MODUL 8

HASH MAP

Di Susun Oleh :

Muhammad Irfani

NIM : 20104037

Dosen

Faisal Dharma Adhinata, S.kom., M.Cs.

PROGRAM STUDI REKAYASA PERANGKAT LUNAK

FAKULTAS INFORMATIKA

INSTITUT TEKNOLOGI TELKOM PURWOKERTO

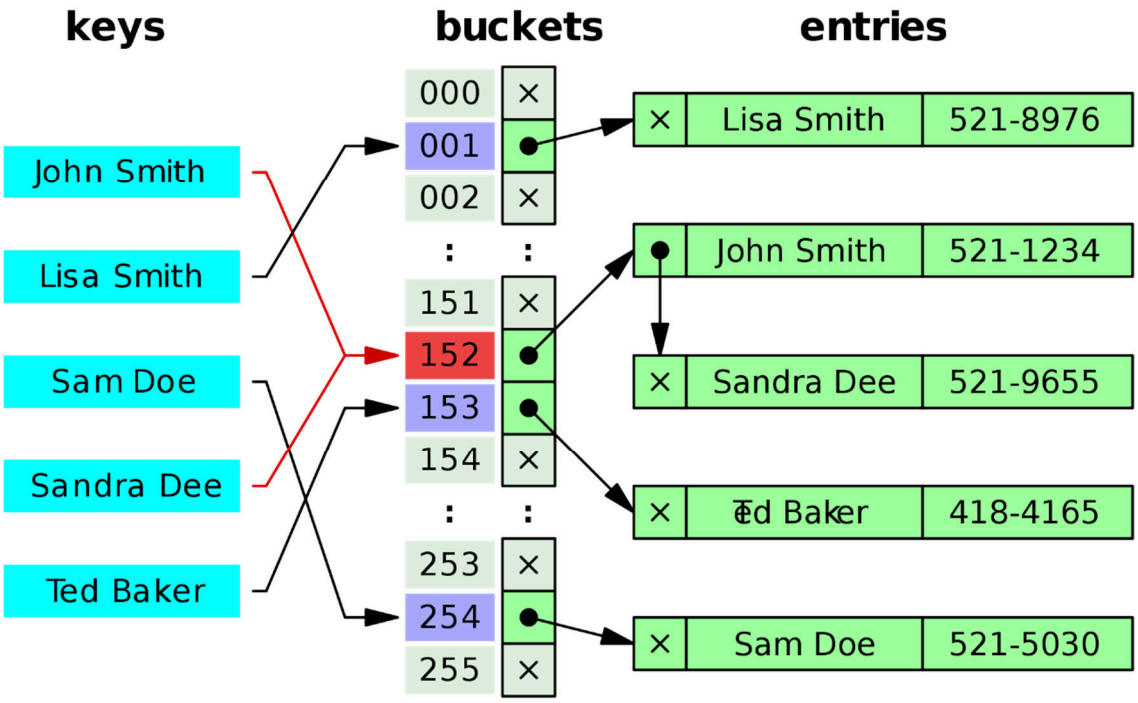
BAB I

Tujuan Praktikum

السَّلَامُ عَلَيْكُمْ وَرَحْمَةُ اللهِ وَبَرَكَاتُهُ

(Assalamu 'alaikum Wr. Wb)

Tujuan dari praktikum ini adalah agar para Mahasiswa dan pembaca mengerti agar bisa bermanfaat, dalam module ini kita akan belajar tentang hashMap, dimana hashMap di implementasikan seperti ini.



BAB II

Source Code

A. Method Entry

```
package com.praktikum;
public class Entry <K, V>{
   //   Deklarasi variable
   K key;
   V val;

   //   Getter & Setter
   public K getKey() {
      return key;
   }

   public void setKey(K key) {
      this.key = key;
   }

   public V getVal() {
      return val;
   }

   public void setVal(V val) {
      this.val = val;
   }

   //   Menentukan lokasi penyimpanan dari pasangan key dan value
   public int hashCode(){
      int prime = 13;
      int mul = 11;
      if(key != null){
         int hashCode = prime * mul + key.hashCode();
         return hashCode;
      }
      return 0;
   }
```

```
//    Method untuk membandingkan kesamaan nilai pada object
    public boolean equals(Object o){
//        Jika object yang dibandingkan bernilai sama
        if(this == o){
            return true;
        }
//        Jika object yang dibandingkan kosong atau tidak sama
        if(o == null || this.getClass().getName() != o.getClass().getName()){
            return false;
        }
        Entry e = (Entry)o;
//        Jika key yang dibadingkan sama
        if(this.key == e.key){
            return true;
        }
        return false;
    }
}
```

