**Gebze Technical University**
**Computer Engineering**

**CSE 222 - 2018 Spring**

# HOMEWORK 5 REPORT

**Ramazan GUVENC**
**161044037**

Course Assistant:Fatmanur esirci

# 1 Double Hashing Map

This part about Question1 in HW5

## 1.1 Pseudocode and Explanation

Double hashing mapte ilk hash methodumu javanin hashCode() Object classindan sagladigi hash methodundan uretmeye karar verdim. tableSize a gore modunu aldiktan sonra 0 dan kucup olup olmadigini kontrol ettim ve ona gore ilk hash fonksiyonum hazirdi. 2. Hash fonksiyonumu ise Youtube daki OPENCOURSE MIT videolarindaki Hashing and Chaning videosundan aldim. 2. Hash methodum universal hashing fonksiyonlarindan. Ve son olarakta hashingin sürekli hizli calismasi için tableSize i kontrol edip 0.5, n/m > 0.5 oldugundan itibaren table i 2 kat buyuttum. Table size = 2^n seklinde gidiyor ve hash methodlarim tek sayilar üretiyor. Silinen değerlerin indexleri ayri bir listede saklaniyor ve böylece diğer methodlar bundan etkilenmiyor.

## 1.2 Test Cases

Hash table size i zorlayarak değiştirdim ve bu degisimi printMap methoduyla nullarida bastirirak gözlemledim. Dogal olarak hash table size i degisince zaten hash methodlarim ona bagli olduğu için bir sknti cikarmadi. Yorum satiri içinde mainde bu testler duruyor.

# 2 Recursive Hashing Set

This part about Question2 in HW5

## 2.1 Pseudocode and Explanation

Write pseudocode and explanation about code design. Indicate what you are using that interfaces, classes, structures, etc. LinkedList<E []> turunde table tutuyorum . Her collusion durumunda yeni bir E[] turunde array oluşturulup bu table a ekleniyor. LoadBalance n/m == 1 olunca devreye giriyor. Her E[] arrayinin tablesize i bir sonraki primeNumbera gore artiriliyor. Basta belirtmeyi unuttum ama printSet ve printMap null dahil tum table i basıyor. Silinen değerlerin indexleri ayri bir listede saklaniyor ve böylece diğer methodlar bundan etkilenmiyor.

## 2.2 Test Cases

Mainde denendi yorum satirinda duruyor. Bi hata bulunamadı.

# 3 Sorting Algortihms

## 3.1 MergeSort with DoubleLinkedList

This part about Question3 in HW5

### 3.1.1 Pseudocode and Explanation

```
If( list.size() < 2 ) return ;
                    Else{
                            Listeyi ikiye bol;
                            MergeSort(right);
                            MergeSort(left);
                            merge(left,right,actualList);
}
```

### 3.1.2 Average Run Time Analysis

Kendi merge kodum grafiktende gorulecegi uzere kitaptaki merge koduyla ayni oranda buyuyor. Tek problem benim merge kodum linkedlist üzerinde calistigi için constanti cok buyuk ve bu yüzden grafik cok dengesiz cikti fakat iki merge koduda O(nlogn) zamanda calisiyor.

