HASHING

Hashing, elimizdeki veriyi kullanarak o veriden elden geldiği kadar benzersiz bir tamsayı elde etme işlemidir. Şifreleme gibi metotlarda yararlanıldığı gibi verinin indexlenmesi ve daha hızlı bulunması için de kullanılan bir yöntemdir. Elimizde 1-100 arası sırasız sayılar olduğu varsayalım. Sayıları son basamağına göre gruplandırırsak (indexlersek), tekrar bulmamız ve erişmemiz daha kolay olacaktır. İndexlemede yapılan temel işlem budur. Ancak aynı indexe sahip olan veriler hash işlemi için birer problemdir. Bu problemi çözmek için farklı probing yöntemleri çıkmıştır:

- Linear Probing
- Quadratic Probing
- Double Hashing

Aşağıdaki linkte 29 elemanlı bir hash tablosu içi simülasyon örnekleri verilmiştir:

https://www.cs.usfca.edu/~galles/visualization/ClosedHash.html

LINEAR PROBING

Lineer Probing index alanların çakışması durumunda bir sonraki boş alanı arar ve gelen elemanı buraya ekler. Örneğin mod 10 a göre indexleme yapıldığını varsayalım. 55 ve 65 sayıları 5 indisine yazılmalıdır. Ancak 55'i ekledikten sonra 5. İndis dolu olduğundan 6. İndis boş ise 65 oraya yazılır. Değilse sonraki boşluk aranır. Sona ulaşması durumunda 0. İndisten devam edilir. Ekleme, listeleme, arama ve silme işlemleri için gereken kodlar aşağıda verilmiştir.

```
void veriEkle(int anahtar) {
    int indis = hashFonksiyonu(anahtar);
    while(tablo[indis] != 0) {
        indis = (indis + 1)%10;
    }
    tablo[indis] = anahtar;
}

int ara (int anahtar) {
    int indis = hashFonksiyonu(anahtar);
    if(anahtar == tablo[indis]) {
        return indis;
    }else {
        while(tablo[indis] != 0) {
            indis = (indis + 1) % 10;
            if(anahtar == tablo[indis]) {
                return indis;
            }
        }
    }
    return -1;
}
```

```
void listele () {
    for (int i = 0; i < 10; i++) {
        System.out.print(tablo[i] + " ");
    }
}

void sil (int anahtar) {
    int indis = ara(anahtar);
    if(indis != -1 && tablo[indis] != 0) {
        tablo[indis] = -1;
    }
}
```

QUADRATIC PROBING

Lineer Probing index alanların çakışması durumunda i*i kadar sonraki boş alanı arar ve gelen elemanı buraya ekler. Örneğin mod 10 a göre indexleme yapıldığını varsayalım. 53 ve 43 sayıları 3 indisine yazılmalıdır. Ancak 53'i ekledikten sonra 3. İndis dolu olduğundan 1. iterayonda i*i yani (1*1 = 1) sonrasına bakar. 4. İndis de dolu ise 2. İterasyonda (2*2=4) sonrasına bakılır. Bu da 7. İndise denk gelmektedir. Boş yer arama işlemi bu şekilde devam eder. Sona ulaşması durumunda elde edilen indis değerinin tablo boyutuna göre modu alınır ve dairesel döngü oluşturulmuş olur. Ekleme, listeleme, arama ve silme işlemleri için gereken kodlar aşağıda verilmiştir.

```
void veriEkle(int anahtar) {
    int indis = hashFonksiyonu(anahtar);
    int yeniIndis = indis;
    int artis = 0;
    while(tablo[indis] != 0) {
        yeniIndis = (indis + artis*artis)%10;
        artis ++;
    }
    tablo[yeniIndis] = anahtar;
}

int ara (int anahtar) {
    int indis = hashFonksiyonu(anahtar);
    if(anahtar == tablo[indis]) {
        return indis;
    }else {
        int yeniIndis = indis;
        int artis = 1;
        while(tablo[indis] != 0) {
            yeniIndis = (indis + artis*artis) % 10;
            if(anahtar == tablo[yeniIndis]) {
                return yeniIndis;
            }
            artis ++;
        }
    }
    return -1;
}
```

```java
void listele () {
    for (int i = 0; i < 10; i++) {
        System.out.print(tablo[i] + " ");
    }
}

void sil (int anahtar) {
    int indis = ara(anahtar);
    if(indis != -1 && tablo[indis] != 0) {
        tablo[indis] = -1;
    }
}
```