B. Deklarasi Attribut

```
private int capacity = 100; // menentukan total kapasitas pasangan key dan value
private int size = 0;
private Entry<K, V> table[] = new Entry[capacity]; // Deklarasi array untuk membuat tabel
```

1. Method hashing,size dan contain Keys

```
//    Membuat kode unik berdasarkan hasil dari method hashcode
   private int Hashing(int hashcode){
      int location = hashcode % capacity;
      return location;
   }

   //    Menentukan ukuran hashMap
   public int size(){
      return this.size;
   }

   //    Mengecek apakah key tersedia atau tidak
public boolean containsKey(Object key){
//       Jika key null
      if (key==null){
         if(table[0].getKey() == null){
            return true; // mengembalikan nilai true
         }
      }
//       Cari lokasi penyimpanan
      int location = Hashing(key.hashCode());
      Entry e = null; // entry null
//       Error handling
      try{
         e = table[location];
      } catch (NullPointerException ex){ }

//       Jika e != null dan kunci pada tabel == kunci yang kita cari
      if(e != null && e.getKey() == key){
         return true; // mengembalikan nilai true
      }
//       Default mengembalikan nilai false
      return false;
   }
```

2. Method Contain Value, get

```
public boolean containsValue(Object value){
      for (int i = 0; i < table.length; i++) {
//          Jika table ke-i tidak kosong dan value yang ada di tabel == value yang di cari
         if(table[i] != null && table[i].getVal() == value){
            return true; // mengembalikan nilai true
         }
      }
      return false; // mengembalikan nilai false
   }

   //   Mengambil data pada hashMap berdasarkan key
   public V get (K key){
      V ret = null;
//       Jika key == null
      Entry<K, V> e = null;
      if(key == null){
//          Error handling
         try {
//             Cek e di table ke-0
            e = table[0];
         } catch (NullPointerException ex){}
//          Jika e ditable[0] != null
         if(e != null) {
//             Mengembalikan nilai value yang ada di table ke-0
            return (V) e.getVal();
         }
      } else {
//             Cari lokasi peyimpanan key dan value
         int location = Hashing(key.hashCode());
         try {
//                cek e di table ke-lokasi
            e = table[location];
         } catch (NullPointerException ex) {}
//             Jika e != null & kunci pada tabel = kunci yang dicati
         if(e != null && e.getKey() == key){
//                Mengembalikan value pada kunci tersebut
            return (V) e.getVal();
         }
      }
//       Mengembalikan nilai default jika semua tidak terpenuhi
      return ret;
   }
```

3. Method Put

```
//   Menaruh data ke dalam hashmap
   public V put(K key, V val){
      V ret = null;
//       Jika key null
      if(key == null){
         ret = putForNullKey(val);
         return ret;
//          Jika key tidak null
      } else {
         int location = Hashing(key.hashCode());
//          Jika lokasi melebihi kapasitas
         if(location >= capacity){
            System.out.println("Rehashing required");
//             Mengembalikan nilai null
            return null;
         }
         Entry<K, V> e = null;
         try {
            e = table[location];
         } catch (NullPointerException ex) {}
//          jika e != null && key pada hashmap == key yang diinputkan
         if(e != null && e.getKey() == key){
//             ret = value yang ada pada key yang diinputkan
            ret = (V) e.getVal();
         } else {
//             Jika e == null && key pada hashmap != key yang diinputkan
            Entry<K, V> eNew = new Entry<K, V>();
//             Set key dan value
            eNew.setKey(key);
            eNew.setVal(val);
            table[location] = eNew;
            size++;
         }
      }
      return ret;
   }
```

