Lembar Kerja Peserta Didik

# Modul 4 - OOP Lanjutan

# A. PETUNJUK PENGERJAAN

1. LKPD ini dikerjakan secara berkelompok.
2. Kelompok menggunakan susunan anggota kelompok yang sudah ditetapkan di awal.
3. Sebelum kalian mengerjakan LKPD ini secara berkelompok, pastikan anda sudah bergabung di GitHub
4. LKPD diisi sesuai dengan perintah yang telah dijelaskan pada setiap soalnya.
5. Jika kalian belum memahami instruksi yang diberikan di dalam LKPD, mintalah penjelasan dari bapak/ibu guru.
6. Setelah selesai mengerjakan soal, persiapkan diri kalian untuk melakukan presentasi penjelasan kode.
7. **DIPERBOLEHKAN MENGGUNAKAN CHATGPT SELAMA PROSES PENGERJAAN.**

# B. POIN TAMBAHAN

1. Apabila kalian mengerjakan tutorial *visibility modifiers* yang diberikan pada **MODUL 4.**
2. Apabila kalian mengerjakan dan mengumpulkan tepat waktu **(WAKTU PENGERJAAN 30 MENIT)**.

# C. PETUNJUK PENGUMPULAN

1. Pengumpulan tugas dilakukan perwakilan oleh ketua kelompok.
2. File yang dikumpulkan cukup dokumen ini saja dengan format .pdf.
    a. Cara mendownload dokumen dengan format pdf:
        i. Buka opsi file yang ada di pojok kiri atas layar.
        ii. Kemudian pilih opsi download.
        iii. Selanjutnya pilih tipe file PDF Documents.
3. Lalu kumpulkan file pada GitHub yang telah disediakan.

# D. REFERENSI TAMBAHAN UNTUK BELAJAR

1. Kalian bisa melihat video dari YouTube berikut apabila kalian masih belum paham sepenuhnya mengenai materi yang dipelajari sebelumnya:
    1) Penjelasan *visibility modifiers* ➜ Belajar Kotlin OOP - 27 Visibility Modifier
    2) Tutorial dasar *visibility modifiers* ➜ Tutorial Kotlin Dasar - 18. Visibility Modifiers

# D. SOAL

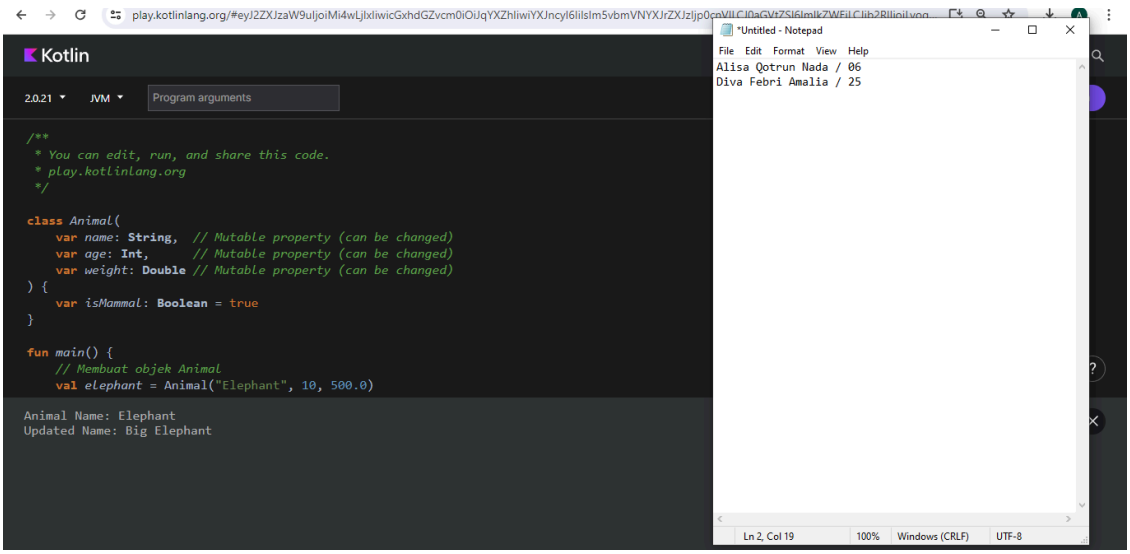
Nama Kelompok : HOK

Nama/No.Absen Anggota Kelompok :

