CSE 341 HW1 RAPORU

Muhammetalp ERDEM 151044006

Main işleyişi:

Güncellenmiş ödevde yapılması gereken kontrollerin hepsini yapıyor.

İnput: There are 3 apples, 12 oranges in the kitchen. Also the count of strawberries is 5. TL 2.45 remaining money. 8 is the number of colas.

Output:

There are three apples, 12 oranges in the kitchen. Also the count of strawberries is five. TL 2.45 remaining money. Eight is the number of colas.

İlk iş olarak dosyadan string okudum. Dosyayı açma, okuma ve kapatma işlerini dersin slaytlarından bulup uyguladım.

İkinci olarak ana döngüyü oluşturdum, bu döngüde aldığım stringin karakterlerini analiz ettim. Döngüyü sonlandırma koşulu olarak ise $zero ile karşılaştırma (Null kontrolu) yaptım. Döngünün mantığı basit; her digit için tek tek beq ile kontrol yaptım(ASCII değeri ile), eğer eşitlik varsa o sayıyla ilgili 'found' branch'ine yönlendirdim.

Eğer alınan karakter bir rakam değil ise döngünün counter(stringe dolaşan işaretçi)'ı bir artırılır ve sayı olmayan bu değer ekrana yazdırılır.

found Brancleri:

0'dan 9'a kadar olan bu branchlerin mantıkları aynı sadece işledikleri değerler farklı. Branch ilk olarak **multidigit** prosedürü ile birkaç digit birlikte mi diye kontrol ettim.

Sonra digit '.' dan sonra cümle başında mı yer alıyor diye kontrol ettim. (**sentencestarter**)

Sentencestarter'dan farklı olarak arada boşluk olmadan direk solunda nokta var ise **dotLeft**'e yönlerdirilir ki bu da tek bir sonucu işaret eder -> floating point.

Ardından sağında nokta var mı (**dotRight**) kontrolü gelir ve bu farklı anlamlara gelebilir, branching kendi açıklamasında detaylandıracağım.

Digit stringin başında mı kontrolü ise direk counter'ın 0 a eşit olup olmadığı kontrol ederek bulunur.

Bu branchlerin hiç birine takılmadıysa o halde digit harfler ile ekrana yazdırılır.

Found Brancleri:

Eğer alınan digit cümlenin başında ise found branchleri tarafından bu branchlere yönlendirilir ve ekrana ilk harfi büyük olacak şekilde yazdırılır.

dotRight:

Bu branch'e sağında '.' olan digitler gelir. '.' 'dan sonra boşluk gelip gelmediği kontrol edilir eğer geliyorsa bu bir cümle sonudur(**dotRightandspace**) ve digit olağan şekilde text'e çevrilir, eğer boşluk gelmiyorsa bu bir floating point sayıdır (**dotRightandnumber**) ve integer olarak ekrana yazdırılır.

dotRightandspace:

Cümle sonu digitlerini ifade eder ve **found** branchlerinden daha yalın **Purefound** branchlerine ihtiyaç duyar. Bu tip digitler normal bir şekilde text'e çevrilir.

Purefound:

Basitçe ihtiyaç sonucu **found** branchlerinin kontrolleri azaltılmış halidir.

sentencestarter:

Cümleyi başlatan digit kontrolü iki aşamalıdır. Digitlerin kendinden önce boşluk olduğu için buraya gelir, branch'in içinde ise o boşluktan önce nokta olup olmadığına bakılır. Eğer varsa baş harfi büyük yazılmak üzere **Found** branchlerine yollanır, eğer yok ise **notsentencestarter** ile **sentencestarte**r'ın altındaki satıra kodu etkilemiycek şekilde yönlendirilir.

notsentencestarter:

sentencestarter'ın yardımcı branch'idir. Yanlışlıkla bu branch'e alınan digitleri geldikleri yere (**found**) gönderir ve bu işi digit'in değer kontrolünü yaparak sağlar.

multidigit:

Prosedür! Önce bir önceki sonra bir sonraki değerleri alır ve rakam olup olmadığını kontrol eder, bu kontroller sonucu herhangi bir yerde olumlu çıkarsa a**pplymultidigit** branch'ine gönderilir. Yoksa jr ile çağrıldığı yere döner.

applymultidigit:

Aldığı digiti rakam olarak yazar ve loop'a geri döner.

exit:

Kodun herhangi bir kısmından çıkış yapmak için kullandım.