4. Method putForNullKey

```
//    Jika akan menaruh value pada key yang bernilai null
    private V putForNullKey(V val) {
        Entry<K, V> e = null;
        try{
            e = table[0];
        } catch (NullPointerException ex){
        }
        V ret = null;
//        Jika e != null && key pada hashmap == null
        if(e != null && e.getKey() == null){
            ret = (V) e.getVal();
//            put value
            e.setVal(val);
            return ret;
        } else {
            Entry<K, V> eNew = new Entry<K, V>();
//            set key jadi null
            eNew.setKey(null);
//            put value
            eNew.setVal(val);
            table[0] = eNew;
            size++;
        }
        return ret;
    }
```

5. main

```
public static void integerMenu(){
   System.out.println("=== Integer Menu ===");
   System.out.println("1. Input data");
   System.out.println("2. Print data");
   System.out.println("3. Check Key");
   System.out.println("4. Check Value");
   System.out.println("5. Check Size");
   System.out.println("6. Back in main menu");
}
public static void mainMenu(){
   System.out.println("=== Menu ===");
   System.out.println("1. Data String Only");
   System.out.println("2. Data String with key");
}

public static void main(String[] args) {
   HashMapImpl<Integer, String> hashMap1 = new HashMapImpl<Integer, String>();
   HashMapImpl<String, String> hashMap = new HashMapImpl<String, String>();
   Scanner scan = new Scanner(System.in);
   boolean end = false;
   do {
      mainMenu();
      System.out.println("Pilih");
      System.out.print("> ");
      int usrInput = scan.nextInt();
      switch (usrInput){
         case 1 :
            hashMap.put("Nama", "Muhammad Irfani");
            hashMap.put("Prodi", "Rekayasa Perangkat Lunak");
            hashMap.put("Gender", "Laki-laki");

            System.out.println("Nama :\t"+hashMap.get("Nama"));
            System.out.println("Prodi :\t"+hashMap.get("Prodi"));
            System.out.println("Gendedr :\t"+hashMap.get("Gender"));

            System.out.println("Check Value >>>\t"+hashMap.containsValue("Muhammad Irfani"));
            System.out.println("Check Key >>>\t"+hashMap.containsKey("Prodi"));
            System.out.println("Check Size >>>\t"+hashMap.size());
            break;

// Muhammad Irfani

// 20104037

// SE04B
```

