**MODUL 8**

**HASH MAP**



**Di Susun Oleh :**

**Muhammad Irfani**

**NIM : 20104037**

**PROGRAM STUDI REKAYASA PERANGKAT LUNAK**

**FAKULTAS INFORMATIKA**

**INSTITUT TEKNOLOGI TELKOM PURWOKERTO**

**2020**

**Dosen**

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

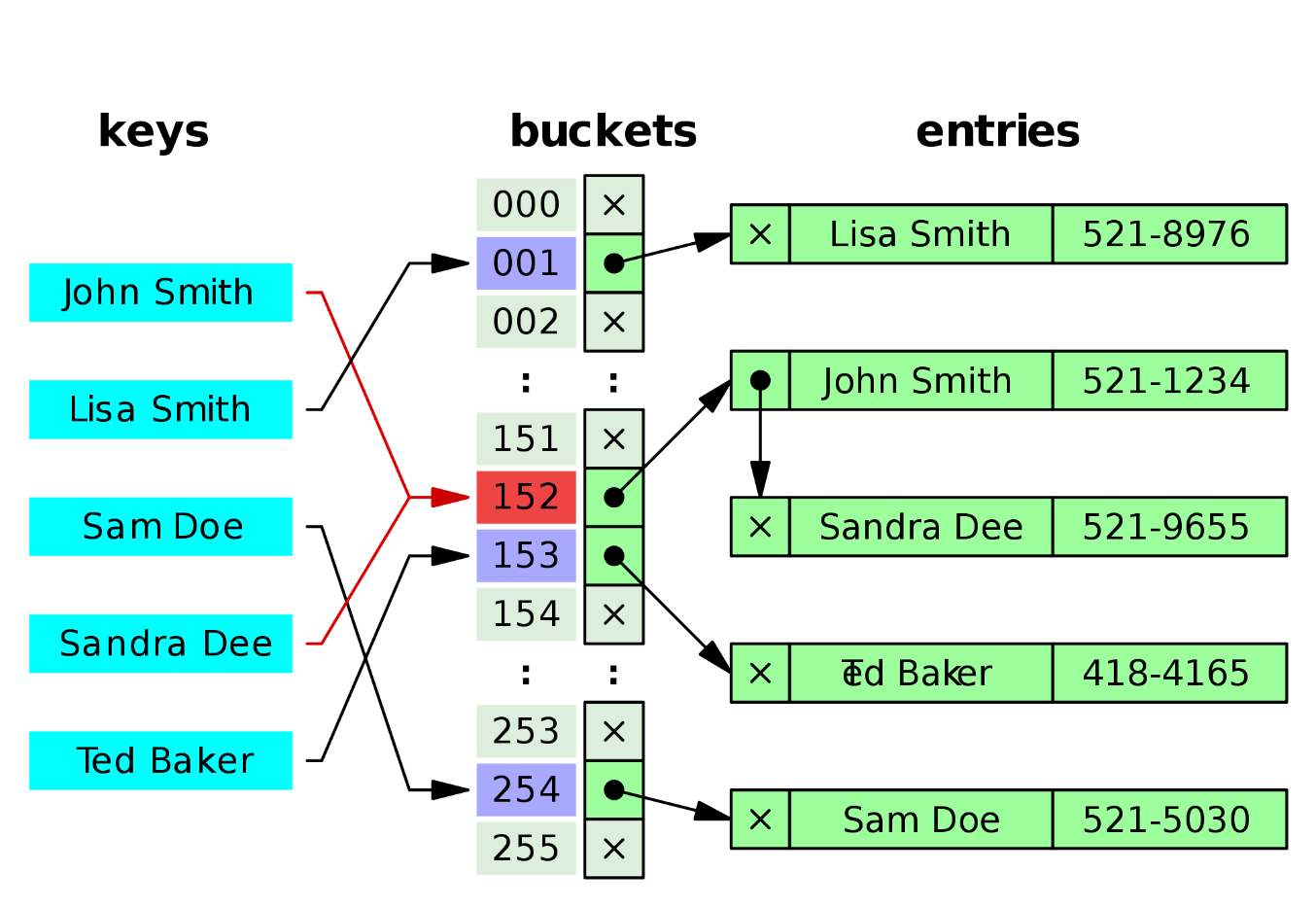
**BAB I**

**Tujuan Praktikum**

**السَّلاَمُ 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**

**2. Method Contain Value, get**

// 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;  
 }

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);

