

**LAPORAN PRAKTIKUM 12**  
**PEMROGRAMAN BERORIENTASI OBJEK (PBO)**



Nama : Alya Gustiani Nur 'Afifah  
NIM : 231511035  
Kelas/Prodi : 2B/D3 Teknik Informatika

**Politeknik Negeri Bandung**

**2024**

1. Link Repository GitHub : <https://github.com/alyagustiani/PBOSem2> (pertemuan 12 & FamilyGiftSystem)
2. Niece class

```
package com.familygift;

import java.util.*;

class Niece extends Person implements Comparable<Niece>, Birthday {
    private final int day;
    private final int month;
    private final Set<Present> presents;

    Niece(String name, int day, int month) {
        super(name);
        this.day = day;
        this.month = month;
        this.presents = new HashSet<>();
    }

    public int getDay() {
        return day;
    }

    public int getMonth() {
        return month;
    }

    public boolean addPresent(Present present) {
        return presents.add(present);
    }

    public Set<Present> getPresents() {
        return Collections.unmodifiableSet(presents);
    }

    public int clearPresents() {
        int count = presents.size();
        presents.clear();
        return count;
    }

    public void listPresents() {
        System.out.println("\nDaftar hadiah untuk " + getName() + ":");
        if (presents.isEmpty()) {
            System.out.println("Belum ada hadiah.");
        } else {
            presents.stream()
                .sorted(Comparator.comparing(p ->
                    p.getGiver().getName()))
                .forEach(present -> System.out.println("Dari " +
                    present.getGiver().getName() +
                        ": " + present.getDescription()));
        }
    }
}
```

```

@Override
public int compareTo(Niece other) {
    return Comparator.comparingInt((Niece n) -> n.month)
        .thenComparingInt(n -> n.day)
        .compare(this, other);
}

@Override
public boolean equals(Object o) {
    if (this == o) return true;
    if (o == null || getClass() != o.getClass()) return false;
    Niece niece = (Niece) o;
    return Objects.equals(getName(), niece.getName());
}

@Override
public int hashCode() {
    return Objects.hash(getName());
}
}

```

a. Deskripsi

Kelas Niece memperluas Person dan mengimplementasikan Birthday. Ini memungkinkan setiap objek Niece untuk memiliki nama, ulang tahun, dan daftar hadiah yang diterima.

b. Penjelasan:

- Niece memiliki atribut day, month, dan presents.
- Mengimplementasikan antarmuka Birthday untuk mendapatkan informasi ulang tahun dan Comparable untuk memungkinkan pengurutan berdasarkan ulang tahun.
- Metode addPresent menambahkan hadiah ke daftar hadiah keponakan.
- Metode clearPresents menghapus semua hadiah dari daftar dan mengembalikan jumlah hadiah yang dihapus.
- Metode listPresents menampilkan semua hadiah yang diterima oleh keponakan, diurutkan berdasarkan nama pemberi hadiah.

3. Uncle class

```

package com.familygift;

import java.util.*;

class Uncle extends Person implements Comparable<Uncle> {
    private final Set<Present> presents;

    Uncle(String name) {
        super(name);
        this.presents = new HashSet<>();
    }

    public boolean addPresent(Niece recipient, String description) {
        boolean isDuplicateGift = presents.stream()
            .anyMatch(p ->

