2017-2018 Bahar Yarıyılı
Veri yapıları ve Algoritmalar Dersi Final Ödevi

Konu : Aritmetik Hesaplayıcı

Problem: Bu ödevde, **yığın(stack)** veri yapısını kullanarak verilen önceki işlem sonuçlarını da kullanarak aritmetik işlemleri yapan bir sistem tasarlanacaktır.

Örnek : Tasarlayacağınız sistem aşağıdaki gibi verilen ardışık işlemleri, bir işlem sonunda elde ettiği sonuçları diğer işlemde kullanarak yapabilmelidir. Eşitliğin sol tarafında görülen her yeni değişkeni tespit etmeli ve bu değişkenin değerini kullanıldığı işlem sonucuna göre güncelleyebilmelidir.

a = 3 ;
b = 2 ;
c = a + b * 4 ;
b = c + b * 2 ;
d = a * (b – c) / 2 ;

Yukarıdaki işlemler sırası ile verildiğinde değişkenlerin değerleri sırası ile aşağıdaki gibi olmalıdır.
a ← 3, b←2, c ← 11 b← 15 d ← 6

Ödev 3 ana bölümden oluşmaktadır :

1. **Yığın işlemlerine ait fonksiyonların yazılması:**
   Yığına eleman **push** etme ve yığından eleman **pop** etme için gerekli fonksiyonları yazınız.

2. **Her satırdaki işlemin çözümü :**
   Verilen her işlem satırındaki işlemi aşağıdaki alt adımlarla yapınız.

   2.1 Infix – Postfix Dönüşümü :
   Infix formunda verilen aritmetik ifadeyi, **aritmetik operatörler için yığın kullanarak** postfix formuna çeviren fonkisyonu yazınız.

   İşlem Adımları :
   1. Giriş bilgisini string olarak okuyunuz.
   2. Okuma yaparken **değişkenler, sayılar ve işlem işaretleri arasında bir boşluk** bırakınız.
   3. Değişken isimlerini tek karakterlik alabilirsiniz.
   4. Giriş stringindeki sayı olan karakterlerin sayı karşılığını elde etmek için bir fonksiyon yazınız. Aritmetik işlemlerde ve işlem sonuçlarında **9'dan büyük sayılar olabilir**
   5. Kullanılan işaretleri şunlar olabilir : + - * / ( )
   6. Yanlış giriş yapılmadığı varsayılmaktadır.
   7. Aritmetik işlemi = işaretinin **sonrasından** başlayarak soldan sağa doğru değerlendiriniz.

      Eğer o anda bakılan bilgi:
      - **sol parantez ise :** sol parantez yığına **push** edilir.
      - **sağ parantez ise : sol parantez çıkana kadar** yığından **pop** işlemi yapılır. Alınan işlem işareti postfix ifadeye eklenir. Sol parantez görüldüğünde pop işlemine son verilir. Parantezler postfix'e eklenmez.
      - **sayı ise :** postfix ifadeye eklenir.
      - **işlem işareti ise :**
        i. yığının en üstünde sol parantez varsa veya en üstteki işaretin önceliği gelen işaretten düşük ise işlem işareti yığına **push** edilir.
        ii. Gelen işaretin önceliği en üstteki işaretin önceliğinden daha düşük ise yığındaki gelen işaretten yüksek öncelikli bütün işaretler için **pop** işlemi yapılır.
        iii. Stackten pop edilen işaretler postfix ifadeye eklenir.
        iv. İşlem işareti yığına **push** edilir.
      - **ifadeler bittiğinde :** yığındaki işaretler sıra ile pop edilerek postfix ifadeye eklenir.

   8. Postfix ifadeyi oluştururken **her ifade arasında bir boşluk** bırakınız.

Örnek:

Giriş : c = a * (b + 4) ;

İşlemler: a değişken olduğu için postfix'e eklenir:

Postfix : a Yığın : Boş

* yığına push edilir:

Postfix : a Yığın : *

(yığına push edilir:

Postfix : a Yığın : * (

b değişken olduğu için postfix'e eklenir:

Postfix : a b Yığın : * (

\+ yığının en üstünde (olduğu için yığına push edilir:

Postfix : a b Yığın : * (+

4 sayı olduğu için postfix'e eklenir:

Postfix : a b 4 Yığın : * (+

) geldiği için ('e kadar olan işaretler yığından pop edilip postfix'e eklenir. (işareti pop edilir ama postfix'e eklenmez.

