ADI: KUTAY

SOYADI:YAMAN

NUMARASI: b171210074

GRUBU: C GRUBU(1. OGRETIM)

VERI YAPILARI 3. ODEV RAPORU

Bu odevde bizden istenilen txt uzantılı iki dosyada verilen dosyaları iki farklı BST'ye yerlestirip bunlar üstünden bir oyun gerçekleştirmek.Oyunun mantığı ise her düğüm kendi altındaki düğümlerin sayisini tutuyor ve sonra bu sayiların toplamina gore kazanan ve kaybeden belirleniyor.Toplami az olan turu kazanir ve 5 tur kazanan galip gelir veya 20 turun sonunda en fazla oyunu alan kazanır seklinde bir oyun gerçekleştirmemiz isteniyor.

Odevi yapardak öğrendiklerim recursive fonksiyonlari artik daha iyi kavradim çünkü ödevi yaparken BST classinin içine normalde BST'ye yazilan methodlardan farkli olarak 2 tane recursive method yazdim ve bunu yazarken recursive mantığını biraz daha iyi kavradım.

Odevi yaparken recursive fonksiyonları ilk yazdığımda direkt değerleri geri döndürmüştüm ancak döndürmeden önce hangi düğümün üstündeysem o dönecek olan değeri üstünde olduğum düğümün soy sayisina atmam gerekiyordu orda biraz zorlandım ancak o problemi şöyle hallettim;

Ilk önce dönen değeri soy sayisina attim ve ardindan o düğümün soy sayisini geri döndürdüm eğer böyle yapmayıp iki kere recursive fonksiyonu çağırsaydım program büyük dosyalarda baya zorlanabilirdi.

ISLEMLER CLASSI

Islemler classinda rakipBST ve benimBST isminde iki tane BST olusturdum çünkü dosyadan okunan sayilari bu arama ağaçlarında saklayacaktım.Olusturduktan sonra dosyadaki sayilari bu ağaçlara okudum ve dosyalari kapattim.Bide iki ağacında soy sayilarini guncel olarak hesaplayan soySayilariniGuncelle Methodu yazdim ama yazmasaydımda olurdu aslında direkt olarak rakipBST ve benimBST üzerinden soySayisiHesapla'yi uygulama içinden çağırabiliyorum zaten.

BST CLASSI

BST classi aslinda bizim derste yazdigmiz ile hemen hemen ayni ancak fazladan soySayisiHesapla,toplamSoySayisiBul,rootSil,enBuyuguSil,rootDegerGetir methodlari var.

SoySayisiHesapla methodu parametre olarak verilen agacın soy sayisini hesapliyor.Mesela rootu verirsek parametre olarak rootunkini hesaplarken ayni zamanda üstünden geçtiği düğümlerin yani tüm düğümlerinde soy sayisini hesaplayıp o dugumun soySayisi ozelligine degeri atiyor ve o degeri geri donduruyorki onun ustundeki dugumunde soySayisini hesaplayabilelim.

toplamSoySayisiBul methodu ise parametre olarak verilen agacin toplam soy sayisini hesapliyor.Mesela koku verdik diyelim ki odevde oyle yapicaz o agacın butun dugumlerındeki soySayisi degiskenlerini toplayip geri donduruyor.

RootSil methodu ise berabere kalma durumunda rootu silmemiz gerekiyordu o yuzden yazdim.

RootDegerGetir methoduda berabere kalma durumunda kok dugumlerin degerlerini birbirine vermesi icin.

EnBuyuguSil methodu ise kazanan taraf karsi tarafin agacindan en buyuk degere sahip olan dugumu kendine eklemesi icin en buyugu ağactan silip degerini geri donduruyor.

OYUN CLASSI

Oyun classinda ise temel oyun kurallarini uyguladim tamamen.Mesela skorlari tutan degiskenler olusturdum ve Islemler classindan nesne olusturacagim icin dosya yollarini parametre olarak kurucuda aldim.oyunuBaslat methodunda Islemler classindan nesne olusturulup dosya okuma islemleri ve soy sayilarini hesaplama islemleri gerceklestiriliyor ve boylelikle oyun ilk tur icin hazir hale geliyor.

SonDurumuEkranaBas methodu ise ilk once soySayilariniGuncelliyor ve sonra dugumleri postorder olarak ekrana bastirdiktan sonra BST classi icinde bulunan toplamSoySayisiBul methodunu kullanarak toplam soy sayilarini hesaplayip ekrana cikartiyor.

OyunuOynat methodunda ise tur sayisi kucuktur 20 oldugu surece donen bir while var ve ayrıca while'ın sonlarinda rakipSkoru ile benimSkoru kontrol eden bir if var eger herhangi biri 5 olduysa while dongusunu bitiriyor yani oyun bitmis oluyor.