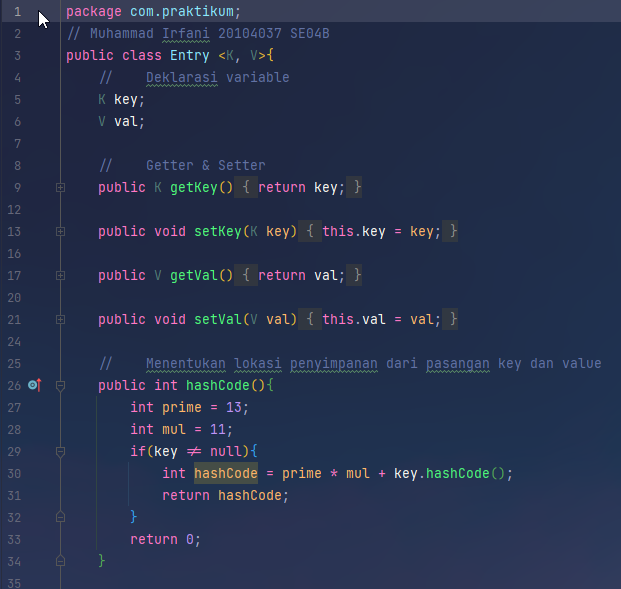
**BAB III**

**Analiasa Source Code Dan Tugas**

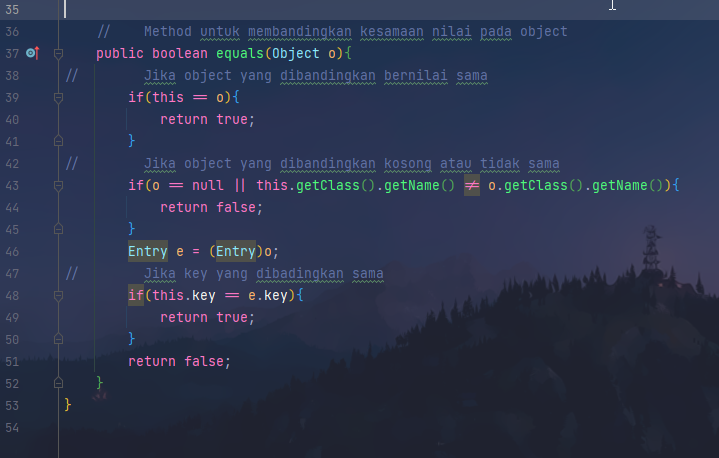
Untuk code programnya teman-teman atau pembaca bisa melihat di Tab source code pada main, atau pada link https://linktr.ee/20104037

**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

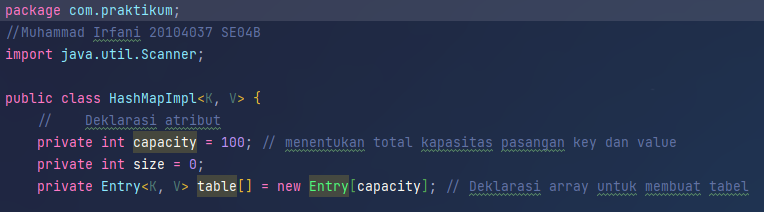


kemudian setelah Membuat entry class selesai kita akan melanjutkan Membuat sebuah class main dari program dimana class main program akan berisi beberapa method yang akan kita gunakan



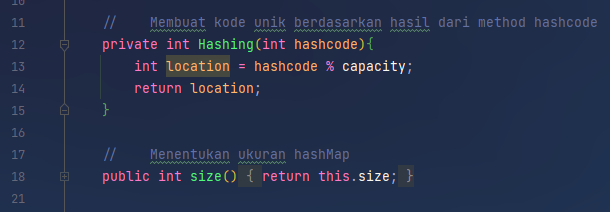
**B. Deklarasi Attribut**

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



Bisa dilihat bahwa capacity adalah 100 digunakan untuk menentukan capacitynya

**1. Method hashing dan Size()**



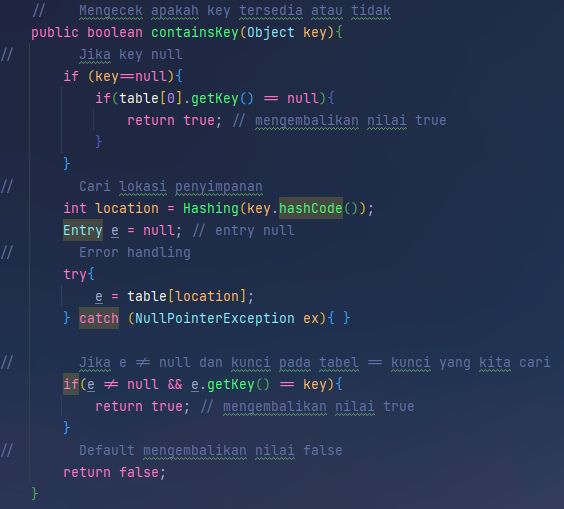
Method hashing digunakan untuk Membuat kode unik, dari namannya hashing sudah kita tau, kemudian ada method size, dimana method size ini digunakan untuk melakukan check size data contohnya,



