename kontrolu
    if(re == 0)
    {
        printf("DOSYA BASARILI BIR SEKILDE TASINDI...\n");

    }
    else
    {
        printf("HATA! DOSYA TASINAMADI. \n");
    }
}
```

6) add_text()

add_text() girdi olarak "dosya adı" ve "eklenecek text" alır ve bunu pass by reference sayesinde adresini alarak işlem yapar.İşlem kısmında "FILE" , " char " veri yapıları önce dosyayı "r" modunda açarak sonrada txt dosyasının sonuna giderek eklemek istediğimiz texti oraya fputs() fonksiyonu sayesinde ekleyip dosyamızı kapatıp fonksiyonumuzu sonlandırıyoruz.

```c
// add_text() fonksiyonu kullanicidan aldigi metni dosya sonuna ekler.
void add_text(char *file_name,char *text_add)
{
    FILE *fp;
    char ch;

    // fopen fonksiyonu olusturmak istedigimiz dosyanin adini uzantisi ile veriyoruz ve "a+"" komutu sayesinde dosyayi okuma ve yazma modunda aciyoruz.
    fp = fopen(file_name, "a+");

    // dosya kontrolu
    if(fp == 0)
    {
        printf("HATAA DOSYA AÇILAMADI..!\n");
    }

    // dosyanin sonuna gidiyoruz
    while (!feof(fp))
    {
        ch = fgetc(fp);
    }


    //int i=0;
    fputs(text_add, fp);
    fclose(fp);
    printf("DOSYA ICINE YAZMA BASARILI.");
}
```

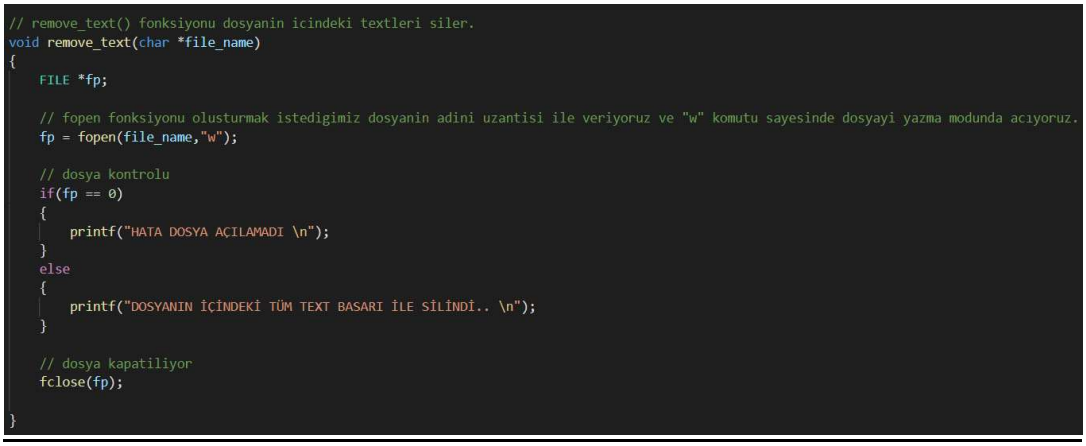
7) location_add_text()

location_add_text() fonksiyonu girdi olarak "dosya adı" , "texti kaçıncı konuma eklemek istiyorsa (konumlar karakter sayısı olarak sayılmalıdır)" , "eklenecek text " ve "yeni dosya adı(kullanıcıdan istenmez)"alır ve bunu pass by reference sayesinde adresini alarak işlem yapar.İşlem kısmında "FILE","char","int" veri yapıları kullanılıp öncelikle dosyayı "r" ve "w" modunda açarak işlemler ve kontroller yapılır sonrasında ise while döngüsü sayesinde dosyanın belirli konumuna kadar ilerleyip yeni dosyaya kopyalıyıp sonrasında eklenecek texti ekleyip kopyalama işlemine devam edip bitiriyor sonrasında yeni dosya adını eskisiyle değiştiriyoruz ve elimizde kullanıcının girdiği dosya oluyor en son olarak dosyaları kapatıp fonksiyonu sonlandırıyoruz.

