Ben G201210085 numaralı 2A grubu öğrencisi Eda Eren. Raporda kısaca bizden istenen bir txt dosyasındaki verilerin okunarak eğer ekleme ise ekleme fonksiyonu ile en sona bir düğüm eklenip istenilen düğüme kadar giderken verileri her seferinde bir sağa kopyalamak; eğer silme ise de belirtilen indexin sağındaki düğümdeki datayı bir soldakine kopyalamak ve böylece sağa doğru ilerledikten sonra en sondaki düğüm silmek.

İlk olarak bir "Node" isimli bir düğüm sınıfı oluşturdum ve bu sınıfın içerisinde public kısımda girilen veri için data isimli bir string, düğümün kendisinden sonra göstereceği adres için next isimli bir pointer ve düğümün kendisinden önce göstereceği adres için de prev isimli başka bir pointer tanımladım. Node isimli fonksiyonda da gelen değerleri alarak bu fonksiyonda bir düğüm oluşturmasını sağladım.

Node.cpp'de bu fonksiyonda gelen bilgiyi data isimli string değer tipindeki değişkene atadım. Düğümden sonraki adresi gösterecek pointer değerini parametre ile nx pointerı ile alıp fonksiyon gövdesinde next'e atadım. Düğümden önceki adresi gösterecek pointer değerini de pr ile alıp prev isimli pointer'a atadım.

List.hpp'de private kısımda ilk düğümü göstermek üzere head isimli bir düğüm pointerı tanımladım. Listenin uzunluğunu atayacağım integer tipinde bir size değişkeni, listedeki son düğümü bulmak üzere kullanacağım bir FindLastNode fonksiyonu ve listede girilen indexteki düğümü bulabilmemi sağlayacak, integer tipinde bir adet parametre alan FindPresentNode isimli fonksiyonu tanımladım.

List.hpp'nin public alanında ise List isimli bir yapıcı fonksiyon, listenin boş mu dolu mu olduğunu anlayabilmemi sağlayacak bir isEmpty fonksiyonu; girilen veri dosyasındaki değerleri listeye eklememi sağlayacak, bir integer ve bir string olmak üzere iki parametreli bir add fonksiyonu, integer tipinde değişken alan ve girilen veriyi silmeyi sağlayan bir remove fonksiyonu, ekrana yazmayı sağlayan bir fonksiyon, listeyi temizleme fonksiyonu ve listeyi tamamen yıkan bir yıkıcı fonksiyon tanımladım.

List.cpp'de ise bu fonksiyonların gövdesini yazdım. İlk olarak List.hpp'yi buraya dahil ettim. Daha sonra FindLastNode isimli fonksiyonu yazmaya başladım. Bu fonksiyonda ilk olarak prev isimli Node tipinde bir pointer oluşturdum ve bu pointerın head'in adresini göstermesini sağladım. Daha sonra for döngüsü içinde itr isimli bir pointer daha tanımladım. Bu pointer bizim düğümde ilerlememizi sağlayacak. Bu pointer'a da headin adresini atadım ve itr'nin next'i null olmadığı sürece for döngüsünün içine girmesini sağladım. Bu for döngüsünde prev'in nextini prev'e atayarak gösterilen adresi bir ilerletmiş oldum ve itrnin nextini itr'ye atarayarak itr'yi de ilerletip döngünün devamlılığını sağladım. En sonda bulunan değeri de return kullanarak döndürdüm ve böylece listedeki son düğümü döndürdüm.

FindPresentNode isimli integer tipinde parametre alan fonksiyonda ise, girilen index değerindeki düğümü bulmayı amaçladım. İlk olarak yine presentNODE isimli bir pointer tanımladım ve bunun da head'in adresini göstermesini sağladım. Daha sonra for döngüsünde de itr isimli bir pointer oluşturup onun da head'i göstermesini ve i değeri index'e eşit olmadığı müddetçe for döngüsüne girmesini sağladım. For döngüsü içerisinde de bu presentNODE değerinin next'ini presentNODE'a atarayak düğümler üzerinde bir ilerlettim. İ değeri index'e eşit olduğunda da for döngüsünden çıktı ve presentNODE'u yani girilen indexteki düğümün adresini döndürdü.

List() isimli yapıcı fonksiyonda da head'i NULL'a eşitleyip size'ı 0 yaptım çünkü liste ilk oluştuğunda yapıcı fonksiyon oluşacak ve ortada henüz bir düğüm olmamış olacak.

İsEmpty() isimli fonksiyonda da size değeri 0'a eşitse 1,değilse 0 döndürmesini sağladım.

