Teslim tarihi
23.11.23
23.59

Bil212 Veri Yapıları

Lab5

Bu labda basit bir HashMap veri yapısı implement edilecektir. Bu veri yapısı verilere hızlı erişim sağlamak amacıyla tasarlanmıştır. Saklanacak anahtar - değer çifti <Integer, String> olacaktır. HashMap veri yapısında ikililerin birbiriyle aynı hash değerine sahip olmasına çakışma(collision) denir. Bu labda çakışma önleme yöntemi (collison avoidance) olarak ayrı zincirleme (separate chaining) yöntemini kullanmanız gerekmektedir. Bu zamana kadar gördüğümüz veri yapılarını (max heap e kadar) hazır olarak kullanabilirsiniz.

Not: Başlangıçtaki bucket sayısı 11, yük faktörü 0.5 tir.

Public Metotlar:

put( ): anahtar değer çiftini tabloya yerleştirir. Tabloda daha önce bu anahtara ait bir değer varsa üstüne yeni değer yazılır.
get( ): anahtara ait değeri döner. Bu anahtara ait değer yoksa null döner.
remove( ): anahtar değer çiftini tablodan siler. Eğer daha önce eklenmemiş ise hiç bir değişiklik yapmaz.
capacity( ): toplam bucket sayısını döner.
size( ): toplam anahtar-değer sayısını döner.
toString( ): veri yapısının her bir bucket ini bir satır olarak ve her bucket taki elemanları da satırlarda gösterecek şekilde tasarlanmalıdır.

Private Metotlar:

hash( ): verilen Integer tipindeki key in basamak değerleri ile kaçıncı basamakta olduğu bilgisi çarpılır ve bu değerler toplanır.
Örneğin 2015 key i için hashCode metodunun döneceği değer 2x4 + 0x3 + 1x2 +5x1 olacaktır.

Not: Put metodunda doğru bucket ı bulurken toplam bucket sayısı ile modulo yapılmalıdır.

Teslim tarihi
23.11.23
23.59

resize( ): toplam anahtar-değer sayısı (yani size) nın kapasiteye oranı yük faktörünü aştığı ilk anda bucket sayısını %50 artırır ve anahtar değer çiftlerini yeniden yerleştirir.

Örneğin: 11 kapasite ile başlandığı durumda 6. çift ekleneceği zaman bucket sayısı 16 ya çıkarılmalıdır. Bundan sonraki artış 9. çift ekleneceği zaman, 16 dan 24 e olacaktır.

anahtar-değer ler veri yapısından silindiğinde bucket sayısı azaltılır. Elemanlar (anahtar-değer çiftleri) yeni yerlerine yerleştirilir. Bu durumdaki kural şöyledir: toplam anahtar-değer sayısı (yani size) nın kapasiteye oranı %25 in altına düştüğünde bucket sayısını %50 azalt.
Örneğin 12 elemanlı, 24 bucket lı bir hashmap ten 7 tane eleman sildiğimizde bucket sayısı 12 ye düşürülmeli.

Sürücü çıktısı:

```
HashMap after initial entries:
Bucket 0: (35, Three)
Bucket 1: (1, One) (28, yirmi sekiz)
Bucket 2:
Bucket 3:
Bucket 4:
Bucket 5:
Bucket 6: (22, Two)
Bucket 7:
Bucket 8:
Bucket 9:
Bucket 10:
```

```
HashMap after resize:
Bucket 0:
Bucket 1: (1, One)
Bucket 2: (58, One)
Bucket 3: (91, Three)
Bucket 4:
Bucket 5:
Bucket 6: (22, Two)
Bucket 7:
Bucket 8:
Bucket 9:
Bucket 10:
Bucket 11: (35, Three)
Bucket 12: (28, yirmi sekiz) (60, Two)
Bucket 13:
Bucket 14: (46, Four)
Bucket 15:
```

Teslim tarihi
23.11.23
23.59

```
Value for key 2: null

Value for key 91: Three
```

```
HashMap after removing key 60:
Bucket 0:
Bucket 1: (1, One)
Bucket 2:
Bucket 3:
Bucket 4:
Bucket 5: (102, bir sifir iki)
Bucket 6: (22, Two)
Bucket 7:
Bucket 8:
Bucket 9:
Bucket 10:
Bucket 11: (35, Three)
Bucket 12: (28, yirmi sekiz)
Bucket 13:
Bucket 14: (46, Four)
Bucket 15:
Bucket 16:
Bucket 17:
Bucket 18: (58, One)
Bucket 19: (91, Three)
Bucket 20:
Bucket 21:
Bucket 22:
Bucket 23:
```

Yüklenecekler:
HashMap.java adlı dosyayı size verilen Hash*Surucu.java ile çalıştıktan sonra* UZAK a yükleyebilirsiniz.