```c
// location_add_text() fonksiyonu dosyanin istenilen yerine metin eklemeyi saglar.
void location_add_text(char *file_name, int location, char *new_file_name, char *text_add)
{
    char ch;
    FILE *source, *target;
    int sayac  = 0;

    // fopen fonksiyonu olusturmak istedigimiz dosyanin adini uzantisi ile veriyoruz ve "r" komutu sayesinde dosyayi okuma modunda acıyoruz.
    source = fopen(file_name, "r");

    // dosya kontrolu
    if (source == NULL) {
        printf("HATALI DOSYA ADI...\n");
        exit(EXIT_FAILURE);
    }
    // fopen fonksiyonu olusturmak istedigimiz dosyanin adini uzantisi ile veriyoruz ve "w" komutu sayesinde dosyayi yazma modunda acıyoruz.
    target = fopen(new_file_name, "w");

    // dosya kontrolu
    if (target == NULL) {
        fclose(source);
        printf("HATAA...\n");
        exit(EXIT_FAILURE);
    }

    int value = 0;
    // dosyada kullanicinin belirttigi yere text ekleme
    while ((ch = fgetc(source)) != EOF)
    {
        if(sayac == (location)) // 10 ya
        {
            fputs(text_add, target);
            sayac++;
        }
        fputc(ch, target);
        sayac++;
    }
    rename(new_file_name, file_name);
    // dosyalar kapatiliyor
    fclose(source);
    fclose(target);
}
```

8) remove_text()

remove_text() fonksiyonuna silinmesini istediğimiz dosyanın ismini vererek çağırıyoruz. İşlemi pass by reference sayesinde adresi alarak tamamlar. Aldığımız dosyayı FILE * tipinde tutuyoruz. fopen() fonksiyonunu 'w' modunu kullanarak açıyoruz. Dosya 'w' modunda açılırsa overwrite ettiği için dosyanın içeriğini siliyor. Gerekli kontrolleri yaptıktan sonra dosyayı fclose() ile kapatıyoruz.



```c
// remove_text() fonksiyonu dosyanin icindeki textleri siler.
void remove_text(char *file_name)
{
    FILE *fp;

    // fopen fonksiyonu olusturmak istedigimiz dosyanin adini uzantisi ile veriyoruz ve "w" komutu sayesinde dosyayi yazma modunda acıyoruz.
    fp = fopen(file_name,"w");

    // dosya kontrolu
    if(fp == 0)
    {
        printf("HATA DOSYA AÇILAMADI \n");
    }
    else
    {
        printf("DOSYANIN İÇİNDEKİ TÜM TEXT BASARI İLE SİLİNDİ.. \n");
    }

    // dosya kapatiliyor
    fclose(fp);

}
```

9) line_display_text()

line_display_text() fonksiyonu girdi olarak "dosya adı" , "kac tane satir ekrana yazdirmak istiyorsaniz integer bir deger" alır ve bunu pass by reference sayesinde adresini alarak işlem yapar.İşlem kısmında ise "FILE" ,"char","int" veri yapıları kullanarak öncelikle dosyamızı "r" modunda açıyoruz sonrasında ise while döngüsü ile başlangıç kısmına geliyoruz başlangıç kısmına geldiğimizde ise while döngüsüyle kullanıcının girdiği satır sayısı kadar ekrana yazdırıyoruz ve kullanıcıya ekrana sonrakı girdiği satır sayısı kadar satır yazdırmak için soru soruyoruz ve eğer "1" değerini girerse line_display_text fonksiyonunu çağırarak sonraki girilen satır sayısı kadar satırı ekrana yazar ancak "1" dışında herhangi bir değer girerse fonksiyon sonlandırılır.