Add isimli index ve item olmak üzere iki parametre alan fonksiyonda girilen veriyi eklemeyi amaçladım. İlk olarak listeye ilk kez veri ekleniyorsa ilk if bloğuna girecek çünkü head null'a eşit olduğunda liste daha yeni oluşmuş oluyor. Bu if bloğunun içerisinde head pointerına yeni oluşacak düğümün adresi atanıyor ve size bir arttırılıyor.

Eğer ilk kez eklenmiyor da zaten halihazırda bir düğüm varsa da else bloğuna girer. Bu blokta da ilk olarak sonuncu isimli pointer'a FindLastNode ile bulduğumuz son düğümün adresi atanır. Eğer sona düğüm eklenecekse, bulunmayan bir index eklenecekse veya – değerli bir index giriliyorsa yine en sona bir düğüm oluşturulacak. Bu if bloğuna girecek ve sonuncu düğümün adresi sonuncuBir isimli pointera aktarılacak. Daha sonra bu son düğümün gösterdiği adrese de yeni bir düğüm eklenip onun adresi aktarılacak. Yeni eklenen düğümün gösterdiği adres son düğümün gösterdiği adres olması gerektiğinden parametreler içinde sonuncuBir->next yazacak ve previnin de son düğümü göstermesi gerektiğinden sonuncuBir yazacak.

Eğer daha önceden bir düğüm varsa ve sona değil de araya veya en başa eklenecekse else bloğuna girer ve sonuncuİki isimli pointera yine son düğümün adresi atanır. Yine yeni bir düğüm oluşturulur ve bu düğümün adresi sonuncuİki isimli pointerın next'ine atanır. Yeni oluşan düğümün de bu pointerı göstermesi sağlandıktan sonra size bir arttırılır. İnteger tipinde size'ın iki eksiği bir i değişkeni tanımlanır. Sona düğüm eklenmeden önceki sonuncu düğümü gösteren sonuncuİki pointerının tuttuğu data bilgisi bir sonraki düğüme kopyalanır. For döngüsüne girdiğimizde itr isimli bir pointer oluşturup ona da sonuncuİki nin adresini tanımlanırız. İ değeri index'e eşit olmadığı müddetçe döngüye girer ve gösterilen düğümün

öncesindeki datayı gösterilen düğümğn datasına aktarır. Daha sonraki satırda da gösterilen düğümü geri geri götürmek için (sola kaydırmak için) previnin adresini kendisine atarız. İ değeri indexi gösterdiğinde döngüden çıkar ve gösterilen düğüme parametre ile verilen data bilgisi girilir.

Remove isimli fonksiyonda girilen veri dosyasındaki indexi alıp o indexi silmeyi amaçladım. Eğer verilen index yoksa veya index eksili bir değerse yine listenin sonundan düğüm silmek üzere if bloğuna girecek. Burada da eğer ilk düğümün next'i nullsa yani listede tek bir düğüm varsa bu if bloğuna girecek ve yeni oluşturulan silinecek isimli pointer'a headin değeri atanıp, head'e null değeri atanacak ve silinecek pointerı silinip size da bir azaltılacak. Eğer geriye kalan son düğüm değilse silinecek olan, else bloğuna girecek ve sondanSilDugum isimli pointera listedeki son düğümün adresi atanacak. Daha sonra da bu düğümün nexti, bu düğümden önceki düğümün nextine atanacak ve düğüm silinip size da bir azaltılacak.

Eğer girilen index varsa ve uygunsa (eksili vs değilse) else bloğuna girecek. Eğer listedeki geriye kalan son düğüm siliniyorsa if bloğuna girecek ve üstteki blokta olduğu gibi burda da head null'a eşitlenip, headin gösterdiği yeri gösteren pointer silinip size eksiltilecek. Eğer verilen index varsa ve silinecek olan düğüm geriye kalan son düğüm değilse else bloğuna girecek ve present isimli pointera listedeki indeksi girilen düğümün adresi atanacak. Daha sonra for döngüsü içerisinde itr isimli bir pointer oluşturulup ona da bu girilen indexteki adresi atanacak ve itr'nin next'i null olmadığı müddetçe for döngüsüne girecek. Bu döngü içerisinde bulduğumuz düğümün adresini tutan pointera sonraki düğümün adresini atayarak düğümde bir ilerliyoruz ve daha sonra da bu düğümdeki datayı önceki düğümün datasına kopyalıyoruz. For döngüsü içerisinde bu şekilde ileri ileri giderek her seferinde bulduğumuz datayı bir önceki düğümün datasına kopyalıyoruz. For döngüsünden çıktıktan sonra da üzerinde olduğumuz düğümün nextini önceki düğümün nextine atayıp üzerinde bulunduğumuz düğümün adresini iade ediyoruz ve size'ı bir azaltıyoruz.

