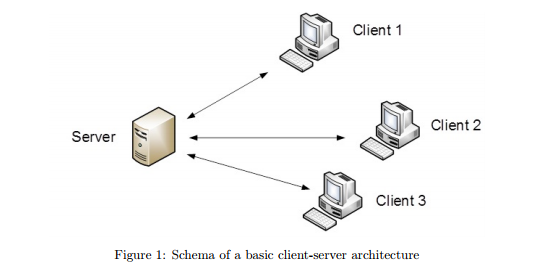
**Temel İstemci-Sunucu Simülasyonu:**



Bu görevde, her istemcinin belirli bir işlemler dizisinde çalıştığı veya bir kullanıcıdan istek (kesme) aldığı temel bir istemci-sunucu mimarisi tasarlayacaksınız (Şekil 1).

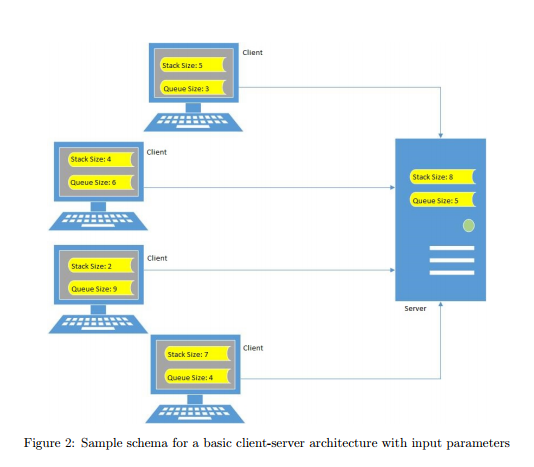
İstemci; istekleri ve süreçleri bir sunucuya gönderir ve benzer şekilde sunucu, istemciler tarafından gönderilen işlemleri ve kendi kesmelerini alır.

**Uygulama Ayrıntıları:**

Daha ayrıntılı olarak, her istemci bir süreç kümesini çalıştırmak için bir sıra yapısına sahip olmalı ve bir kullanıcıdan kesmeler kümesini işlemek için bir yığın yapısına sahip olmalıdır. Ardından müşteriler süreçlerini iletir ve sunucuya keser (daha özel olarak sunucu sırasına).

Benzer şekilde, sunucu da istemcilerin süreçlerini alıp çalıştırmak için bir sıra yapısına sahip olmalı ve kendi kesmelerini işlemek için bir yığın yapısına sahip olmalıdır.

Sonunda, program uygulama aşamasında aşağıdaki ayrıntıları takip etmelidir;



1. Hem istemciler hem de sunucu için, kuyruk yapısına kıyasla yığın yapısının daima daha fazla önceliğe sahip olması gerekir. Bu, istemci kuyruğunu işleme koymadan önce, istemci yığındaki kesmeler her zaman denetlenmeli ve istemci kuyruğundaki işlemlerden önce sunucuya gönderilmelidir. Benzer şekilde, sunucu kümesindeki kesmeler her zaman sunucu sırasındaki işlemlerden önce çalıştırılmalıdır.
2. İstemci yığınındaki en son kesme önce sunucuya gönderilmeli ve sunucu yığında en son kesme işlemi önce çalıştırılmalıdır.
3. Mimaride, her zaman sadece bir sunucu bulunur, ancak farklı giriş senaryolarında müşteri sayısı değişir. Benzer şekilde, her iki istemci için (her biri için) ve sunucu, yığın boyutu ve kuyruk boyutu, giriş durumuna bağlı olarak da değişebilir. Ancak yine de, istemcilerin ve sunucunun, sabit olan yalnızca bir sıra ve bir yığın yapısı olacaktır. Dolayısıyla, yukarıdaki değişken faktörler için programınızda dinamik bellek ayırma kullanmalısınız. Örneğin, Şekil 2, dört istemci ve bir sunucu ve her bir istemcinin ve sunucunun yığınının ve sıra boyutunun (5,3), (4,6), (2,9), (7, 4) ve (8,5) sırasıyla.
4. Her istemci ve sunucuda yığın ve kuyruk yapıları için mümkün olan her farklı durumu kontrol etmeli ve bunlarla baş etmelisiniz. İstemcinin yığın doluysa, bir sonraki kesme komutunu yok sayılacaktır. İstemcinin yığını boşsa, geçerli işlemi istemcinin sırasına göndermeye devam etmelisiniz. Müşterinin sırası doluysa, bir sonraki işlem komutunu yoksayarsınız. Son olarak, müşterinin kuyruğu boşsa, bir gönderme komutu olduğunda bir hata karakteri (hata kuralları için madde 8'e bakınız) verecektir. Aynı kurallar sunucu için de geçerlidir.
5. Sunucu (İstemci boyutu +1) sayısı alırken, istemciler (1, Müşteri boyutu) aralığında sayı alabilir.
6. Süreçler ve kesmelerin adları İngilizce alfabedeki küçük harflerden oluşur ('a' ile 'z' arasında).
7. Program, dört temel komut çeşidini işleyecektir. Her komut türü üç yuvadan oluşur:

• Müşteriler için İşlem İsteği Komutu:

Biçim: 'A' 'Müşteri Numarası' 'İşlem Karakter Adı'

Bu komutta:

- İlk yuvadaki 'A' karakteri, belirli bir müşterinin kuyruk yapısına ek olacağı anlamına gelir.

- Bu komuttaki ikinci alan, hangi müşterinin sırasının bir işlemin ekleneceğini belirtir. Bu, (1, Müşteri boyutu) aralığında bir sayı alır.