```c
void line_display_text(char *file_name, int location, int begin, int increment)
{
    FILE *fp;
    // fopen fonksiyonu olusturmak istedigimiz dosyanin adini uzantisi ile veriyoruz ve "r" komutu sayesinde dosyayi okuma modunda acıyoruz.
    fp = fopen(file_name,"r");

    // dosya kontrolu
    if(fp == 0)
    {
        printf("HATA DOSYA AÇILAMADI \n");
    }

    char content;
    int counter = 0;

    // print edilecek satirlarin baslangic noktasina ilerleme
    while ((content = fgetc(fp)) != EOF)
    {
        if (counter == begin)
        {
            break;
        }
        else if (content == '\n') counter++;
    }

    // kullanicinin girdigi sayi kadar satiri ekrana yazdiriyor.
    while ((content = fgetc(fp)) != EOF && begin < location)
    {
        if (content == '\n') begin++;
        putchar(content);
    }

    char cmd[255];
    printf("DIGER %d SATIRI GORMEK ISTERSENIZ 1 YOKSA CIKIS YAPMAK ICIN HERHANGI BIR DEGER GIRINIZ: ", increment);
    fgets(cmd, 255, stdin);
    cmd[strcspn(cmd, "\n")] = '\0';

    // kullanici "1" girerse tekrar belirttigi sayida satiri ekrana yazdirir
    if(!strcmp(cmd, "1"))
    {
        line_display_text(file_name, (location + increment), location, increment);
    }
    else
    {
        printf("CIKIS YAPILDI.\n");
    }
}
```

10) main()

main fonksiyonu "char *", "char []" ,"char *[]","int" veri yapıları kullanılarak işlemleri yaparız.İşlem kısmında while döngüsün kullanıcı "exit" girine kadar kullanıcıdan input almaya devam edecek şekilde tasarlanmıştır ve kullanıcının girdiği inputu " " karakterine göre ayırarak komut sistemi oluşması sağlanmıştır.

(260-294).satır arası : Kullanıcıdan aldığımız inputu "char []" veri yapısına atanır. Sonrasında strtok() fonksiyonu sayesinde boşluklara göre aldığımız input'u ayırarak "char *" veri yapısına atarız. Ardından for döngüsü sayesinde "char *" veri yapısındaki değeri "char * []" veri yapısındaki değişkene eşitleyip if koşulunda strcmp() fonksiyonu sayesinde komut adıyla karşılaştırıp gerekli koşulları kontrol ettikten sonra kullanıcının istediği komutu yerine getiririz.

```
260  int main()
261  {
262      char cmd[255];
263      char *tokens[20];
264      char *text;
265
266      printf("-----------------------------------------------------------------------------------------------\n--
267
268      memset(cmd, 0, 255);
269
270      // kullanici "exit" komutunu girene kadar dongu devam eder.
271      while (strcmp(cmd, "exit"))
272      {
273          // read
274          char *token;
275          fgets(cmd, 255, stdin);
276          cmd[strcspn(cmd, "\n")] = '\0';
277          token = strtok(cmd, " "); // inputu bosluklara gore ayirarak adresini token pointer'ina atiyoruz.
278
279          // token pointer'i adreslerini pointer bir diziye atiyoruz
280          for (int i=0; i<20; i++)
281          {
282              tokens[i] = token;
283              token = strtok(NULL, " ");
284          }
285
286          // tokens arrayindeki char degiskenini komut ismi ile karsilastirir.
287          if (!strcmp(tokens[0], "create_file"))
288          {
289              // komut kontrolu
290              if(tokens[1] == NULL)
291              {
292                  printf("Lutfen komutu dogru girdiginizden emin olunuz.\n");
293              }
294
```

```
295             // help ozelligi
296             else if(!strcmp(tokens[1], "/h"))
297             {
298                 printf("---------------------------create_file help---------------------------\n'k
299             }
300             //Help kontrolu
301             else if(!strcmp(tokens[1], "/H"))
302             {
303                 printf("Lutfen komutu dogru girdiginizden emin olunuz.\n");
304             }
305             else
306             {
307                 create_file(tokens[1]);
308             }
309         }
310 
311         // tokens arrayindeki char degiskenini komut ismi ile karsilastirir.
312         else if (!strcmp(tokens[0], "delete_file"))
313         {
314             // help ozelligi
315             if (tokens[1] == NULL)
316             {
317                 printf("Lutfen komutu dogru girdiginizden emin olunuz.\n");
318             }
319             else if(!strcmp(tokens[1], "/h"))
320             {
321                 printf("---------------------------delete_file help---------------------------\n'k
322             }
323             //Help kontrolu
324             else if(!strcmp(tokens[1], "/H"))
325             {
326                 printf("Lutfen komutu dogru girdiginizden emin olunuz.\n");
327             }
```

