T.C.
SAKARYA ÜNİVERSİTESİ

BİLGİSAYAR VE BİLİŞİM BİLİMLERİ FAKÜLTESİ
BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

PROGRAMLAMA DİLLERİNİN PRENSİPLERİ ÖDEV RAPORU

2. ÖDEV

B201210387 - Aleyna Elif ÖZKAN

SAKARYA

Mayıs, 2021

Programlama Dillerinin Prensipleri Dersi

2. ÖDEV

ALEYNA ELİF ÖZKAN

B201210387

1 - C

Özet

Bu ödevde verilen 3 adet json dosyasını C dili ile okuyup parçaladıktan sonra, bu verileri kullanarak işlemler yapmamız istenmişti. json dosyalarını manuel olarak parçalamak hem hata riskini arttıracağından hem de çok zaman alacağından yardımcı kütüphaneler araştırmamız gerekti. Aynı zamanda nesne yönelimli olmayan bir dil olan C'yi nesne yönelimliymiş gibi kullanmamız istendi. Bunu kodu farklı dosyalara ayırarak ve struct veri yapısını kullanarak yapmaya çalıştık. Sanki bir sınıf yazıyormuş gibi olabilmesi için fonksiyonları da ona göre uyarlamamız gerekiyordu. Bunları yapıp başarıyla dosyadaki verileri elde ettikten sonra birkaç küçük hesaplama yapıp sonrasında ekrana output olarak yazdırdığımızda bu ödevin bizden istediği her şeyi yapmış oluyorduk.

Anahtar Kelimeler: sınıf, json, C , nesne

1. GELİŞTİRİLEN YAZILIM

1.1 Ödevde bizden istenenler

Ödevin, derste gösterildiği şekilde nesne yönelimli benzetimi yapılarak her yapının başlık ve kaynak dosyaları ayrı bir şekilde ve başlık dosyasında metot gövdesi olmayacak bir şekilde tasarlanması.

1.2 Geliştirilen algoritma

Öncelikle json dosyalarını okumak ve parçalayabilmek için yazılmış kütüphanelerden en uygun olanını deneyerek seçtim. Sonrasında dosyaları okuyarak oluşturduğum karakter göstericilerine atadım. Kullanabileceğim en uygun kütüphaneyi seçtikten sonra (jsmn.h dosyası) içinde yazılmış olan hazır kütüphanelerden işime yaracak olan fonksiyonları tespit ettim. Parçalama fonksiyonlarını kullanarak ayırıcı kelimeler ("_id"," Sembol" vs.) yardımıyla -dosyaların bu formatta olacağını varsayarak- verileri birbirinden ayırdım. Dosyaların ilk ve son satırında kullanabileceğimiz bir veri olmadığından dolayı onları işleme almadım. Böylece struct yapılarının içinde parçalanmış verileri saklamış oldum. Sonrasında bu verileri birbiri ile karşılaştırarak istenen işlemleri gerçekleştirdim. Önce emirler dosyasından elde ettiğim verileri taradım. Burada alım veya satım yapılmak istenen hisseleri hisseler adlı json dosyasından tespit edip verilerini elimde tuttum. Ayını şeyi porfoy.json dosyası için de yaptım. Kullanıcının portföyünde verilen satış emri varsa yeteli adette olup olmadığını kontrol ettim. Eğer olmayan bir hisseyi satmaya çalışıyorsa hata bastırdım. Aynı şekilde kullanıcı daha önce portföyde olmayan bir hisseyi aldıysa, portföyü tutmuş olduğum arrayi güncelleyerek alınan yeni hisseyi kurucu fonksiyon yardımıyla ekledim. Eğer kullanıcı portföyde daha önceden bulunan bir hisseyi aldıysa bu hissenin portföydeki bilgilerini güncelledim. Sonrasında ise satılan hisselerin kar ve zararlarını hesapladım. Güncel portföy bilgileri ile kar ve zararları da ekrana bastırdım. Böylece ödevi tamamlamış oldum.

1.3 Dosyaların ve fonksiyonların açıklanması

1.3.1 Emir.h dosyasının içeriği

Bu dosyanın içerisinde EMIR adlı struct, bu struct içinde dosyanın içinde ayırmamız gereken id, sembol, işlem ve adet bilgileri bulunur. Aynı zamanda yapıcı ve yıkıcı fonksiyonları içerir.

1.3.2 Hisse.h dosyasının içeriği

Bu dosyanın içerisinde HISSE adlı struct, bu struct içinde dosyanın içinde ayırmamız gereken id, sembol, ad ve fiyat bilgileri bulunur. Aynı zamanda yapıcı ve yıkıcı fonksiyonları içerir.

1.3.3 Portfoy.h dosyasının içeriği

Bu dosyanın içerisinde PORTFOY adlı struct, bu struct içinde dosyanın içinde ayırmamız gereken id, sembol, maliyet ve adet bilgileri bulunur. Aynı zamanda yapıcı ve yıkıcı fonksiyonları içerir.