1) Alisa Qotrun Nada / 06
2) Diva Febri Amalia / 25

**TUTORIAL 1 - TUTORIAL PENGGUNAAN VISIBILITY MODIFIERS PUBLIC PADA KOTLIN**

1. Lampirkan gambar *screenshot* kode kalian dan sertakan notepad yang berisi nama anggota kelompok kalian!




**Notes:** Gambar hanya contoh, jadi bisa kalian hapus ya.

2. Lampirkan kode yang kalian buat.

```
class Animal(
    var name: String,  // Mutable property (can be changed)
    var age: Int,      // Mutable property (can be changed)
    var weight: Double // Mutable property (can be changed)
) {
    var isMammal: Boolean = true
}

fun main() {
    // Membuat objek Animal
    val elephant = Animal("Elephant", 10, 500.0)

    // Mengakses dan memodifikasi properti publik
```

```
    println("Animal Name: ${elephant.name}") // Output:
Elephant
    elephant.name = "Big Elephant"           // Modifying the
name
    println("Updated Name: ${elephant.name}") // Output: Big
Elephant
}
```

**Notes:** Kode hanya contoh, jadi bisa kalian hapus ya.

3. Selanjutnya jelaskan secara singkat cara kerja kode kalian!

**Kelas `Animal`**:

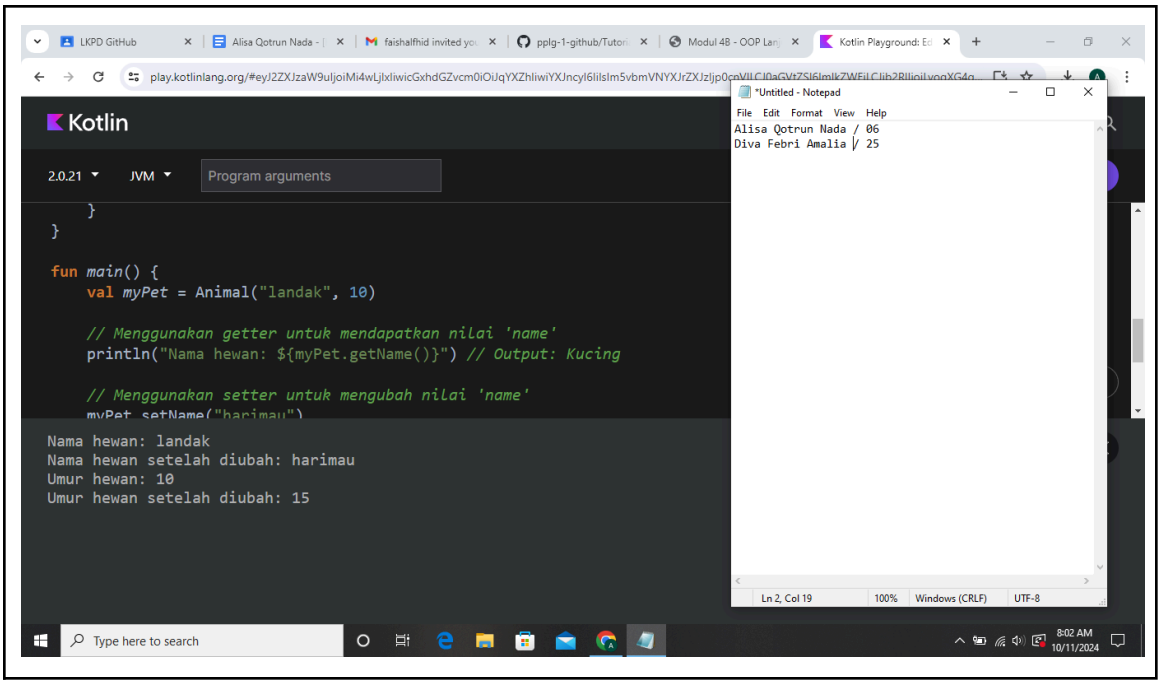
- Memiliki tiga properti **mutable** (dapat diubah):
  - `name`: Nama hewan.
  - `age`: Usia hewan.
  - `weight`: Berat hewan.
- Juga memiliki properti `isMammal` dengan nilai default `true`, yang menunjukkan apakah hewan tersebut adalah mamalia.