```
328             else
329             {
330                 delete_file(tokens[1]);
331             }
332         }
333 
334         // tokens arrayindeki char degiskenini komut ismi ile karsilastirir.
335         else if (!strcmp(tokens[0], "rename_file"))
336         {
337             // komut kontrolu
338             if(tokens[1] == NULL)
339             {
340                 printf("Lutfen komutu dogru girdiginizden emin olunuz.\n");
341             }
342             // help ozelligi
343             else if(!strcmp(tokens[1], "/h"))
344             {
345                 printf("---------------------------rename_file help---------------------------\n'k
346             }
347             // komut kontrolu
348             else if (tokens[2] == NULL)
349             {
350                 printf("Lutfen komutu dogru girdiginizden emin olunuz.\n");
351             }
352             else
353             {
354                 rename_file(tokens[1], tokens[2]);
355             }
356         }
357 
358         // tokens arrayindeki char degiskenini komut ismi ile karsilastirir.
```

```
358         // tokens arrayindeki char degiskenini komut ismi ile karsilastirir.
359         else if (!strcmp(tokens[0], "copy_file"))
360         {
361             // komut kontrolu
362             if(tokens[1] == NULL)
363             {
364                 printf("Lutfen komutu dogru girdiginizden emin olunuz.\n");
365             }
366             // help ozelligi
367             else if(!strcmp(tokens[1], "/h"))
368             {
369                 printf("----------------------------copy_file help----------------------------\n'komu
370             }
371             // komut kontrolu
372             else if (tokens[2] == NULL)
373             {
374                 printf("Lutfen komutu dogru girdiginizden emin olunuz.\n");
375             }
376             else
377             {
378                 file_copying(tokens[1], tokens[2]);
379             }
380         }
381 
382         // tokens arrayindeki char degiskenini komut ismi ile karsilastirir.
383         else if (!strcmp(tokens[0], "move_file"))
384         {
385             // komut kontrolu
386             if(tokens[1] == NULL)
387             {
388                 printf("Lutfen komutu dogru girdiginizden emin olunuz.\n");
389             }
```

```
390             // help ozelligi
391             else if(!strcmp(tokens[1], "/h"))
392             {
393                 printf("----------------------------move_file help----------------------------\n'komu
394             }
395             // komut kontrolu
396             else if (tokens[2] == NULL)
397             {
398                 printf("Lutfen komutu dogru girdiginizden emin olunuz.\n");
399             }
400             else
401             {
402                 move_file(tokens[1], tokens[2]);
403             }
404         }
405 
406         // tokens arrayindeki char degiskenini komut ismi ile karsilastirir.
407         else if (!strcmp(tokens[0], "add_text"))
408         {
409             // komut kontrolu
410             if(tokens[1] == NULL)
411             {
412                 printf("Lutfen komutu dogru girdiginizden emin olunuz.\n");
413             }
414             // help ozelligi
415             else if(!strcmp(tokens[1], "/h"))
416             {
417                 printf("----------------------------add_text help----------------------------\n'komu
418             }
419             // komut kontrolu
420             else if (tokens[2] == NULL)
421             {
422                 printf("Lutfen komutu dogru girdiginizden emin olunuz.\n");
423             }
```