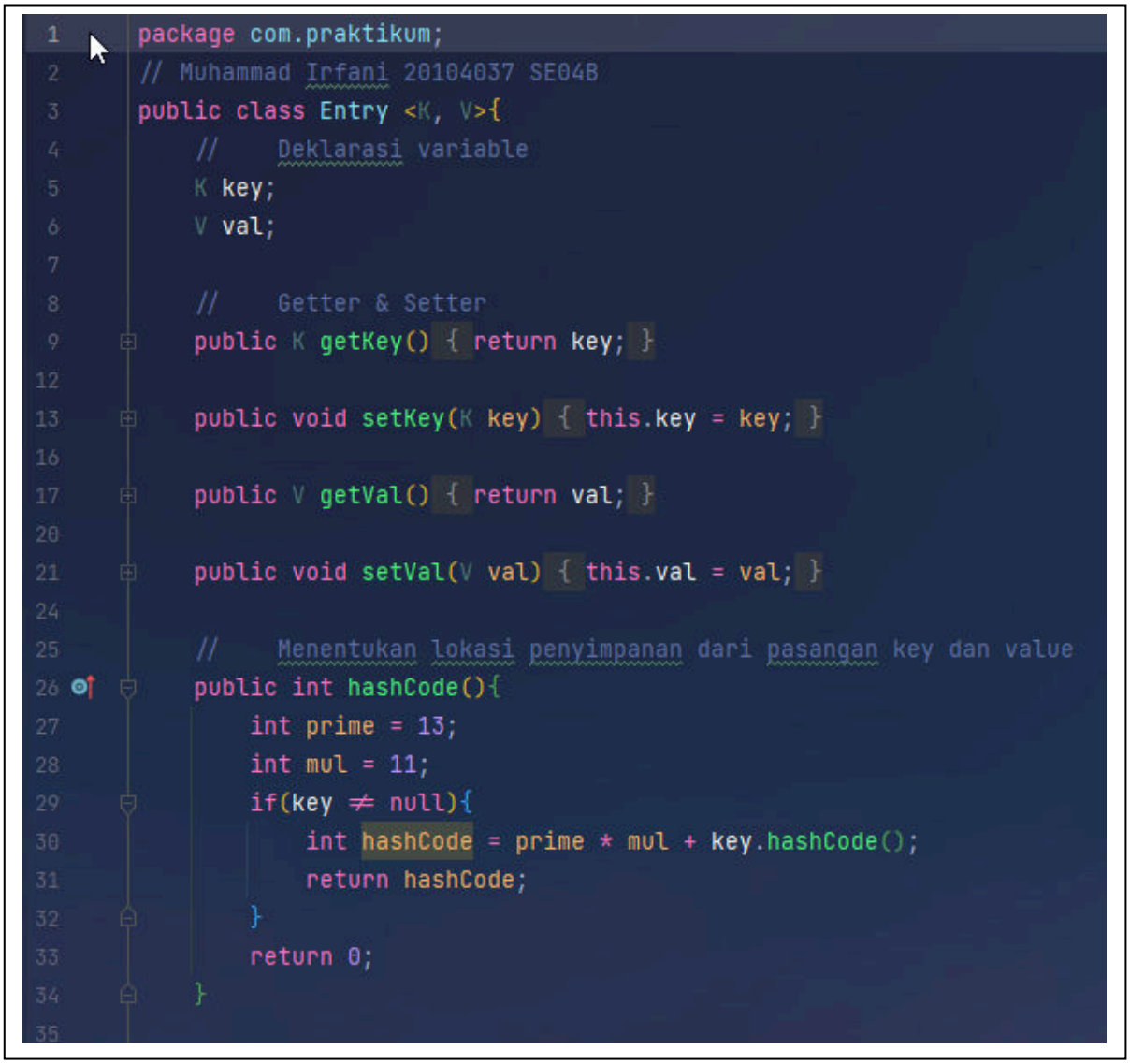
```
case 2 :
        boolean repeat = true;
        do {
           integerMenu();
           System.out.print(">\t");
           usrInput = scan.nextInt();
           if (usrInput == 1){
              System.out.print("Input Key : ");
              int integerInput = scan.nextInt();
              System.out.print("input String Value :");
              String stringInput = scan.next();
              hashMap1.put(integerInput, stringInput);
           }else if (usrInput == 2){
              System.out.print("Input Key >\t");
              int key = scan.nextInt();
              System.out.println(hashMap1.get(key));
           }else if (usrInput == 3){
              System.out.println("Input Contain search >\t");
              int containInteger = scan.nextInt();
              System.out.println(">>>"+hashMap1.containsKey(containInteger));
           }else if (usrInput == 4){
              System.out.print("Input >\t");
              String containString = scan.next();
              System.out.println(">>>" + hashMap1.containsValue(containString));
           }else if (usrInput == 5){
              System.out.println(">>>"+hashMap1.size());
           }else if (usrInput == 6){
              repeat = false;
           }
        }while (repeat == true);
        break;
   }
}while (end ==  false);
```

