Yığın Veri Yapısı Örnek Uygulama 1: Geçen haftaki derste 10'luk tabandaki sayının 2'lik tabandaki karşılığının yazdırılması işlemi yığın veri yapısı kullanılarak gerçekleştirilmişti.

Yığın Veri Yapısı Örnek Uygulama 2: Parantez Eşleme Örneği

Aritmetik ifadede aç parantez ile kapat parantezin birbiri ile eşleşecek şekilde yazımını yığın veri yapısı ile kontrol edelim.

```
typedef struct Yigin
      {   int indis;
          char eleman[YBoyut]; // karakter değerleri üzerinden işlem yapılacak
      }Yiginlar;


int Yigina_Ekle(char ekle)


int Parantez_Kotrol(char *islem)
{ int i;
      for (i=0;i<strlen(islem);i++) // işlem metninin tüm karakterleri için
      {     if (islem[i]=='(') // aç parantez varsa
            Yigina_Ekle(islem[i]);  // yığına ekle
            else
            {     if (islem[i]==')')    // kapat parantez varsa
                        if(Yigin_Bosmu()==-1) // karşılığı yoktur HATA 1
                              return -1;
                        else
                              Yigindan_Cikar(); // yığından çıkar

            }
      }
      if (Yigin_Bosmu()!=-1) return 0;//metin bitti yığında eleman varsa HATA 2
      else
      return 1; // metin bitti yığın boşsa ifade DOĞRU
}

Yiginlar Yeni_Yigin;
void main()
{
      int i;
      char secim;
      char *islem;//="(3+4)/5)";
      clrscr();
      printf("Kontrol stringi=>");
      scanf("%s",islem);
      i=strlen(islem);
      i=Parantez_Kotrol(islem);
      switch(i)
      {
      case -1:printf("Hata: Açma Parantezi Eksik"); // HATA 1 =>  6*(5-3))
                  break;
      case 0:printf("Hata: Kapama parantezi Eksik"); // HATA 2 =>  6*(5-3
               break;
      case 1:printf("Parantez Hatası Yok"); // DOĞRU =>  6*(5-3)
      }
  }
```

Yığın Veri Yapısı Örnek Uygulama 3: Infix ifadenin Postfix ifadeye dönüştürülmesi:

Infix: a+b : operatör operandların arasında.
Postfix: ab+ : operator operantlardan sonra.
Prefix: +ab : operator operantlardan önce.

Infix notasyonun bir dezavantajı operatörlerin değerlendirme kontrolü için parantezler kullanılmasıdır. Postfix ve prefix notasyonlarda parantezler kullanılmaz. Infix notasyonu kullanan yüksek seviyeli dillerde ifadeler direkt değerlendirilemezler. En genel değerlendirme tekniği infix notasyonun postfix notasyona çevrilerek değerlendirilmesidir.

Infix notasyon postfix notasyona çevrilirken aritmetik ifade ilk değerden itibaren okunur. İfade de rakam varsa postfix karşılığı kısmına yazılır. İfade de operatör varsa [+, -, *, /, (,)] aşağıdaki kurala göre hareket edilir.

Operatörün infix ifadedeki öncelik değeri > yığının üstündeki operatörün öncelik değeri ise infix ifadedeki operatör yığına eklenir. Aksi durumda ise yığından operatör çıkarılır postfix karşılığa yazılır ve infix ifadedeki operatörde kalınmaya devam edilir. Yığın boş iken içindeki öncelik değeri 0 kabul edilir. ")" ile karşılaşılırsa "(" e kadar olan tüm operatörler yığından çıkarılır postfix karşılığa yazılır. "(" ve ")" postfix karşılığa yazılmaz. Infix ifade bittikten sonra yığında kalan elemanlar sırasıyla çıkarılarak postfix karşılığa yazılır.

Operatör	Infix ifadedeki öncelik değeri	Yığın içindeki öncelik değeri
)	-	-
^	4	3
*, /	2	2
+, -	1	1
(	4	0

Örnek: A+B*C-D/E infix ifadesini postfix ifadeye çevirelim (parantez kullanılmayan örnek).

Infix İfade	Yığın	Postfix Karşılık
A+B*C-D/E		
+ B*C-D/E		**A**
B*C-D/E	+	A
***C-D/E	+	A**B**
C-D/E	* +	AB
-D/E	* +	AB**C**
-D/E	+	ABC*
-D/E		ABC*+
D/E	-	ABC*+
/E	-	ABC*+**D**
E	/ -	ABC*+D
	/ -	ABC*+DE
	-	ABC*+DE/
		ABC*+DE/-

Örnek: A*(B+C)*D infix ifadesini postfix ifadeye çevirelim (parantez kullanılan örnek).

Infix İfade	Yığın	Postfix Karşılık
A*(B+C)*D		
***(B+C)*D		**A**
(B+C)*D	*****	A
B+C)*D	**(** *	A
+C)*D	(*	A**B**
C)*D	**+** (*	AB
)*D	+ (*	AB**C**
***D	(*	ABC**+**
***D	*****	ABC+
***D		ABC+***
D	*	ABC+*
	*	ABC+***D**
		ABC+*D*

Yığın Veri Yapısı Örnek Uygulama 4: Postfix ifadenin sonucunun hesaplanması

Postfix ifadenin ilk elemanından itibaren son elemana kadar geçerli eleman rakam (operand) ise yığına eklenir. Operatör [+, -, (,), *, /] ise sırasıyla yığından iki rakam çıkarılır. 2. çıkan rakam ile 1. çıkan rakama ilgili operatör işlemi yapılır ve işlem sonu yığına eklenir (toplamda yığından iki rakam çıkmış, yığına bir rakam eklenmiş olur). Postfix ifade bitince yığında kalan rakam değeri postfix ifadenin sonucudur.

Postfix İfade	Yığın
5 2 4 - *	
2 4 - *	5
4 - *	2 5
- *	4 2 5
*	-2 5
	-10

Postfix ifadenin sonucunun hesaplanması algoritması

```
Algoritma posfix_degerlendir (değer ifade <string>)
      ifade_boyutu=string boyutu
      Yiginlar Yeni_Yığın;
      i=0;
      Loop (index<ifade_Boyutu)
            if (ifade[i]==operand) Yigina_Ekle(ifade[i]);
            Else
                  operand2=Yigindan_Cikar();
                  operand1=Yigindan_Cikar();
                  operator=ifade[index];
                  deger=hesapla(operand1,operator,operand2);
                  Yigina_Ekle(deger);
            End if;
            i= i+1;
      end loop;
      return Yigindan_Cikar();
end Algoritma;
```