```

```

p.getDescription().equalsIgnoreCase(description));

        boolean isDuplicateReceived = recipient.getPresents().stream()
            .anyMatch(p ->
p.getDescription().equalsIgnoreCase(description));

        if (isDuplicateGift || isDuplicateReceived) {
            System.out.println("Peringatan: Hadiah duplikat tidak
diperbolehkan!");
            return false;
        }

        Present present = new Present(this, recipient, description);
        presents.add(present);
        recipient.addPresent(present);
        return true;
    }

    public void listPresents() {
        System.out.println("\nDaftar hadiah dari " + getName() + ":");
        if (presents.isEmpty()) {
            System.out.println("Belum memberikan hadiah.");
        } else {
            presents.stream()
                .sorted(Comparator.comparing(p ->
p.getRecipient().getName()))
                .forEach(present -> System.out.println("Untuk " +
present.getRecipient().getName() +
": " + present.getDescription()));
        }
    }

    @Override
    public int compareTo(Uncle other) {
        return this.getName().compareToIgnoreCase(other.getName());
    }

    @Override
    public boolean equals(Object o) {
        if (this == o) return true;
        if (o == null || getClass() != o.getClass()) return false;
        Uncle uncle = (Uncle) o;
        return Objects.equals(getName(), uncle.getName());
    }

    @Override
    public int hashCode() {
        return Objects.hash(getName());
    }
}

```

a. Deskripsi

Kelas Uncle memperluas Person dan mengimplementasikan Comparable<Uncle>. Setiap objek Uncle memiliki nama dan daftar hadiah yang diberikan.

b. Penjelasan

- Uncle memiliki atribut presents yang merupakan Set.

- Mengimplementasikan Comparable<Uncle> untuk memungkinkan pengurutan berdasarkan nama.

#### 4. Present class

```
package com.familygift;

import java.util.*;

class Present {
    private final Uncle giver;
    private final Niece recipient;
    private final String description;

    public Present(Uncle giver, Niece recipient, String description) {
        this.giver = giver;
        this.recipient = recipient;
        this.description = description;
    }

    public Uncle getGiver() { return giver; }
    public Niece getRecipient() { return recipient; }
    public String getDescription() { return description; }

    @Override
    public boolean equals(Object o) {
        if (this == o) return true;
        if (o == null || getClass() != o.getClass()) return false;
        Present present = (Present) o;
        return Objects.equals(description, present.description);
    }

    @Override
    public int hashCode() {
        return Objects.hash(description);
    }
}
```

##### a. Alasan penambahan

Kelas Present ditambahkan untuk memisahkan logika yang terkait dengan hadiah dari kelas Uncle dan Niece, sehingga menjaga kode tetap bersih dan terorganisir. Dengan mengelola hadiah dalam kelas tersendiri, kita dapat dengan mudah menambah, menghapus, dan mengakses hadiah yang diberikan oleh paman kepada keponakan. Ini juga meningkatkan reusabilitas dan keterbacaan kode, serta memudahkan pemeliharaan dan pengembangan fitur baru terkait hadiah di masa depan.

##### b. Deskripsi

Kelas Present mewakili hadiah yang diberikan oleh seorang paman kepada seorang keponakan. Ini berisi informasi tentang pemberi, penerima, dan deskripsi hadiah.

##### c. Penjelasan

- Present memiliki atribut giver, recipient, dan description.
- Metode getGiver, getRecipient, dan getDescription memungkinkan akses ke informasi hadiah.