```
424             else
425             {
426                 int i = 2;
427 
428                 // tokens'in elemani NULL olana kadar tokens'e girilen elamanlarini text pointer'ina
429                 while(tokens[i] != NULL)
430                 {
431                     text = *(tokens + i);
432                     i++;
433                     add_text(tokens[1], text);
434                 }
435             }
436         }
437 
438         // tokens arrayindeki char degiskenini komut ismi ile karsilastirir.
439         else if (!strcmp(tokens[0], "remove_text"))
440         {
441             // komut kontrolu
442             if(tokens[1] == NULL)
443             {
444                 printf("Lutfen komutu dogru girdiginizden emin olunuz.\n");
445             }
446             // help ozelligi
447             else if(!strcmp(tokens[1], "/h"))
448             {
449                 printf("---------------------------remove_text help---------------------------\n'k
450             }
451             //Help kontrolu
452             else if(!strcmp(tokens[1], "/H"))
453             {
454                 printf("Lutfen komutu dogru girdiginizden emin olunuz.\n");
455             }
456             else
457             {
458                 remove_text(tokens[1]);
459             }
```

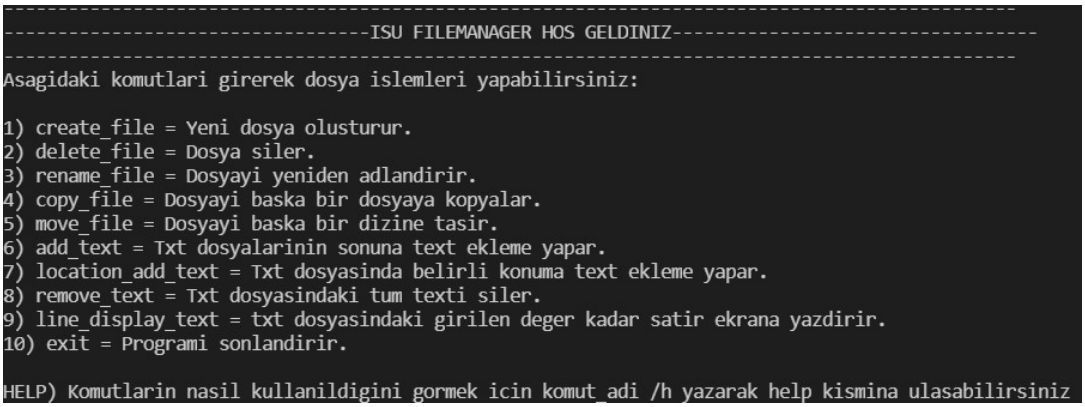
(464-500). satır arası: Kullanıcıdan aldığımız veriyi kontrol ettikten sonra fonksiyona göndermeden önce sscanf() fonksiyonu sayesinde "char" veri yapısını "int" veri yapısına çeviriyoruz. Sonrasında while döngüsü sayesinde "char * []"nin elemanı NULL olana kadar dizide olanları "char *" veri yapısına eşitliyoruz. Sonrasında location_add_text() fonksiyonunu çağırarak "char *" veri yapısındaki değeri kullanıdan girilen location'a yazıyoruz. Ardından while dongüsünde counter'ı kullanıcının girdiği text'i strlen() fonksiyonu sayesinde uzunluğunu alıp location'a ekliyoruz. Bunun sayesinde girilen girdinin tersten yazılmasını engelliyoruz. Örnek olarak: "Mevlüt Yaşar" eklenmek istenirse "Yaşar Mevlüt" şeklinde yazılmaması için bu kullanılır.

```
464         // tokens arrayindeki char degiskenini komut ismi ile karsilastirir.
465         else if (!strcmp(tokens[0], "location_add_text"))
466         {
467             // komut kontrolu
468             if (tokens[1] == NULL)
469             {
470                 printf("Lutfen komutu dogru girdiginizden emin olunuz.\n");
471             }
472             // help ozelligi
473             else if(!strcmp(tokens[1], "/h"))
474             {
475                 printf("---------------------------location_add_text help---------------------------\n'komut_adi' bosluk birakarak uzantisiyla beraber 'dosya adi', har
476             }
477             // komut kontrolu
478             else if(tokens[2] == NULL && tokens[3] == NULL)
479             {
480                 printf("Lutfen komutu dogru girdiginizden emin olunuz.\n");
481             }
482             else
483             {
484                 int location = 0;
485                 int counter = 0;
486                 sscanf(tokens[2], "%d", &location); // string girilen degeri integer'a ceviriyoruz.
487 
488                 int i = 3;
489                 // tokens'in elemani NULL olana kadar tokens'e girilen elamanlarini text pointer'ina yazar ve location_add_text fonksiyonunu cagirir.
490                 while(tokens[i] != NULL)
491                 {
492                     text = *(tokens + i);
493                     i++;
494                     location_add_text(tokens[1], (location + counter), "a.txt", text);
495                     counter += strlen(text);
496                 }
497                 printf("BELIRTILEN LOKASYONA EKLEME BASARILI.\n");
498             }
499         }
500 
```

