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

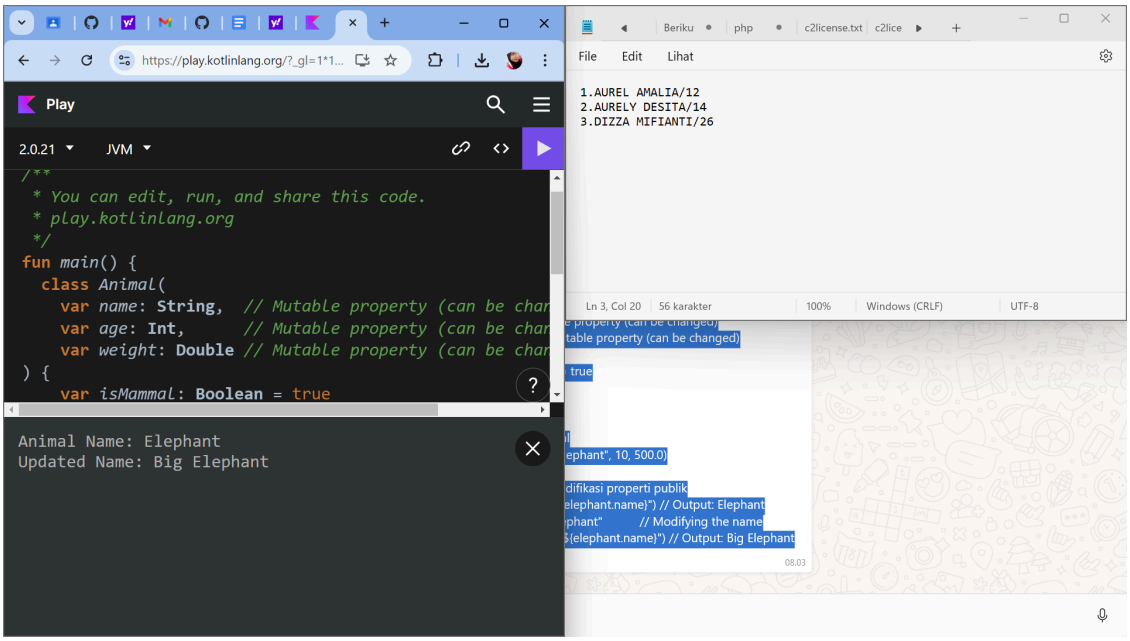
Nama Kelompok :.....ZOMBIE DORA.....

Nama/No.Absen Anggota Kelompok :

1) ...aurel amalia/12
2) ...aurely desita/14
3) ...dizza mifianti/26
4) ...
5) dst..

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.

```
/**
 * You can edit, run, and share this code.
 * play.kotlinlang.org
 */
fun main() {
  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
}


    // 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




}
    }
```

3. Selanjutnya jelaskan secara singkat cara kerja kode kalian!

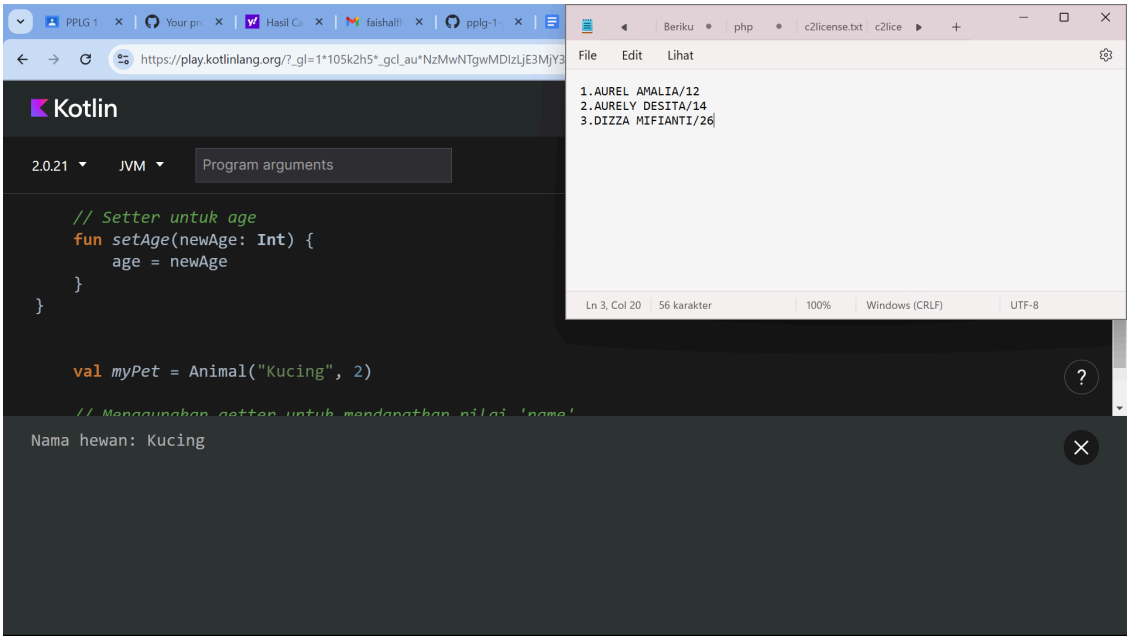
Class Animal:

Kelas Animal memiliki tiga properti yang dapat diubah (mutable), yaitu:
name: Nama hewan (tipe String).
age: Umur hewan (tipe Int).
weight: Berat hewan (tipe Double).
Ada juga properti isMammal, yang secara default diinisialisasi dengan nilai true.
Fungsi main:

