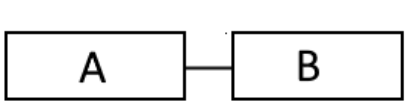
İNFİX 'TEN POSTFİX DÖNÜŞÜMÜN ÖNEMİ

Bilgisayar çarpmayı bile toplama üzerinden yapan aptal bir makinedir. Bilgisayar böylece işlemleri de lineer şekilde gerçekleştirir. Dolayısıyla '(' ve ')' parantezleri matematikte işlem önceliği belirtse de bilgisayara işlemleri kolay kılmak adına parantezleri kaldırıp işlem önceliği korunmalıdır. Bu eski tip hesap makinelerinde ve günümüzde de kullanılan INFIX -> POSTFIX dönüşümünü gerektirir.

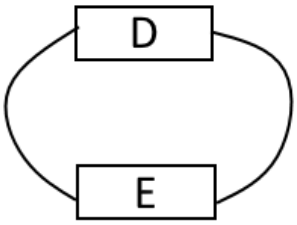
Örneğin, INFIX (Normal) ifadesi = x*y(5*z) +10 olan bir ifadenin POSTFIX (Düzenlenmiş) hali şudur ; xy*5z*/10+ . Böylece hem parantezler kaldırılıp hem de işlem önceliği korunmuş oluyor.

Biz Aritmetik işlemler yerine mantıksal ifadeler olan AND ve OR işlemi kullanacak olsak da işlem önceliği önemlidir (Kaynak : Rosen Descreate Mathematics)



Gösterim : A*B
Operand : A,B
Operator : *

A	B	AND	Bilgisayar Gösterimi
0	0	0	&& (AND)
0	1	0	
1	0	0	
1	1	1	



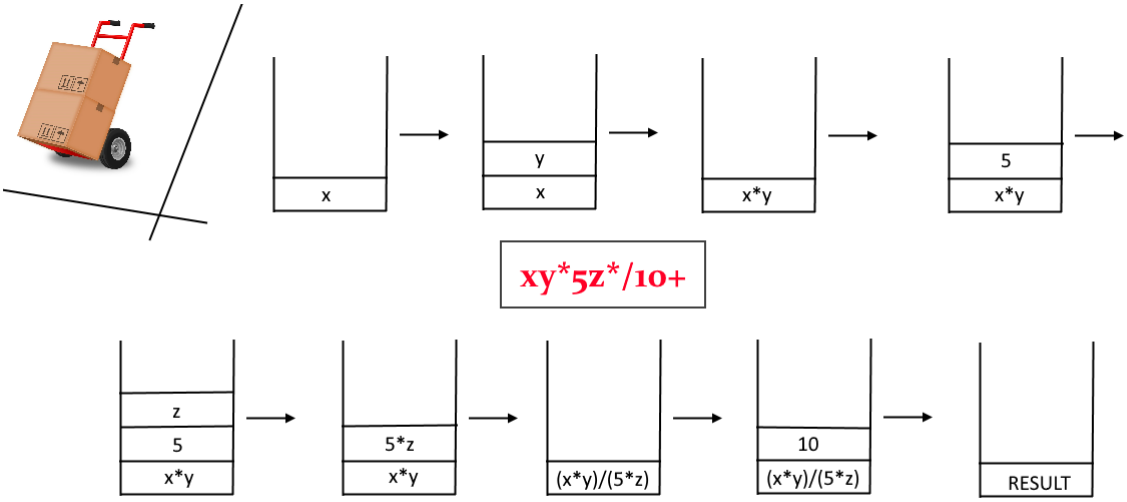
Gösterim : D+E
Opeand : D,E
Operator : +

D	E	OR	Bilgisayar G.
0	0	0	|| (OR)
0	1	1	
1	0	1	
1	1	1	

Bu hatırlatmalardan sonra stack(çıkın) yapısına göz atalım.

STACK VERİ YAPISI KULLANMANIN ÖNEMİ (STACK DATA STRUCTURE)

Stack bilgisayarda verileri saklarken kullandığımız yapıdır. Pazar arabasında son koyduğunuz poşeti ilk almanız gibi, STACK yapısında da LIFO(LAST IN FIRST OUT) yani son giren ilk çıkar felsefesi vardır. Böylece POSTFIX çevrilmiş ifadede operand gelirse stack'e ekliyoruz eğer operatör gelirse de son 2 stacki (son 2 poşeti) çıkarıp işlemi yapıp stack 'e (Pazar arabasına) geri koyuyoruz. En sonda eleman kalmadığında geriye kalan tek şey, sonuç olmuş oluyor.



YAZDIĞIM JAVA KODUNUN VE ALGORİTMANIN ÇALIŞMA AŞAMALARI

1-) Girişten denklemi al ve POSTFIX ifadeye dönüştür.

2-) Sadece operandları alan only_Operands değişkenine operandların atamasını yap. Tekrarlı olanları alma.

3-) Bu n operand için n! Permütasyondaki tüm varyasyonları oluştur. [permute()]

4-) Bu varyasyonların her biri için örneğin,

ABC varyasyonu için,

- ➔ **DENKLEMDE A 'YI SIFIR YAP.**
- ➔ **DENKLEM 0 İSE DİĞER VARYASYONA GEÇ. (ACB)**
- ➔ **DENKLEM 0 DEĞİLSE 0 B'Yİ SIFIR YAP.**
- ➔ **DENKLEM 0 İSE DİĞER VARYASYONA GEÇ. (ACB)**
- ➔ **DENKLEM 0 DEĞİLSE C'Yİ SIFIR YAP VE DENE**
- ➔ **….**

5-) Denklem 0 olmadığı durumlarda pointOfWiev değişkenini 1 arttırıyoruz ki sistem imzasında X 1:3 mü 2:3 mü kaçıncıda 0 olduğunu anlayalım.

6-) Denklem 0 ise diğer varyasyona geçiyoruz. Geçerken denklemi eski haline getiriyoruz (RESET), signature[] adını verdiğimiz diziye kaydediyoruz ve pointOfWiev i 0lıyoruz. Sıfırlamamızın nedeni yeni varyasyon için tekrar Xn:T! Hesabı yapacağımızdan geliyor.

7-) En son n! İ buluyoruz ve dizideki herbir elemana bölüyoruz. Bu arada dizi demek, birçok sayı,ifade, vb.. aynı anda erişebildiğimiz yapıya deniyor ve dizi dediğimiz şeyin mantığı da yukardaki stack mantığından ileri gelmektedir.

Not : Hocam Word dosyasını yeniledim ve akış diyagramlarını gönderdim. Çok uğraştım umarım beğenirsiniz. Bu arada akış şemaları hem pdf hem de .visio şeklinde. MİCROSOFT OFFICE VİSİO programı ile düzenleyebilirsiniz. Allah 'a emanet olun ☺