## 5. Family class

```
package com.familygift;

import java.util.*;

public class Family {
    private final Map<String, Uncle> uncles;
    private final Map<String, Niece> nieces;

    public Family() {
        uncles = new HashMap<>();
        nieces = new HashMap<>();
    }

    public boolean addNiece(Niece niece) {
        if (findNiece(niece.getName()) != null) {
            System.out.println("Peringatan: Keponakan dengan nama " +
niece.getName() + " sudah ada!");
            return false;
        }
        nieces.put(niece.getName().toLowerCase(), niece);
        System.out.println("Berhasil menambahkan keponakan: " +
niece.getName());
        return true;
    }

    public boolean addUncle(Uncle uncle) {
        if (findUncle(uncle.getName()) != null) {
            System.out.println("Peringatan: Paman dengan nama " +
uncle.getName() + " sudah ada!");
            return false;
        }
        uncles.put(uncle.getName().toLowerCase(), uncle);
        System.out.println("Berhasil menambahkan paman: " +
uncle.getName());
        return true;
    }

    public Niece findNiece(String name) {
        return nieces.get(name.toLowerCase());
    }

    public Uncle findUncle(String name) {
        return uncles.get(name.toLowerCase());
    }

    public void listNieces() {
        System.out.println("\nDaftar Keponakan (urut berdasarkan ulang
tahun):");
        if (nieces.isEmpty()) {
            System.out.println("Belum ada keponakan yang terdaftar.");
        } else {
            nieces.values().stream()
                .sorted()
                .forEach(niece ->
System.out.println(niece.getName()));
        }
    }
}
```

```

    }
}

public void listUncles() {
    System.out.println("\nDaftar Paman (urut alfabetis):");
    if (uncles.isEmpty()) {
        System.out.println("Belum ada paman yang terdaftar.");
    } else {
        uncles.values().stream()
            .sorted()
            .forEach(uncle ->
                System.out.println(uncle.getName()));
    }
}

// Method untuk testing
public void generateSampleData() {
    // Menambahkan paman
    addUncle(new Uncle("Albert"));
    addUncle(new Uncle("Bill"));
    addUncle(new Uncle("Charlie"));

    // Menambahkan keponakan
    addNiece(new Niece("Amy", 15, 3));
    addNiece(new Niece("Beatrice", 22, 6));
    addNiece(new Niece("Claire", 10, 9));

    // Menambahkan beberapa hadiah
    Uncle albert = findUncle("Albert");
    Uncle bill = findUncle("Bill");
    Uncle charlie = findUncle("Charlie");

    Niece amy = findNiece("Amy");
    Niece beatrice = findNiece("Beatrice");
    Niece claire = findNiece("Claire");

    albert.addPresent(amy, "The Wonder of Computers");
    albert.addPresent(beatrice, "Programming Book");
    albert.addPresent(claire, "Art Kit");

    bill.addPresent(amy, "Painting Set");
    bill.addPresent(beatrice, "Science Kit");

    charlie.addPresent(claire, "Story Book");

    // Mencoba menambahkan hadiah duplikat
    bill.addPresent(claire, "Art Kit"); // Seharusnya gagal karena
    duplikat
}
}

```

a. Deskripsi

Kelas Family mengelola daftar paman dan keponakan. Ini memungkinkan penambahan, pencarian, dan pengurutan anggota keluarga.

b. Penjelasan

- Family memiliki dua atribut uncles dan nieces yang merupakan Map.

- Metode `addNiece` dan `addUncle` memungkinkan penambahan anggota keluarga ke dalam `Map` dengan pengecekan duplikasi.
- c. Penggunaan Generics
- Deklarasi kelas: `<U extends Person, N extends Person & Birthday>`.
  - Deklarasi metode: `public boolean addNiece(N niece)` dan `public boolean addUncle(U uncle)`.
  - Atribut kelas: `private final Map<String, U> uncles` dan `private final Map<String, N> nieces`.

Memungkinkan `Family` untuk mengelola berbagai tipe data yang memperluas `Person` dan mengimplementasikan `Birthday`.

## 6. Person abstract class

```
package com.familygift;

abstract class Person {
    private final String name;

    public Person(String name) {
        this.name = name;
    }

    public String getName() {
        return name;
    }
}
```

### a. Alasan penambahan

Kelas `Person` ditambahkan untuk mengabstraksi properti umum dari anggota keluarga seperti `Uncle` dan `Niece`. Ini membantu mengurangi duplikasi kode dengan menyediakan atribut dan metode umum, seperti nama, di satu tempat. Dengan adanya kelas dasar ini, kita memastikan bahwa setiap anggota keluarga memiliki nama dan metode `getName`, sehingga memudahkan pengelolaan dan pemeliharaan kode.

### b. Deskripsi

Kelas `Person` bertindak sebagai kelas dasar umum untuk setiap anggota keluarga yang memiliki nama. Kelas ini menyediakan satu metode.