- Bu komuttaki üçüncü alan, eklenecek işlemin adını belirtir.

• İstemciler / Sunucu için Kesme İsteği Komutu:

Biçim: 'İstemci / Sunucu Numaram' '' Kesme Karakter Adı '

Bu komutta:

- Birinci yuvadaki 'ben' karakteri, belirli bir istemciye veya sunucunun yığın yapısına ek olacağı anlamına gelir.

- Bu komut türünün ikinci yuvası, hangi istemcinin (veya sunucunun istasyonunun) bir kesicinin ekleneceğini belirtir. (1, Müşteri boyutu + 1) aralığında bir sayı alır.

- Bu komuttaki üçüncü yuva, eklenecek kesme adını belirtir.

• İstemciler İçin Komut Gönder:

Biçim: 'S' 'Müşteri Numarası' 'G'

Bu komutta:

- İlk yuvadaki 'S' karakteri, belirli bir istemciden sunucuya bir işlem veya kesme gönderileceği, daha belirgin olarak sunucunun sırasına ekleneceği anlamına gelir.

- Bu komut türünün ikinci yuvası, sunucudan bir istemci, bir işlem veya kesme gönderileceğini belirtir. Aralığı içinde bir sayı alır (1, Müşteri boyutu)

- Üçüncü alan her zaman 'G' karakterini alacaktır, bu da geçersiz bir yuva anlamına gelir ve yalnızca yoksayılabilir.

• Sunucu için Komutu Çalıştır:

Biçim: 'O' 'G' 'G'

Bu komutta:

- İlk yuvadaki 'O' karakteri, sunucunun geçerli işlemini başlatmaya veya kesmeye başlamasını sağlar.

- İkinci ve üçüncü yiv her zaman bir 'G' karakteri alacaktır.

1. Üç tür hata kutusu için üç farklı tür hata karakteri vermelisiniz:

• Belirli bir müşterinin sırası doluysa, hata karakteri "1" vermelisiniz. (İşlem ilavesi sırasında)

• Belirli bir müşterinin yığın doluysa, hata karakteri '2' vermelisiniz. (Kesme ilavesi sırasında)

• Belirli bir istemcinin yığını boş olsa ve sıra da boşsa, '3' hata karakteri vermelisiniz. (İstemciler / sunucu için bir komut gönderirken / çalıştırırken)

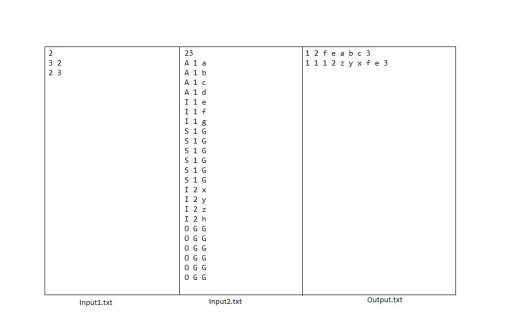
Yukarıdaki aynı kurallar, sunucunun yığını ve sıra yapısı için de geçerlidir. Özellikle, üçüncü kural için hem hata sırası hem de yığın yapıları hata karakteri '3' vermek için boş olmalıdır. Belirli bir istemci veya sunucu için, yalnızca yığın yapısı boş olduğunda ve istemciler / sunucu için gönderme / çalıştır komutu verildiğinde, yığın yapısı için bir hata karakteri vermemelisiniz. Sıraya geçmeli ve normal şekilde devam etmelisiniz.

**Giriş ve çıkış:**

Programınız, iki giriş dosyası için üç komut satırı bağımsız değişkeni (dosya yolu) ve çalıştırmaya başladığında bir çıktı dosyası almalıdır:

**Çalıştır Biçimi:**

Program 'Giriş 1 için Dosya Yolu' 'Giriş 2 için Dosya Yolu' 'Çıktı için Dosya Yolu' İki girdi dosyasındaki girdiyi okuyacak ve sonuçlarınızı bir çıktı dosyasına yazacaksınız (Şekil 3'e bakınız). Bir çıktı olarak, her bir istemci ve sunucu için bir günlük geçmişi oluşturmanız beklenir. Günlük geçmişi, her bir istemci / sunucunun gönderme / çalıştırma geçmişi ve hata kodlarının geçmişine sahip olacaktır.



Şekil 3: Örnek Giriş ve Çıktı Dosyaları: (Sol): Input1.txt dosyasındaki ilk satır, sistemin içerdiği kaç öğe (sunucu da dahil olmak üzere iki, bu durumda bir istemci ve bir sunucu) belirtir. İkinci satırda, belirli bir istemcinin sıra ve yığın boyutları sırasıyla belirtilir. Son satır (Üçüncü satır, bu örnekte) sırasıyla sunucu sırasını ve yığın boyutlarını belirtir. (Orta): Input2.txt dosyasındaki ilk satır, dosyadaki toplam komut sayısını belirtir. İkinci satırdan komutlar başlar (Sağ): Output.txt dosyasında, ilk satırdan başlayarak, her istemcinin günlük geçmişini yazmalısınız (bir satır, bir istemcinin geçmişinden oluşur). Son olarak, son satırda sunucunun günlük geçmişini de yazmanız gerekir. Özellikle, her satır için tarihin soldan başlayıp sağdan bittiğini unutmayın.

**Rapor:**

1. Deney sayfasından kopyala yapıştırmayın.
2. Sorundan anladığınız şeyi kısaca açıklayın.
3. Çözümünüz hakkında ayrıntılı bilgi verin.
4. İşlevlerinizin ve diğer kod parçalarının ayrıntılı bir açıklamasını sağlayın.