Modul Praktikum DAA:
Hashing#2-Collision Handling

1. Linear Probing

Linear probing merupakan salah satu cara mengatasi collision pada hashing dengan open addressing. Dengan cara ini, jika terjadi collision, maka akan dicari index berikutnya yang masih kosong untuk digunakan. Fungsi hash yang digunakan adalah: $h(k,i) = (h'(k) + i)\ mod\ m$. Dengan $h'(k)$ adalah fungsi hash awal dan *i* adalah *probe number*.

Latihan: Membuat Collision Handling dengan Linear Probing

1. Gunakan kelas **Data** yang sudah dibuat pada praktikum minggu lalu.
2. Buatlah sebuah kelas **LinearProbeHash** untuk mengimplementasikan collision handling dengan linear probing.
3. Mirip dengan minggu lalu, tambahkan atribut M dan hashTable. Ada sebuah tambahan atribut DELETE_MARK yang merupakan konstanta dengan tipe **Data**. Objek ini digunakan sebagai penanda slot yang isinya sudah di-delete, untuk membedakan dengan slot yang memang masih kosong:

Pikirkan sejenak, apa bedanya slot yang memang masih kosong, dibandingkan dengan slot yang isinya di-delete ?

   ```
   //ukuran hash table
   int M;
   //untuk menandai apakah sebuah indeks sudah pernah dihapus isinya.
   final static Data DELETE_MARK = new Data(0, "MARK DELETE");

   //tabel hash
   Data hashTable[];
   ```

4. Buatlah constructor untuk kelas anda :

   ```
   public LinearProbeHash(int size){
       this.M=size;
       hashTable=new Data[this.M];
   }
   ```

5. Buatlah method hashFunction $h(k) = (h'(k) + i)\ mod\ M$.

   ```
   /*
     **Method ini digunakan untuk menghitung
     **nilai hash dengan h(key,i)=(key+1)%M
     **dengan i adalah probe number dan M ukuran tabel.
   */
   public int hashFunction(int key, int i){
       int hashValue;
       hashValue=(key+i)%this.M;
       return hashValue;
   }
   ```

6. Buatlah method untuk menambahkan pasangan key dan value ke dalam tabel hash.

```
/*
 **Method ini digunakan untuk menambahkan
 **sebuah key dan value-nya ke dalam tabel hash
 **kembalikan nilai true jika berhasil meng-insert
 **kembalikan false jika sebaliknya
*/
public boolean insert(int key, String value){
   int i=0;
   int hashValue;
   Data newdata=new Data(key, value);
   while(i<this.M){
      hashValue=this.hashFunction(key,i);
      if(this.hashTable[hashValue]==null||this.hashTable[hashValue].equals(DELETE_MARK)){
         this.hashTable[hashValue]=newdata;
         return true;
      }
      else i=i+1;
   }
  return false;
}
```

Sebelum anda melanjutkan ke step berikutnya, cobalah implementasikan method `public String delete(int key)` sendiri. Contoh solusi ada di halaman berikutnya.

7. Buatlah method untuk menghapus key dan value yang tersimpan di dalam tabel hash.

```
/*
  **Method ini digunakan untuk menghapus
  **key dan value yang pada tabel
  **berdasarkan key yang dimasukkan.
  **mengembalikan value jika key
  **ditemukan dan berhasil dihapus
*/
public String delete(int key){
   int i=0;
   int hashValue;
   String tempValue="";
   while(i<this.M){
      hashValue= this.hashFunction(key,i);
      if(this.hashTable[hashValue]==null) return null;
      else if(hashTable[hashValue].key==key){
         tempValue=hashTable[hashValue].value;
         hashTable[hashValue]=DELETE_MARK;
         return tempValue;
      }
      else i++;
   }
   return null;
}
```

8. Buatlah sebuah kelas **LinearProbeTester** untuk menguji operasi insert dan delete yang telah dibuat.

```
public class LinearProbeTester{
  public static void main(String[] args){
      LinearProbeHash MH=new LinearProbeHash(3);
      MH.insert(300, "Tono Sungkono Mardiono");
      MH.insert(450, "Rani Maryani Hardiyani");
      String deleted=MH.delete(301);
      System.out.println("Deleted: "+ deleted);
      String delete=MH.delete(300);
      System.out.println("Deleted: "+ delete);
  }
}
```

Latihan Mandiri :
Pada kelas **LinearProbeHash**, tambahkanlah method untuk operasi search!

2. Quadratic Probing

Quadratic probing merupakan salah satu teknik collision handling dengan open addressing. Fungsi hash yang digunakan adalah: $h(k,i) = (h'(k) + c_1 i + c_2 i^2) mod\ M$, dengan $h'(k)$ adalah fungsi hash awal, dan *i* adalah probe number, serta c_1 dan c_2 adalah dua buahkonstanta yang ditentukan dari awal.

3. Double Hashing

Quadratic probing merupakan salah satu teknik collision handling dengan open addressing. Fungsi hash yang digunakan adalah: $h(k,i) = (h_1(k) + i \times h_2(k)) \bmod m$, dengan $h_1(k)$ adalah fungsi hash awal dan $h_2(k)$ adalah sebuah fungsi hash lain yang berbeda dengan h_1.

Latihan Mandiri:

Implementasikanlah kelas **DoubleHash** dengan memodifikasi fungsi hash pada kelas LinearProbeHash diatas. Ditentukan bahwa ***$h_1(k) = (2 * k + 5) \bmod m$*** dan ***$h_2(k) = m - (k \bmod m)$***