Maka size nya akan manjadi 1



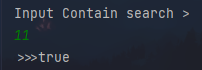
**2. Method Contain Key**



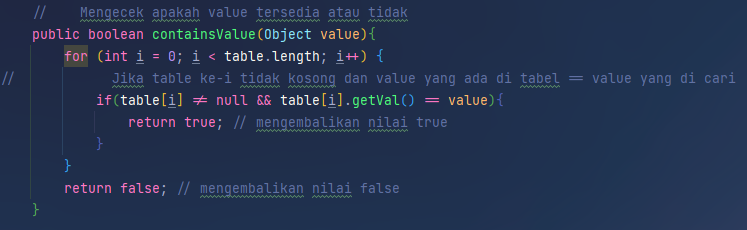
Digunakan untuk melakukan check apakah key yang user inputkan ada apa tidak

Contohnya

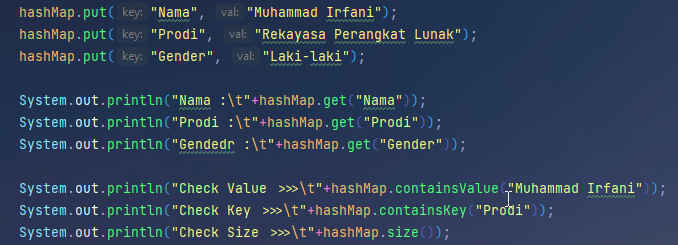
Saya sudah input data key 11 tadi ya, maka jika ada data 11 akan true



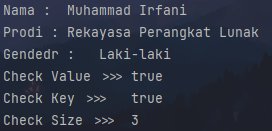
**3. Contain Value**



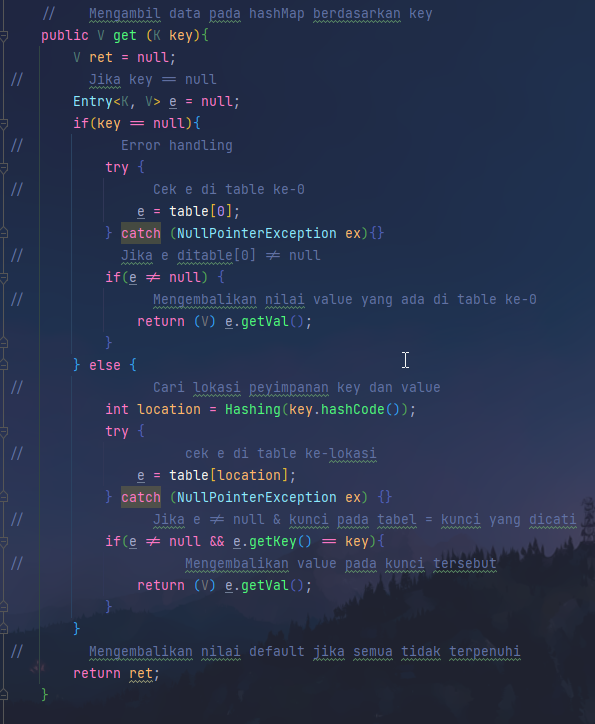
Method ini digunakan untuk melakukan apakah Value yang kita cari ada pada data apa tidak contohnya



Dalam case ini saya sudah Membuat prebuild data pada program, saya akan mencoba untuk mencari value **Muhammad Irfani,** maka outputnya akan menjadi seperti ini



**4. Method get**



Method get ini digunakan untuk melakukan penarikan data atau data akan bisa di printout oleh system dengan mengunakan systemOutPrint, contohnya

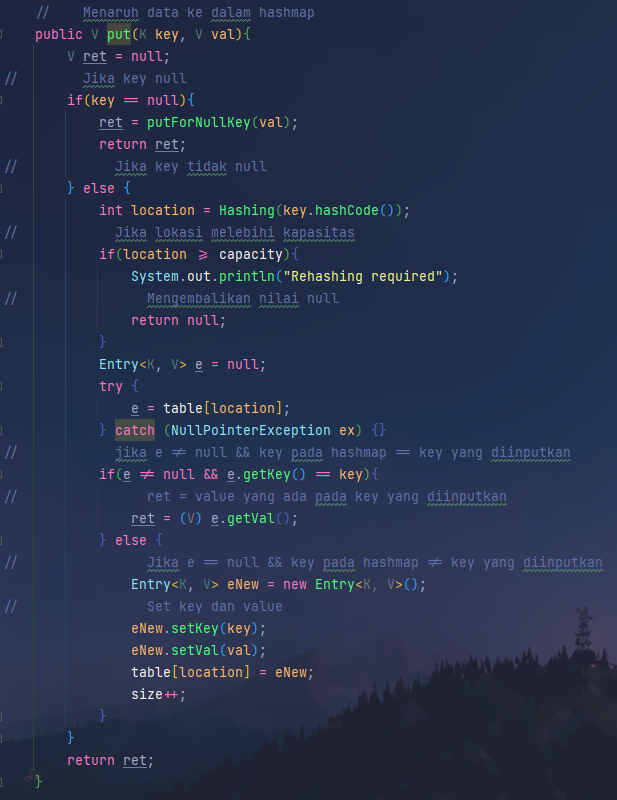
Saya akan input data dengan **key 1 dan value apel**