Membuat objek elephant dari kelas Animal dengan nilai awal: "Elephant" (nama), 10 (umur), dan 500.0 (berat).
Mengakses dan mencetak properti name dari objek elephant.
Memodifikasi properti name menjadi "Big Elephant".
Mencetak nama yang sudah 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.

```
fun main() {
    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
    }
```

```
}


    val myPet = Animal("Kucing", 2)

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

    // Menggunakan setter untuk mengubah nilai 'name'
    myPet.setName("Banteng")

}
```

3. Selanjutnya jelaskan secara singkat cara kerja kode kalian!

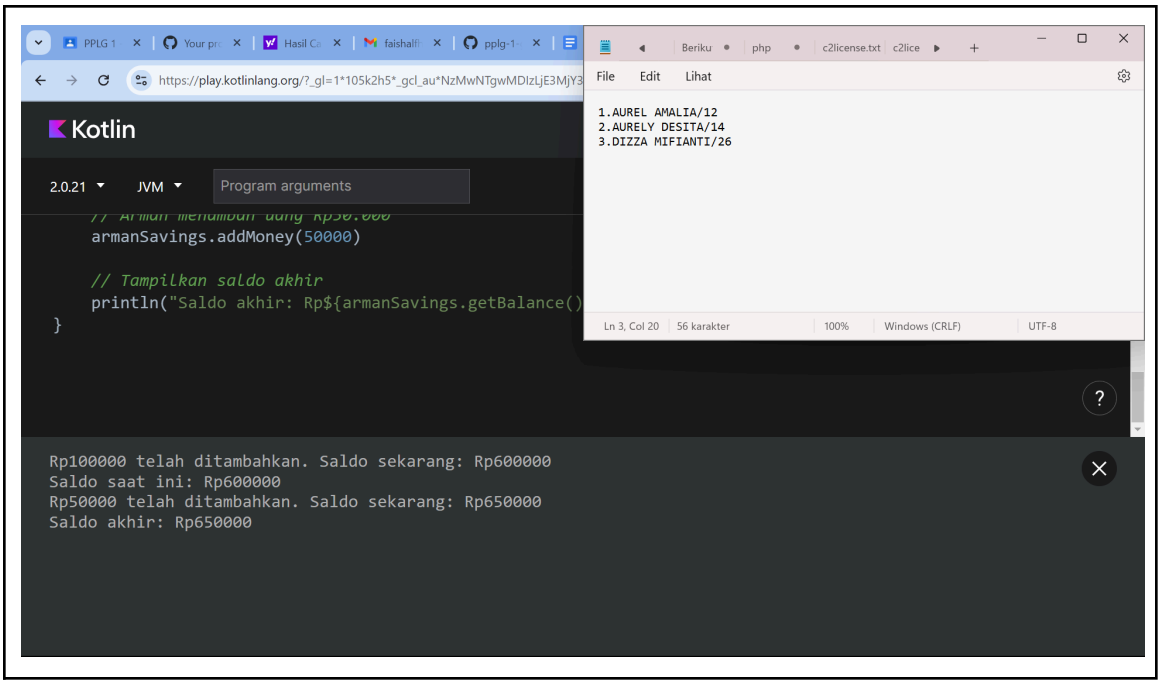
Class Animal:

Dua properti, name dan age, dideklarasikan sebagai private, sehingga tidak bisa diakses langsung dari luar kelas.
Fungsi getter (getName() dan getAge()) digunakan untuk mengembalikan nilai properti.
Fungsi setter (setName() dan setAge()) digunakan untuk mengubah nilai properti.
Fungsi main:

Membuat objek myPet dari kelas Animal dengan nilai awal "Kucing" sebagai nama dan 2 sebagai umur.
Menggunakan getter (getName()) untuk mencetak nama awal hewan.
Menggunakan setter (setName()) untuk mengubah nama hewan menjadi "Banteng".
Mencetak nama yang sudah diubah.
Menggunakan getter (getAge()) untuk mencetak umur hewan awal.
Menggunakan setter (setAge()) untuk mengubah umur hewan menjadi 3.
Mencetak umur yang sudah diubah.

Studi kasus berapa kamu? 3 - Nama studi kasusnya apa?

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



2. Lampirkan kode yang kalian buat.

```
fun main() {

  class Savings(private var balance: Int) {

    // Fungsi untuk menambahkan uang
    fun addMoney(amount: Int) {
        if (amount > 0) {
            balance += amount
            println("Rp$amount telah ditambahkan. Saldo sekarang: Rp$balance")
        } else {
            println("Jumlah yang ditambahkan harus lebih dari 0.")
        }
    }

    // Fungsi untuk melihat saldo saat ini
    fun getBalance(): Int {
        return balance
    }
}


    val armanSavings = Savings(500000)  // Saldo awal Rp500.000

    // Arman menambah uang Rp100.000
    armanSavings.addMoney(100000)
```

```
    // Tampilkan saldo saat ini
    println("Saldo saat ini: Rp${armanSavings.getBalance()}")

    // Arman menambah uang Rp50.000
    armanSavings.addMoney(50000)

    // Tampilkan saldo akhir
    println("Saldo akhir: Rp${armanSavings.getBalance()}")
}
```

3. Selanjutnya jelaskan secara singkat cara kerja kode kalian!

Class Savings:

Properti balance dideklarasikan sebagai private sehingga tidak dapat diakses langsung dari luar kelas.
Fungsi addMoney(amount: Int) menambahkan jumlah uang yang diberikan ke dalam saldo (balance). Fungsi ini juga mengecek apakah jumlah uang yang ditambahkan lebih dari 0.
Fungsi getBalance() mengembalikan nilai saldo saat ini.
Fungsi main:

Membuat objek armanSavings dengan saldo awal Rp500.000.
Arman menambahkan Rp100.000 ke dalam saldo, kemudian saldo diperbarui dan ditampilkan.
Arman menambahkan Rp50.000 lagi, dan saldo akhir ditampilkan.*