**Fungsi `main()`**:

- Membuat objek `elephant` dari kelas `Animal` dengan nama "Elephant", usia 10 tahun, dan berat 500 kg.
- Mengakses properti `name` untuk menampilkan nama hewan.
- Memodifikasi properti `name` dari "Elephant" menjadi "Big Elephant".
- Menampilkan nama yang telah diperbarui.

## TUTORIAL 2 - TUTORIAL PENGGUNAAN VISIBILITY MODIFIERS PRIVATE PADA KOTLIN

1. Lampirkan gambar screenshot kode kalian dan sertakan notepad yang berisi nama anggota kelompok kalian!




2. Lampirkan kode yang kalian buat.

```
class Animal(private var name: String, private var age: Int) {

    // Getter untuk name
    fun getName(): String {
        return name
    }

    // Setter untuk name
    fun setName(newName: String) {
        name = newName
    }

    // Getter untuk age
    fun getAge(): Int {
        return age
    }

    // Setter untuk age
    fun setAge(newAge: Int) {
        age = newAge
    }
}

fun main() {
    val myPet = Animal("landak", 10)
```

```
    // Menggunakan getter untuk mendapatkan nilai 'name'
    println("Nama hewan: ${myPet.getName()}") // Output: Kucing

    // Menggunakan setter untuk mengubah nilai 'name'
    myPet.setName("harimau")
    println("Nama hewan setelah diubah: ${myPet.getName()}") // Output: Banteng

    // Menggunakan getter untuk mendapatkan nilai 'age'
    println("Umur hewan: ${myPet.getAge()}") // Output: 2

    // Menggunakan setter untuk mengubah nilai 'age'
    myPet.setAge(15)
    println("Umur hewan setelah diubah: ${myPet.getAge()}") // Output: 3
}
```

3. Selanjutnya jelaskan secara singkat cara kerja kode kalian!

**Kelas `Animal`**:

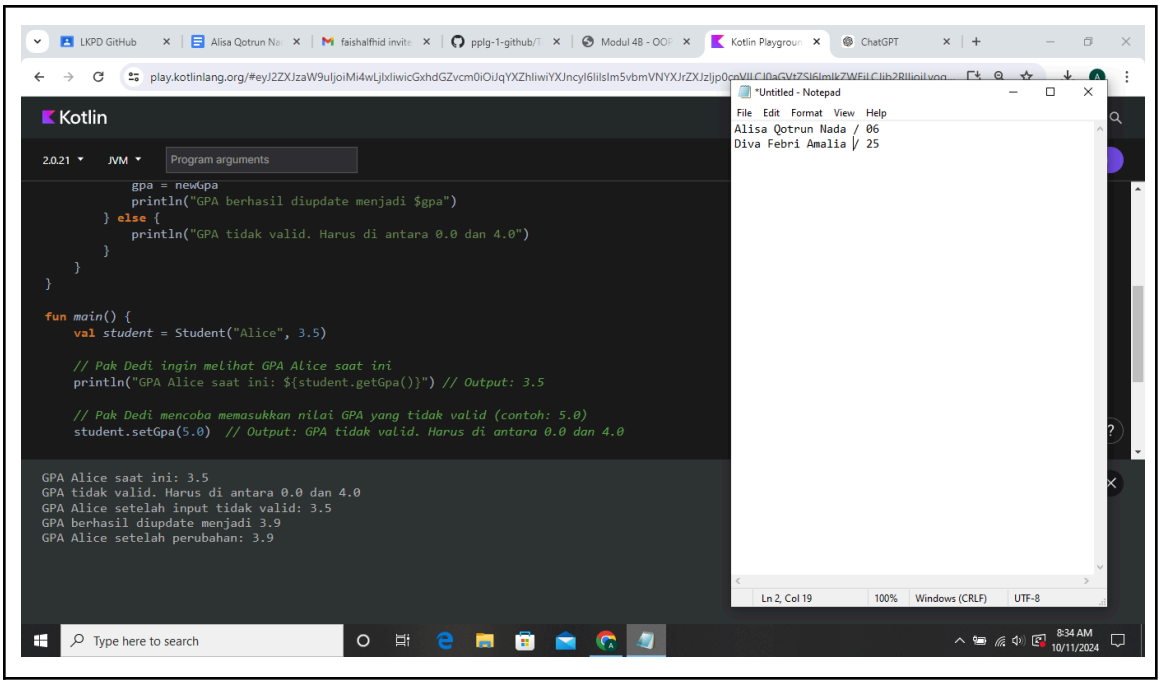
- Memiliki dua properti privat:
  - `name`: Nama hewan.
  - `age`: Usia hewan.
- Fungsi `getName()` digunakan untuk mengakses (getter) nilai `name`.
- Fungsi `setName(newName: String)` digunakan untuk mengubah (setter) nilai `name`.
- Fungsi `getAge()` digunakan untuk mengakses nilai `age`.
- Fungsi `setAge(newAge: Int)` digunakan untuk mengubah nilai `age`.