Ekrana yazma fonksiyonunda da son düğümse ekrana o düğümün datasını yazıyor ama sembolü yazmıyor, son düğüm değilse de önce datayı sonra da sembolü yazdırıyoruz ekrana.

Clear fonksiyonu ile de liste boş olmayana kadar remove fonksiyonuna 0 indexini göndererek en baştan itibaren düğümleri teker teker siliyoruz. Ancak burada liste hayatta kalmaya devam ediyor.

List isimli yıkıcı fonksiyonda ise clear fonksiyonunu çağırıyoruz ve listenin hayatına son vermiş oluyoruz.

Test.cpp'de ise ilk olarak isimler isimli bir liste oluşturdum. Dosyadaki verileri aktarmak için girilenVeri isimli string tipinde bir değişken, ilk harfin hangisi olduğunu aktarmak için command isimli char tipinde bir değişken, girilen index değerini almak için int tipinde index isimli bir değişken, girilen veri içindeki isim değerini almak için yine string tipinde girilenİsim isimli bir değişken ve "#" sembolünün yerini tutan size_t tipli position isimli bir değişken tanımladım. İfstream kullanarak dosyaOku isimli bir nesne oluşturdum ve parametrelerine veri.txt dosyasımı ekledim. Bu dosyayı okurken ".\\doc\\veri.txt" yolunu kullandığı için kendi veri dosyanızı okurken doc klasöründeki veri.txtyi değiştirebilirsiniz.

Eğer dosya açılamadıysa if bloğuna girerek dosyanın açılamadığını bildirmesini sağladım.

Eğer dosya açıldıysa burası atlanacak ve while döngüsüne gelecek. Getline ile dosyaOkudaki değerler girilenVeri'ye aktarıldığı sürece içeri gireceğiz. Command değişkenine girilen verideki ilk harf değerini atıyorum. Eğer command değişkeni 'E' harfi ise ekleme yapılması gerekiyor. Bu yüzden if bloğuna girecek. Girilen verinin uzunluk değerini lengthOfVeri isimli int tipindeki değişkene atıyorum. Daha sonra '#' işaretinin yerini find metodu ile bulup daha önceden tanımladığım position isimli metoda aktarıyorum. Substr metodu ile istenen işlemi gösteren harf, parantez ve '#' sembolünden sonraki kısımdan başlayarak, '#' sembolüne kadarki kısmı almasını sağlıyorum. '#'sembolünün yerinden 2 yi çıkartmamız bize bunu sağlıyor. Böylece girilen index değerini tam olarak alabiliyoruz ve bu değeri temporary isimli string bir değişkene atıyorum çünkü substr int bir değere aktarılamıyor. Daha sonra bu temporary değişkeninin değerini stoi metodu ile integer tipine dönüştürüp index isimli int değişkene aktarıyorum. Daha sonra yine subst ile girilenVerideki '#' sembolünden sonraki indexten başlayıp parantezden bir önceki yere kadar olan kısmı alıyorum ve girilenIsim isimli string değişkene atıyorum. Verinin uzunluğundan '#' sembolünün konumunu ve 2 yi çıkardığımızda bu bize parantezden bir önceki yere kadar gitmemizi sağlıyor.Daha sonra isimler ile add fonksiyonunu çağırarak parametreler haline değerleri atıyorum.

Eğer ilk harf S ise, yani command(komut) S ise silme işlemi yapılmalı. Bunun için de ilk olarak ")" sembolünün konumunu buluyorum. Substr ile 2. İndexten başlayarak [çünkü mesela S(2) dendiğinde girilen index 2. İndexten itibaren yazılmaya başlanmış oluyor.], ")" sembolünün olduğu index'e kadarki kısmı alıyor ve temporary isimli string değişkene aktarıyorum. Daha sonra bu değişkendeki değeri stoi metodu ile int'e dönüştürüp index isimli int değişkene atıyorum ve isimler ile remove'u çağırarak bu index değerini o foksiyona gönderiyorum. En sonda da *isimler diyerek ekranda bunu görmeyi sağlıyor, isimler'i siliyor ve dosyaOku nesnesini kapatıyorum. Makefile'da da gerekli derleme ve çalıştırma kodlarını yazıyorum.