### c. Penjelasan

- `Person` memiliki atribut `name` yang merupakan `String`.
- Metode `getName` mengembalikan nama anggota keluarga.

## 7. Birthday interface

```
package com.familygift;

interface Birthday {
    int getDay();
    int getMonth();
}
```



a. Alasan penambahan

Antarmuka Birthday ditambahkan untuk mengelola informasi ulang tahun dengan cara yang lebih terstruktur. Dengan mendefinisikan metode `getDay` dan `getMonth`, kita memastikan bahwa setiap kelas yang memerlukan informasi ulang tahun dapat mengimplementasikan antarmuka ini. Ini sangat berguna untuk pengurutan dan pengelolaan data ulang tahun, serta memberikan fleksibilitas jika ada entitas lain yang memerlukan informasi ulang tahun di masa depan.

b. Deskripsi

Antarmuka Birthday menyediakan dua metode untuk mendapatkan hari dan bulan ulang tahun.

c. Penjelasan

- Birthday memiliki dua metode `getDay` dan `getMonth` untuk mendapatkan informasi ulang tahun.
- Digunakan oleh kelas yang memerlukan informasi ulang tahun untuk pengurutan.

8. Main class

```
package com.familygift;

public class Main {
    public static void main(String[] args) {
        Family family = new Family();

        // Generate sampel data
        family.generateSampleData();

        // Menampilkan semua data
        family.listUncles();
        family.listNieces();

        // Menampilkan hadiah dari setiap paman
        Uncle albert = family.findUncle("Albert");
        Uncle bill = family.findUncle("Bill");
        Uncle charlie = family.findUncle("Charlie");

        if (albert != null) albert.listPresents();
        if (bill != null) bill.listPresents();
        if (charlie != null) charlie.listPresents();

        // Menampilkan hadiah untuk setiap keponakan
        Niece amy = family.findNiece("Amy");
        Niece beatrice = family.findNiece("Beatrice");
        Niece claire = family.findNiece("Claire");

        if (amy != null) amy.listPresents();
        if (beatrice != null) beatrice.listPresents();
        if (claire != null) claire.listPresents();
    }
}
```

a. Deskripsi

Kelas Main berfungsi sebagai titik masuk untuk program ini. Di sini, kita membuat objek Family dan menguji berbagai fungsionalitas dengan menambahkan data sampel dan menampilkan informasi tentang anggota keluarga.