Postfix : a b 4 + * Yığın : Boş

; işareti aritmetik ifadenin bittiğini gösterir. Bu durumda eğer yığında hala eleman var ise sıra ile pop edilip postfix'e eklenmelidir.

Ara Çıkış : a b 4 + *

2.2 Postfix İfadenin Çözülmesi :

İlk aşamanın sonunda elde edilen postfix ifadeyi **yığın veri yapısı kullanarak** çözen işlemi yapan fonkisyonu yazınız.

İşlem Adımları :

Postfix ifade soldan sağa doğru değerlendirilir.

Eğer o anda bakılan karakter:

- **sayı ise :** sayı yığına **push** edilir.
- **değişken ise** : değişkenin değeri yığına **push** edilir.
- **işlem işareti ise :** yığının üstündeki iki değer **pop** edilerek aralarında bu işlem yapılır. İşlem sonucu yığının en üstüne yerleştirilir.

Örnek:

Ara Çıkış : a b 4 + * ve a ←2 b ←3 için aritmetik ifade aşağıdaki gibi çözülür:

İşlemler: a değişken olduğu için sayı değeri yığına push edilir:

Yığın : 2

b değişken olduğu için sayı değeri yığına push edilir:

Yığın : 2 3

4 sayı olduğu için yığına push edilir:

Yığın : 2 3 4

\+ işlem işareti olduğu için yığının en üstündeki iki sayı pop edilir. Aralarında toplama işlemi yapılır. Toplam sonucu yığına push edilir.

Yığın : 2 7

* işlem işareti olduğu için yığının en üstündeki iki sayı pop edilir. Aralarında çarpma işlemi yapılır. İşlem sonucu yığına push edilir.

Yığın : 14

Postfix ifade bittiği için yığındaki sayı pop edilir. Bu sayı c değişkeninin yeni değeri olur.

Ekran Çıktıları :

Giriş olarak bütün işlemleri satır satır bir seferde alınız.

Her satırdaki ifade için ayrı ayrı :

- aritmetik ifadeyi,
- ifadenin postfix karşılığını,
- işlem sonucunda değişen değişkenin yeni değerini

ekrana yazdırınız.

Algoritmanızın programını C dilinde yazınız.

Teslim İşlemleri:
Ödevler **10 Haziran 2018 23.59'a** kadar github ortamına yerleştirilmelidir. GitHub ortamında yapılması gereken işlemler ekteki dokümanda ayrıntılı olarak yazılmıştır.
Ödev üzerinde yazacağınız bütün açıklamaları ve değişken isimlerini İngilizce yazınız.
Raporda bulunması gereken bilgileri ve teslim işlemleri ile ilgili detayları Arş. Grv. Zeynep Banu Özger'in sayfasından takip ediniz.

Değerlendirme:Ödeviniz aşağıdaki gibi değerlendirilecektir:

Algoritma Tasarımı ve Programın Çalışması: (%80)

1. Ödev, istenilen işlerin tamamını yerine getirmelidir.
2. Gereksiz kontrollerden ve işlemlerden arınmış bir tasarım yapılmalıdır.
3. Programda gerekli alt modüller belirlenerek her modül ayrı fonksiyon olarak yazılmalıdır.
4. Program hatasız çalışmalıdır.
5. Programın çalışması sırasında, konuyu bilmeyen kişilerin rahatlıkla anlayabilmesi için, giriş ve çıkışlarda mesajlarla bilgi verilmelidir.

Rapor Dokümantasyonu: (%20)

1. Raporun kapak sayfasında, dersin adı, öğrencinin ad, soyad ve numarası, ödev konusu bilgileri yer almalıdır.
2. Yöntem, uygulama ve sonuç bölümlerinde, yukarıda açıklaması verilen bilgiler anlatılmalıdır.
3. Kaynak kodda değişken deklerasyonu yapılırken her değişken tek satırda tanımlanmalı, tanımın yanına değişkenin ne için kullanılacağı açıklama olarak yazılmalıdır.
4. Değişken ve fonksiyon(veya metod) isimleri anlamlı olmalıdır.
5. Her fonksiyonun (veya metodun) yaptığı iş, parametreleri ve dönüş değeri açıklanmalıdır.
6. Gerekli yerlerde açıklama satırları ile kodda yapılan işlemler açıklanmadır.
7. Gereksiz kod tekrarı olmamalıdır.
8. Kaynak kodun formatı düzgün olmalıdır.