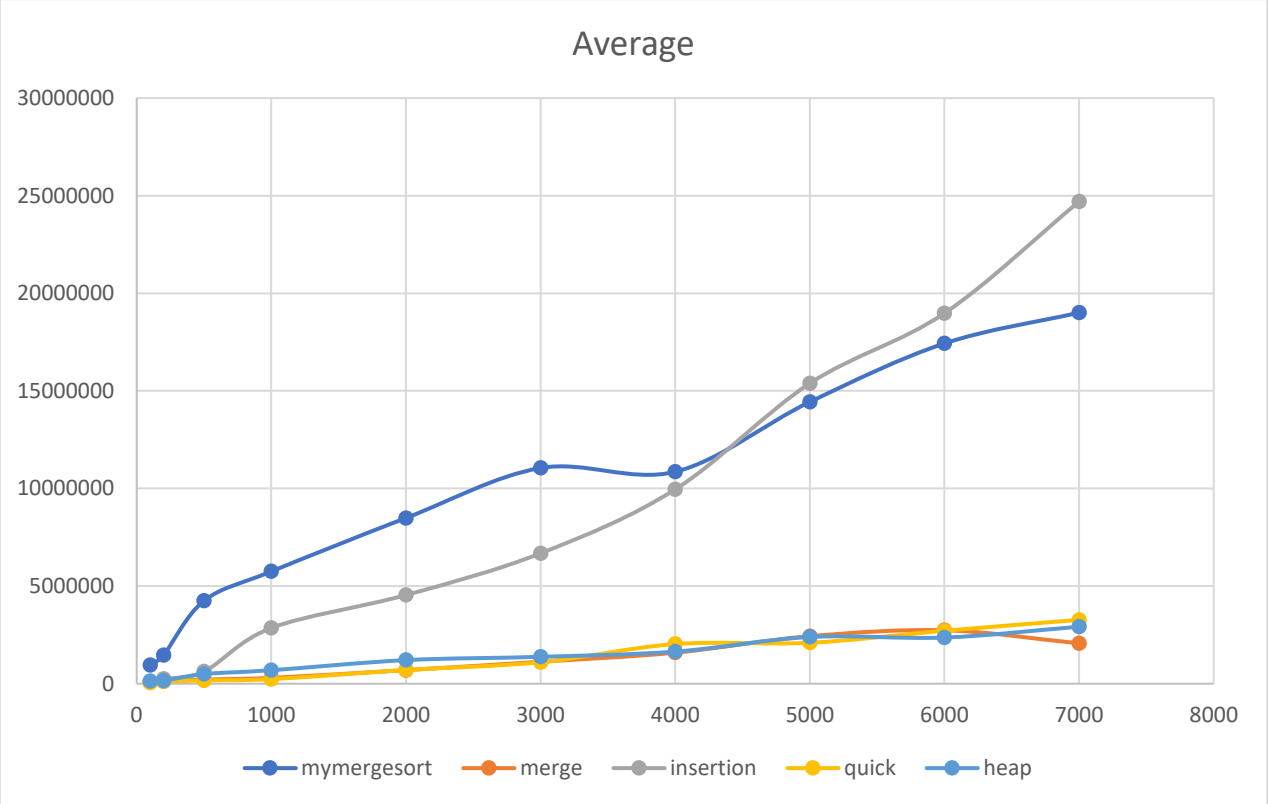
### 3.1.3 Wort-case Performance Analysis

Merge,Heap ve benim yazdigim merge sortun worst case senaryolari yine O(nlogn) cunku her seferinde ayni isi yapıyorlar.

QuickSort ise asagidaki grafiklerden de gorulecegi uzere worst case i O(n^2) ve bu sorted veya tersten sorted listelerde geçerli.

InsertionSort ise Tersten Sortlu bir array gelince O(n^2) oluyor.

# 4 Results




Yukaridada belirttiğim gibi aslinda benim mergesortumda O(nlogn) de calisiyor fakat constanti biraz fazla oldu. Hatta soyle bir değer verebilirim. Kitaptaki merge sort n = 7000 icin ortalama 2070572.7 nanosaniyede ve n = 100 icin ortalama 69615.9 saniyede calisti. Ve arada 29 kat var.

Benim yazdigim merge sort ise n= 7000  icin 19008434 nanosaniyede ve n = 100 icin 950591 nanosaniyede calisti. Aradaki kat ise 19.9kat.


Worst caseler için  sadece insesrtionort ve quick sortu yaptım cunku diğerleri her turlu ayni işlevi yaptiklari için worst case bir seyi farkettirmiyor cunkuoyle bir  caseleri yok. Bu iki sortun worst caselerinini ne  olduğu yukarda aciklandi.