b. Penjelasan

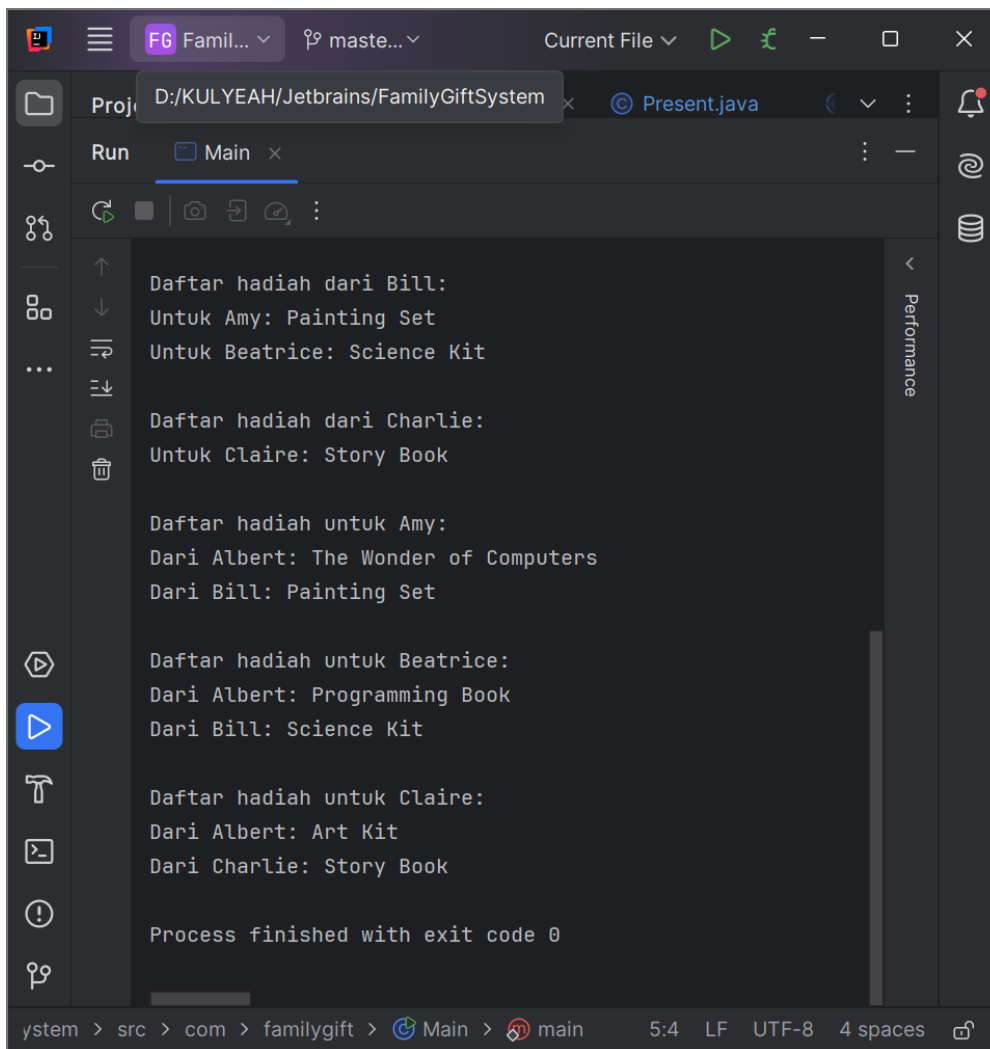
- Kelas `Main` adalah tempat eksekusi utama program terjadi.
- Membuat objek `Family` dan memanggil metode untuk mengelola data anggota keluarga.