**Fungsi `main()`**:

- Membuat objek `myPet` dari kelas `Animal` dengan nama "Kucing" dan usia 2 tahun.
- Menggunakan fungsi getter untuk menampilkan nama hewan, dan fungsi setter untuk mengubah nama menjadi "Banteng".
- Menggunakan getter untuk menampilkan usia hewan, dan setter untuk mengubah usia menjadi 3.

**Studi kasus ke 2 - Nilai GPA (IPK) Mahasiswa**

1. Lampirkan gambar screenshot kode kalian dan sertakan notepad yang berisi nama anggota kelompok kalian!




2. Lampirkan kode yang kalian buat.

```
class Student(val name: String, private var gpa: Double) {

    // Fungsi untuk mendapatkan nilai gpa
    fun getGpa(): Double {
        return gpa
    }

    // Fungsi untuk mengatur nilai gpa dengan validasi
    fun setGpa(newGpa: Double) {
        if (newGpa in 0.0..4.0) {
            gpa = newGpa
            println("GPA berhasil diupdate menjadi $gpa")
        } else {
            println("GPA tidak valid. Harus di antara 0.0 dan 4.0")
        }
    }
}

fun main() {
    val student = Student("Alice", 3.5)

    // Pak Dedi ingin melihat GPA Alice saat ini
    println("GPA Alice saat ini: ${student.getGpa()}") // Output: 3.5

    // Pak Dedi mencoba memasukkan nilai GPA yang tidak valid (contoh: 5.0)
```

```
    student.setGpa(5.0)  // Output: GPA tidak valid. Harus di antara 0.0 dan 4.0

    // Lihat apakah nilai GPA berubah setelah input tidak valid
    println("GPA Alice setelah input tidak valid: ${student.getGpa()}") // Output: 3.5

    // Pak Dedi memperbarui GPA Alice ke nilai yang valid (contoh: 3.9)
    student.setGpa(3.9)  // Output: GPA berhasil diupdate menjadi 3.9

    // Tampilkan GPA setelah perubahan
    println("GPA Alice setelah perubahan: ${student.getGpa()}") // Output: 3.9
}
```

3. Selanjutnya jelaskan secara singkat cara kerja kode kalian!

**Getter (`getGpa`)**:

- Fungsi ini digunakan untuk mengembalikan nilai `gpa`.

**Setter (`setGpa`)**:

- Fungsi ini digunakan untuk memperbarui nilai `gpa` dengan validasi. Nilai GPA harus berada dalam rentang 0.0 hingga 4.0.
- Jika nilai GPA valid, maka nilai akan diperbarui.
- Jika nilai GPA tidak valid, akan muncul pesan error.