LAPORAN PRAKTIKUM
PEMROGRAMAN BERBASIS OBJEK
PRAKTIKUM 10: "EKSPRESI LAMBDA"

Disusun Oleh :

Aprilyanto Setiyawan Siburian
24060121120022

LAB C

DEPARTEMEN ILMU KOMPUTER / INFORMATIKA
FAKULTAS SAINS DAN MATEMATIKA
UNIVERSITAS DIPONEGORO
SEMARANG
2023

1. DiskonLambda.java

```
/**
 * File : DiskonLambda.java 31/05/2023
 * Nama/NIM : Aprilyanto Setiyawan Siburian
 * Deskripsi : ekspresi lambda dasar, digunakan untuk menghitung diskon
**/

interface IDiskon{
  public double hitungDiskon(int harga);
}

public class DiskonLambda{
  public static void main(String[] args){
  //tanpa lambda
  IDiskon diskonMerdeka = new IDiskon(){
        public double hitungDiskon(int harga){
              return harga-(harga*0.3);
        }
  };
  //dengan lambda
  IDiskon diskonLebaran = (harga) -> harga - (harga * 0.4);
  //dengan lambda dengan block statement
  IDiskon diskonBiasa = harga -> {
        return harga - (harga * 0.1);
  };
  System.out.println("Diskon                  Merdeka                 :
"+diskonMerdeka.hitungDiskon(45000));
  System.out.println("Diskon                  Lebaran                 :
"+diskonLebaran.hitungDiskon(45000));
  System.out.println("Diskon                   Biasa                  :
"+diskonBiasa.hitungDiskon(45000));
  }
}
```

Implementasi diskonLebaran:

```
IDiskon diskonLebaran = (harga) -> harga - (harga * 0.4);
```

Pada implementasi ini, ekspresi lambda harga - (harga * 0.4) langsung diberikan sebagai nilai kepada objek diskonLebaran yang bertipe IDiskon. Ekspresi lambda ini merupakan implementasi dari method hitungDiskon dalam interface IDiskon dengan mengurangi harga awal dengan diskon sebesar 40% (0.4).

Implementasi diskonBiasa:

```
IDiskon diskonBiasa = harga -> {
    return harga - (harga * 0.1);
};
```

Pada implementasi ini, ekspresi lambda dituliskan sebagai block statement yang mengandung pernyataan return. Block statement ini berisi implementasi dari method hitungDiskon dalam interface IDiskon, dengan mengurangi harga awal dengan diskon

sebesar 10% (0.1). Ekspresi lambda ini kemudian diberikan sebagai nilai kepada objek diskonBiasa yang bertipe IDiskon.

Dalam kedua kasus tersebut, ekspresi lambda digunakan untuk mengimplementasikan method hitungDiskon dalam interface IDiskon. Perbedaannya terletak pada cara penulisan dan penggunaan block statement dalam ekspresi lambda.

2. LambdaList.java

```
import java.util.ArrayList;
/**
 * File : LambdaList.java 31/05/2023
 * Nama/NIM : Aprilyanto Setiyawan Siburian/24060121120022
 * Deskripsi : implementasi lambda pada list, digunakan sebagai parameter
pada method
**/

public class LambdaList{
   public static void main(String[] args){
         ArrayList<String> mahasiswaList = new ArrayList<>();
         mahasiswaList.add("Adi");
         mahasiswaList.add("Bambang");
         mahasiswaList.add("Cici");
         mahasiswaList.add("Didi");
         //lambda digunakan sebagai parameter
         mahasiswaList.forEach((nama) -> System.out.println(nama));
   }
}
```

3. LambdaHashmap.java

```
/*
 * File : LambdaHashmap.java  (06/06/2023)
 * Penulis : Aprilyanto Setiyawan Siburian (24060121120022)
 * Deskripsi: Menggunakan lambda untuk menampilkan data mahasiswa
*/

import java.util.HashMap;

public class LambdaHashmap {
    public static void main(String[] args) {
        HashMap<String, String> mahasiswaMap = new HashMap<>();
        mahasiswaMap.put("24060121120022",     "Aprilyanto     Setiyawan
Siburian");
        mahasiswaMap.put("24060121130102", "Sierra");
        mahasiswaMap.put("24060121140178", "Cipung");
        mahasiswaMap.put("24060121120021", "Popoy");

        // menggunakan lambda
        mahasiswaMap.forEach((nim, nama) -> System.out.println(nim + " :
" + nama));
```

```
    }
}
```

Lambda expression akan mencetak elemen-elemen yang ada dalam sebuah HashMap yang berisi daftar mahasiswa. Pertama, kita membuat objek HashMap dengan tipe data kunci dan nilai yang berupa String. Kemudian, kita menambahkan beberapa entri ke dalam HashMap tersebut dengan menggunakan metode put(). Setelah itu, menggunakan lambda expression, kita melakukan iterasi pada setiap elemen dalam HashMap dan mencetak pasangan kunci dan nilai dengan format "kunci : nilai" menggunakan metode forEach(). Dalam lambda expression, parameter pertama merupakan kunci (nim) dan parameter kedua merupakan nilai (nama), dan kita menggunakan System.out.println() untuk mencetak hasilnya. Dengan demikian, setiap elemen dalam HashMap akan dicetak satu per satu.