1.3.4 jsmn.h dosyasının içeriği

Bu dosyanın içerisinde json dosyalarını parçalayabilmek için kullanılan fonksiyonlar bulunur. Hazır bir kütüphanedir.

1.3.5 Banka.h dosyasının içeriği

Bu dosyanın içerisinde Hisse, Portföy ve Emir türünden oluşturulmuş arrayler, bu dosyaların satır sayıları ve yazdığım fonksiyonların ön tanımlamaları bulunur.

1.3.6 Emir.c dosyasının içeriği

Bu dosyanın içerisinde EmirOlustur () adlı kurucu fonksiyon ve EmirYoket() adlı yıkıcı fonksiyon bulunur.

1.3.7 Hisse.c dosyasının içeriği

Bu dosyanın içerisinde HisseOlustur () adlı kurucu fonksiyon ve HisseYoket() adlı yıkıcı fonksiyon bulunur.

1.3.8 Portfoy.c dosyasının içeriği

Bu dosyanın içerisinde PortfoyOlustur () adlı kurucu fonksiyon ve PortfoyYoket() adlı yıkıcı fonksiyon bulunur.

1.3.6 Banka.c dosyasının içeriği

Bu dosya, asıl işlemlerin gerçekleştirildiği yerdir. Öncelikle bankaOlustur() fonksiyonu içinde obje oluşturuyormuş gibi bir tanımlama ile This adıyla Banka türünden bir değişken oluşturulur. Sonrasında Banka structının içindeki bilgiler doldurulmaya başlanır. Dosyalar okunur ve satır sayıları hesaplanır. Ardından dosyalar parçalanır ve This objesinin içine kaydedilir. Bütün işlemler bu değişken üzerinden yapılır.

hisseParcala() fonksiyonu helper değişkenler yardımıyla jsmn.h dosyasındaki fonksiyonları kullanarak verileri parçalar. Bir satır parçalandıktan sonra helper değişkenler HisseOlustur() adlı kurucu fonksiyona yollanarak struct arrayine kaydedilmiş olur. Sonrasında helper değerler sıfırlanır ve döngünün başına dönülür. Dosya bitene kadar bu işlemler tekrarlanır. Dosya sonuna gelindiğinde satır sayısı da kaydedilir.

portfoyParcala() ve emirParcala() fonksiyonları hisseOlustur() fonksiyonu ile aynı mantıkta çalışır. Sadece ayrılacak anahtar kelimeler farklıdır.

JSONOku() fonksiyonu .json dosyalarını karakter arrayine atayarak bu arrayi döndürür.

satirSayisi() fonksiyonu dosyaların satır sayısını hesaplayıp bunu döndürür. İlk ve son satırları kullanmadığımız için bunları hesaplamaz.

Strndup() fonksiyonu C++ ‘de bulunan ancak C’de bulunmayan bir string fonksiyonudur. İhtiyacım olduğundan dolayı elle dosyaya eklemek zorunda kaldım. İstenen uzunlukta karakter göstericisi döndürür.

emirleriUygula() fonksiyonu emirleri teker teker gerçekleştirir. Eğer satış emri verildiyse satisYap() fonksiyonunu çağırıp toplam kar ve zararı hesaplar. Eğer alım emri verildiyse alisYap() fonksiyonunu çağırır. Toplam kar ve zararı ve güncel portföyü ekrana bastırır.

satisYap() fonksiyonu gerekli kontrolleri gerçekleştirerek istenen emirleri gerçekleştirir. Satılmak istenen hissenin portöyde varlığını ve adedini kontrol eder. Eğer mümkün olan bir işlem ise bunu gerçekleştirir. Mümkün olmayan bir işlem ise hata mesajı bastırır. Satış yapıldıktan sonra bu hisse porfoyDuzenle() fonksiyonu çağırılarak portföyden silinir.

alisYap() fonksiyonu da gerekli kontrolleri sağlayarak emirleri gerçekleştirir. Almak istenen hisse portföyde bulunmuyorsa kurucu fonksiyon ile portföye eklenir. Eğer portföyde varsa, portföyün bilgileri güncellenir.

portfoyDuzenle() fonksiyonu satılan hisseyi siler.

2. ÇIKTILAR

Verilen örnek json dosyalarına göre çıktı gösterildiği şekildedir.

3. SONUÇ

Bu ödev sayesinde dosya hiyerarşisi, makefile yazımı, nesne yönelimli olmayan bir dili nesne yönelimliye benzeterek kullanma, manuel olarak kütüphane dahil etme, dosya okuma ve parçalama gibi birçok şeyi pekiştirmiş olduk. Birçok .c ve .h dosyalarını birbirleri ile uyumlu kullanabilecek şekilde yazmayı öğrendik. Özellikle obje yaratabilecek şekilde bir benzetim yapmak zordu. Ancak nesne yönelimli mantığını anlayabilmemiz için güzel bir proje yapmış olduk.

Referanslar

[1] jsmn kütüphanesi - https://github.com/zserge/jsmn

[2] kütüphane örnekleri - https://github.com/zserge/jsmn/blob/master/example/simple.c