BAB III

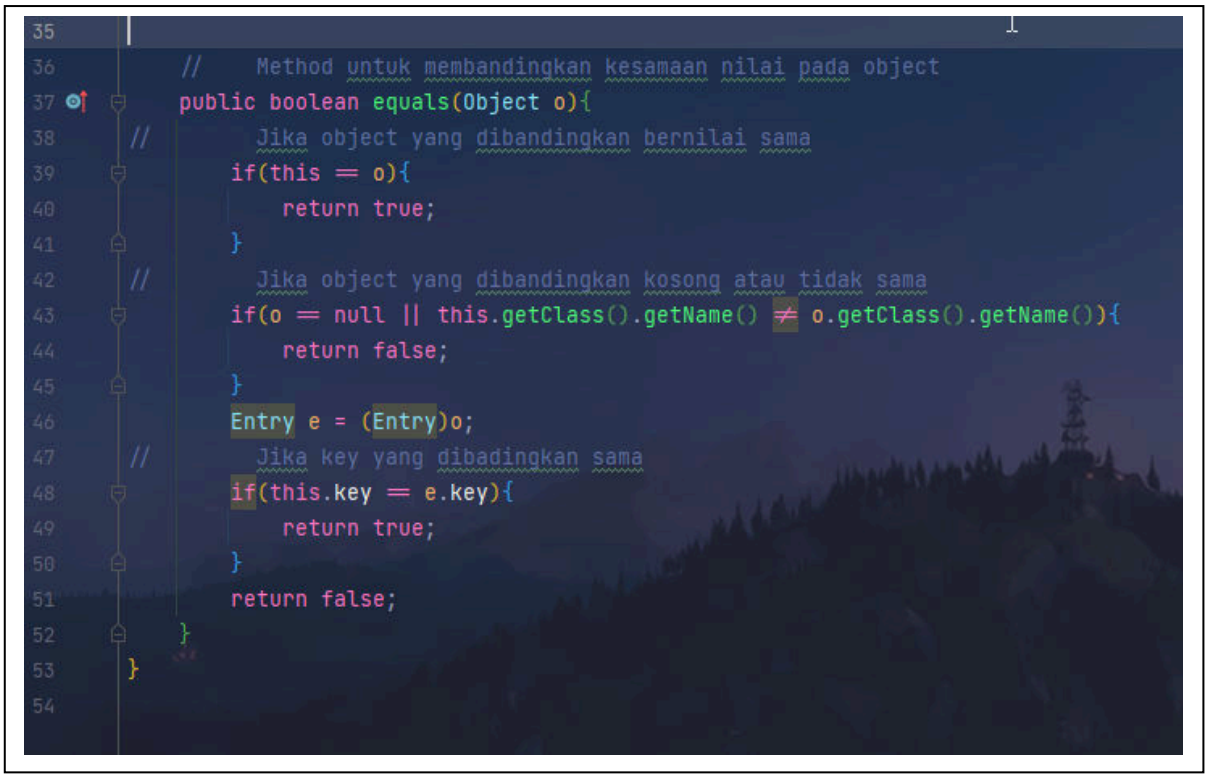
Analiasa Source Code Dan Tugas

A. Membuat method entry

Pertama kita akan Membuat method entry, dimana method ini berguna untuk method-method lainnya, buat class baru dengan nama Entry kemudian kita akan memasukkan kode seperti ini



```java
package com.praktikum;
// Muhammad Irfani 20104037 SE04B
public class Entry <K, V>{
    //    Deklarasi variable
    K key;
    V val;

    //    Getter & Setter
    public K getKey() { return key; }

    public void setKey(K key) { this.key = key; }

    public V getVal() { return val; }

    public void setVal(V val) { this.val = val; }

    //    Menentukan lokasi penyimpanan dari pasangan key dan value
    public int hashCode(){
        int prime = 13;
        int mul = 11;
        if(key != null){
            int hashCode = prime * mul + key.hashCode();
            return hashCode;
        }
        return 0;
    }

}
```



```
35
36          //    Method untuk membandingkan kesamaan nilai pada object
37          public boolean equals(Object o){
38          //      Jika object yang dibandingkan bernilai sama
39              if(this == o){
40                  return true;
41              }
42          //      Jika object yang dibandingkan kosong atau tidak sama
43              if(o == null || this.getClass().getName() != o.getClass().getName()){
44                  return false;
45              }
46              Entry e = (Entry)o;
47          //      Jika key yang dibadingkan sama
48              if(this.key == e.key){
49                  return true;
50              }
51              return false;
52          }
53      }
54
```

kemudian setelah Membuat entry class selesai kita akan melanjutkan Membuat sebuah class main dari program dimana class main program akan berisi beberapa method yang akan kita gunakan

B. Deklarasi Attribut

Deklarasi attribute akan kita gunakan untuk menentukan berapa banyak key atau data yang akan di tampung oleh system

```java
package com.praktikum;
//Muhammad Irfani 20104037 SE04B
import java.util.Scanner;

public class HashMapImpl<K, V> {
    //    Deklarasi atribut
    private int capacity = 100; // menentukan total kapasitas pasangan key dan value
    private int size = 0;
    private Entry<K, V> table[] = new Entry[capacity]; // Deklarasi array untuk membuat tabel
```

Bisa dilihat bahwa capacity adalah 100 digunakan untuk menentukan capacitynya