```
495
496         // tokens arrayindeki char degiskenini komut ismi ile karsilastirir.
497         else if(!strcmp(cmd, "line_display_text"))
498         {
499             // komut kontrolu
500             if(tokens[1] == NULL)
501             {
502                 printf("Lutfen komutu dogru girdiginizden emin olunuz.\n");
503             }
504             // help ozelligi
505             else if(!strcmp(tokens[1], "/h"))
506             {
507                 printf("----------------------------line_display_text help----------------------------\n'komut_adi' bo
508             }
509             else if(tokens[2] == NULL)
510             {
511                 printf("Lutfen komutu dogru girdiginizden emin olunuz.\n");
512             }
513             else
514             {
515                 int location = 0;
516                 int begin = 0;
517                 sscanf(tokens[2], "%d", &location);
518                 int increment = location;
519                 line_display_text(tokens[1], location, begin, increment);
520             }
521         }
522 
523         // kullanici "exit" komutunu girerse program sonlanir.
524         else if(!strcmp(tokens[0], "exit"))
525         {
526             break;
527         }
528 
529         // kullanicin hatali giris yapmasi durumu
530         else
531         {
532             printf("HATALI GIRIS YAPTINIZ TEKRAR DENEYINIZ.\n");
533         }
534 
535     }
```

KULLANIM KILAVUZU

1) Isu File Manager Programı Giriş Ekranı



2) create_file

'komut_adi' bosluk birakarak uzantisiyla beraber 'dosya adi' giriniz.

Örnek olarak: 'create_file metin.txt'

```
create_file hello.txt
DOSYA OLUSTURULDU..
```

3) delete_file

'komut_adi' bosluk birakarak uzantisiyla beraber 'dosya adi' giriniz.

Örnek olarak: 'delete_file metin.txt'

```
delete_file hello.txt
DOSYA BASARI ILE SILINDI..
```

4) rename_file

'komut_adi' bosluk birakarak uzantisiyla beraber 'dosya adi' ve bosluk birakarak uzantisiyle 'yeni dosya adi' giriniz.

Örnek olarak: 'rename_file metin.txt yeni_metin.txt'

```
rename_file hello.txt newName.txt
DOSYA ADI BASARI ILE DEGISTIRILDI..
```

5) copy_file

'komut_adi' bosluk birakarak uzantisiyla beraber kopyasini olusturmak istediginiz 'dosya adi' ve bosluk birakarak kopyalanacak 'dosyanin adi' giriniz.

Örnek olarak: 'copy_file metin.txt yeni_metin.txt'

```
copy_file newName.txt hello.txt
DOSYA KOPYALAMA BAŞARILI...
```

6) move_file

'komut_adi' bosluk birakarak path ve uzantisiyla beraber 'dosya adi' ve bosluk birakarak yeni path 'dosya adi' giriniz.

Örnek olarak: 'move_file path/metin.txt metin.txt'

```
move_file hello.txt texts/hello.txt
DOSYA BASARILI BIR SEKILDE TASINDI...
```

7) add_text

'komut_adi' bosluk birakarak uzantisiyla beraber 'dosya adi' ve bosluk birakarak eklemek istediginiz 'text'i' giriniz.

Örnek olarak: 'add_text metin.txt Mevlut Yasar.'

```
add_text newName.txt hello
DOSYA ICINE YAZMA BASARILI.
```

8) location_add_text

'komut_adi' bosluk birakarak uzantisiyla beraber 'dosya adi', hangi lokasyona eklemek istiyorsaniz (karakter sayisi olarak), 'yeni dosya adi', 'eklemek istediginiz text'i' giriniz.

Örnek olarak: 'location_add_text metin.txt 20 yeni_metin.txt Dogukan Nadir.'