c. Penggunaan Generics

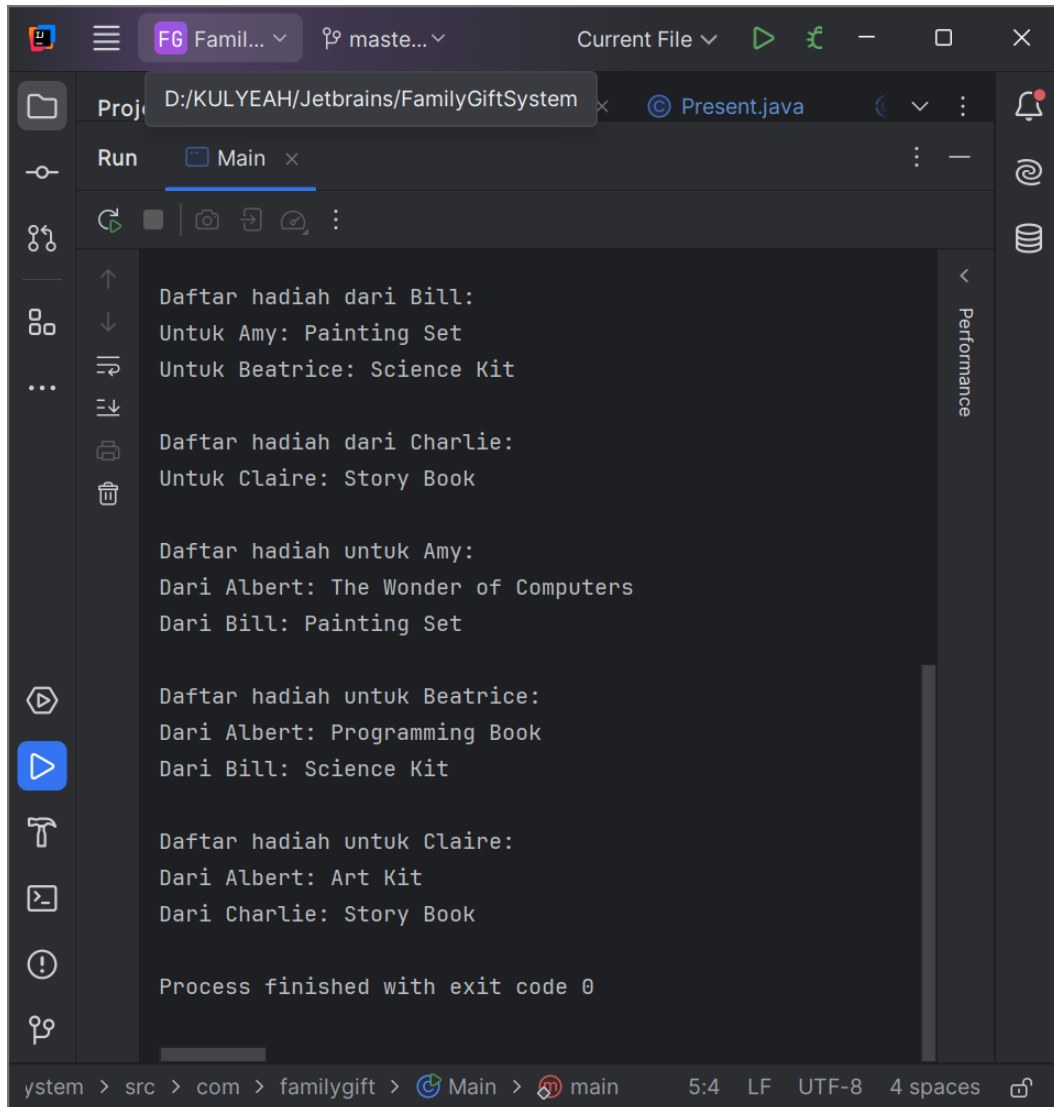
- `Family<Uncle, Niece> family = new Family<>()`

Memungkinkan penggunaan kelas `Family` dengan tipe data lain yang sesuai di masa mendatang.

## Hasil Akhir



```
Daftar hadiah dari Bill:  
Untuk Amy: Painting Set  
Untuk Beatrice: Science Kit  
  
Daftar hadiah dari Charlie:  
Untuk Claire: Story Book  
  
Daftar hadiah untuk Amy:  
Dari Albert: The Wonder of Computers  
Dari Bill: Painting Set  
  
Daftar hadiah untuk Beatrice:  
Dari Albert: Programming Book  
Dari Bill: Science Kit  
  
Daftar hadiah untuk Claire:  
Dari Albert: Art Kit  
Dari Charlie: Story Book  
  
Process finished with exit code 0
```



The screenshot shows an IDE window with a dark theme. The top bar includes a menu icon, a project name 'FG Famil...', a file name 'maste...', and a 'Current File' dropdown. The main editor area displays the output of a Java program. The output is organized into sections for each person's gift list, followed by a final summary and a process completion message. The status bar at the bottom shows the file path 'ystem > src > com > familygift > Main > main', line and column numbers '5:4', and encoding 'LF UTF-8 4 spaces'.

```
Daftar hadiah dari Bill:  
Untuk Amy: Painting Set  
Untuk Beatrice: Science Kit  
  
Daftar hadiah dari Charlie:  
Untuk Claire: Story Book  
  
Daftar hadiah untuk Amy:  
Dari Albert: The Wonder of Computers  
Dari Bill: Painting Set  
  
Daftar hadiah untuk Beatrice:  
Dari Albert: Programming Book  
Dari Bill: Science Kit  
  
Daftar hadiah untuk Claire:  
Dari Albert: Art Kit  
Dari Charlie: Story Book  
  
Process finished with exit code 0
```