Maka saat saya akan melakukan print out data

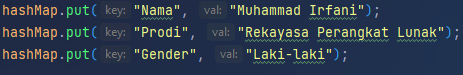


**5. Method Put**

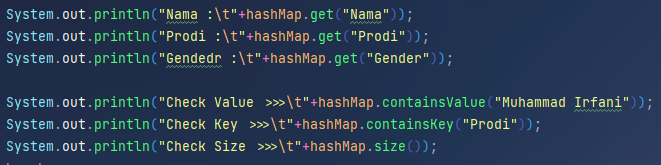


Method put ini digunakan unput melakukan input data pada hashMap, contohnya

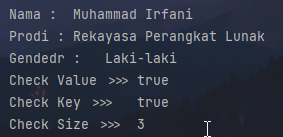
Saya akan mencoba melakukan pre-build input pada program



Maka saat saya akan mencoba melakukan output



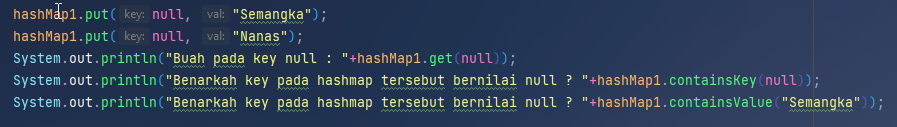
Output:



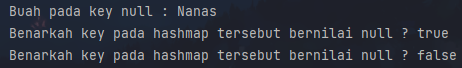
**6. putForNullKey**



Method ini digunakan untuk melakukan input Null pada key, contohnya saya akan mencoba untuk melakukan pre-build code, dengan code seperni ini

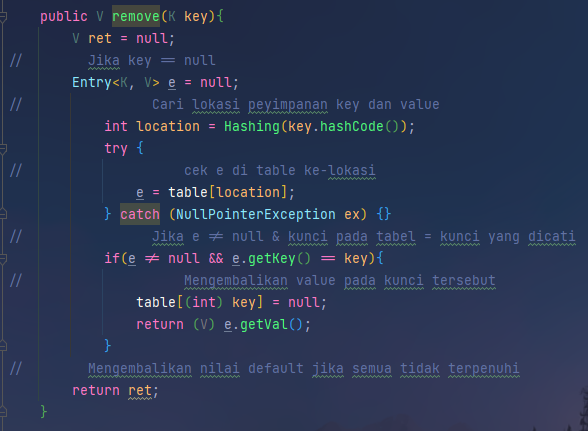


Maka akan melakukan output

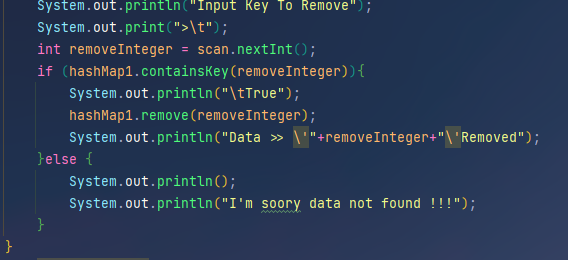


**Tugas**

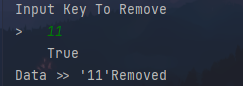
**1. Method Remove**



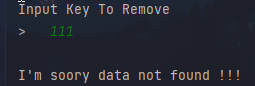
Method ini digunakan untuk melakukan remove data dengan cara user melakukan input key yang mau di hapus, berikut mainnya



Disini saya sudah inputkan data, yaitu dengan key 11, maka saat saya melakukan remove akan terjadi seperti ini



Dan jika key yang dimasukkan salah maka akan seperti ini



**BAB IV**

**Kesimpulan**

HashMap adalah sebuah class yang berisi sekumpulan pasangan nilai (value) dan kunci (key). Nilai bisa dalam bentuk string, integer, boolean, float, double, dan objek. Sedangkan untuk key biasanya dalam bentuk string dan integer, jadi intinnya hashMap itu seperti account dimana setiap akun mempunyai username dan password, terima kasih sekian dari saya, jika ada kesalahan saya mohon maaf sebesar-besarnya.

وَ السَّلاَمُ عَلَيْكُمْ وَرَحْمَةُ اللهِ وَبَرَكَاتُهُ

( Wassalamu 'alaikum Wr. Wb)