```
location_add_text newName.txt 10 newName1.txt Merhaba
BELIRTILEN LOKASYONA EKLEME BASARILI.
```

9) remove_text

'komut_adi' bosluk birakarak uzantisiyla beraber 'dosya adi' giriniz.

Örnek olarak: 'remove_file metin.txt'

```
remove_text newName11.txt
DOSYANIN İÇİNDEKİ TÜM TEXT BASARI İLE SİLİNDİ..
```

10) line_display_text

'komut_adi' bosluk birakarak uzantisiyla beraber 'dosya adi', kac tane satir ekrana yazdirmak istiyorsaniz integer bir deger giriniz.

Örnek olarak: 'line_display_text metin.txt 10 '

```
line_display_text metin.txt 3
deneme1
deneme12
deneme13
DIGER 3 SATIRI GORMEK ISTERSENIZ 1 YOKSA CIKIS YAPMAK ICIN HERHANGI BIR DEGER GIRINIZ: 1
deneme14
deneme15
deneme16
DIGER 3 SATIRI GORMEK ISTERSENIZ 1 YOKSA CIKIS YAPMAK ICIN HERHANGI BIR DEGER GIRINIZ: 1
deneme17
deneme18
deneme19
DIGER 3 SATIRI GORMEK ISTERSENIZ 1 YOKSA CIKIS YAPMAK ICIN HERHANGI BIR DEGER GIRINIZ: exit
CIKIS YAPILDI.
```

11) exit

```
exit
Exiting the command line.
```

12) /h (help)

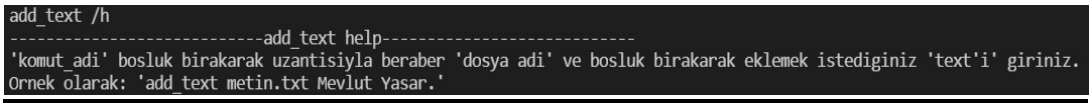
```
create_file /h
------------------------------create_file help------------------------------
'komut_adi' bosluk birakarak uzantisiyla beraber 'dosya adi' giriniz.
Ornek olarak: 'create_file metin.txt'
```

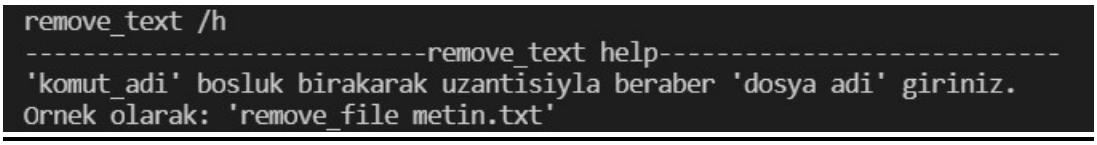
```
delete_file /h
------------------------------delete_file help------------------------------
'komut_adi' bosluk birakarak uzantisiyla beraber 'dosya adi' giriniz.
Ornek olarak: 'delete_file metin.txt'
```

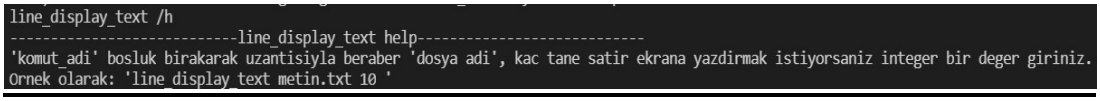
```
rename_file /h
----------------------------rename_file help----------------------------
'komut_adi' bosluk birakarak uzantisiyla beraber 'dosya adi' ve bosluk birakarak uzantisiyle 'yeni dosya adi' giriniz.
Ornek olarak: 'rename_file metin.txt yeni_metin.txt'
```

```
copy_file /h
----------------------------copy_file help----------------------------
'komut_adi' bosluk birakarak uzantisiyla beraber kopyasini olusturmak istediginiz 'dosya adi' ve bosluk birakarak kopyalanacak 'dosyanin adi'ni'  giriniz.
Ornek olarak: 'copy_file metin.txt yeni_metin.txt'
```

```
move_file /h
----------------------------move_file help----------------------------
'komut_adi' bosluk birakarak path ve uzantisiyla beraber 'dosya adi' ve bosluk  birakarak yeni path 'dosya adi' giriniz.
Ornek olarak: 'move_file path/metin.txt metin.txt'
```







NOT: Demo sunumu sırasında location_add_text() fonksiyonundaki pürüz giderildi. Kaynak dosyasına ve rapora eklendi.

Mevlüt Yaşar 190701121

Kadirhan Şahin 190701115

Doğukan Nadir 190701096