PAM Pengembangan Aplikasi Mobile

Laporan Praktikum 3

NIM
11S21014
Nama
Dedi Andre Martua Raja Panggabean
Tanggal dibuat
28 February 2024, 11:07

score
98.03

Praktikum 3 (100%)

98.03 / 100

Code Analysis

20 / 20

A. Kotlin OOP

100 / 100

1. Class & Object

100 / 100

Catatan

1. Kelas Mahasiswa:
- Ini adalah contoh kelas dengan constructor primer.
- Properti yang dideklarasikan dalam constructor primer dapat diakses di dalam kelas.
- Metode tampilkanInfo() digunakan untuk menampilkan informasi mahasiswa.

2. Kelas Buah:
- Ini adalah contoh kelas dengan secondary constructor.
- Properti kelas diinisialisasi melalui secondary constructor.
- Metode tampilkanInfo() digunakan untuk menampilkan informasi buah.

3. Kelas Barang:
- Ini adalah contoh kelas dengan constructor primer dan secondary constructor.
- Secondary constructor dipanggil dengan kata kunci this.
- Metode tampilkanInfo() digunakan untuk menampilkan informasi barang.

4. Kelas Mobil:
- Ini adalah contoh kelas dengan blok init untuk inisialisasi properti.
- Saat objek mobil dibuat, pesan "Objek Mobil dibuat" akan dicetak.
- Metode tampilkanInfo() digunakan untuk menampilkan informasi mobil.

5. Data class Person:
- Ini adalah contoh data class yang menyediakan fungsi umum seperti toString(), equals(), dan copy().
- Properti hanya diinisialisasi dalam constructor primer.
- Metode toString() digunakan untuk mencetak informasi tentang objek.

Dari main() function:
- Objek dari setiap kelas dibuat dan diinisialisasi dengan nilai yang sesuai.
- Metode tampilkanInfo() dipanggil untuk menampilkan informasi yang relevan dari setiap objek.
- Contoh penggunaan apply{} untuk menginisialisasi properti objek mobil.

kode ini mengilustrasikan penggunaan constructor primer, constructor sekunder, blok init, dan data class dalam Kotlin. Hal ini membantu memahami berbagai cara untuk menginisialisasi objek dan bekerja dengan kelas dalam bahasa pemrograman Kotlin.

2. Encapsulation

100 / 100

Catatan

1. Kelas Person:
- Ini adalah kelas yang menyimpan informasi tentang seseorang, seperti nama dan usia.
- Properti name dan age dideklarasikan sebagai privat, artinya hanya dapat diakses di dalam kelas Person itu sendiri.

2. Metode Getter dan Setter:
- Metode getName() dan setName(newName: String) digunakan untuk mengakses dan mengatur nilai properti name.
- Metode getAge() dan setAge(newAge: Int) digunakan untuk mengakses dan mengatur nilai properti age.
- Contoh validasi juga diberikan dalam metode setter untuk properti age, di mana usia harus bernilai positif.

3. Fungsi main():
- Objek person dari kelas Person dibuat.
- Metode setter digunakan untuk mengatur nilai properti name dan age.
- Metode getter digunakan untuk mendapatkan nilai properti name dan age.
- Nilai yang diperoleh kemudian dicetak untuk ditampilkan.

4. Hasil:
- Setelah eksekusi, program akan mencetak nama dan usia yang telah diatur sebelumnya.

3. Inheritance

100 / 100

Catatan

penggunaan kelas abstrak (Binatang) sebagai kelas induk, dengan metode abstrak (bersuara()) yang harus diimplementasi oleh kelas turunannya. Selanjutnya, kelas turunan (Kucing) mewarisi sifat-sifat dan metode dari kelas induk, sambil menambahkan metode tambahan khusus untuk kelas itu sendiri.

4. Polymorphism

100 / 100

Catatan

- Overloading dan overriding adalah dua cara yang berbeda untuk menerapkan polimorfisme dalam pemrograman. Overloading terjadi pada compile-time, di mana metode yang akan dipanggil ditentukan oleh compiler berdasarkan jumlah atau jenis parameter. Sementara overriding terjadi pada runtime, di mana metode yang akan dipanggil ditentukan saat program berjalan, berdasarkan tipe objek yang sebenarnya saat itu.
- Overriding memungkinkan kelas turunan untuk memberikan implementasi yang berbeda untuk metode yang sama yang dideklarasikan di kelas induknya. Hal ini memungkinkan untuk penggunaan polimorfisme yang lebih fleksibel dan dinamis.

5. Interface

100 / 100

Catatan

onsep interface untuk mendefinisikan kemampuan objek dan kemudian mengimplementasikannya dalam sebuah kelas. Dengan menggunakan fungsi yang menerima objek yang mengimplementasikan interface, program dapat secara polimorfik

mengoperasikan objek tanpa memperhatikan detail implementasinya, meningkatkan fleksibilitas dan modularitas kode.

6. ObjectCompanion

100 / 100

Catatan

Kelas `KonverterTemperatur` menggunakan companion object untuk menyimpan konstanta dan logika terkait konversi suhu antara Celsius dan Fahrenheit. Metode dalam companion object memungkinkan penggunaan kelas tanpa perlu membuat instance langsung, sehingga memudahkan untuk melakukan konversi suhu secara langsung menggunakan metode yang tersedia.

7. Package

100 / 100

Catatan

1.Definisi: Package adalah cara untuk mengorganisir dan mengelompokkan kode dalam Kotlin.

2.Deklarasi: Package dideklarasikan di awal file Kotlin menggunakan kata kunci package.

3.Struktur: Struktur package mencerminkan struktur proyek dan membantu dalam penataan kode.

4.Subpackage: Anda dapat memiliki package di dalam package, yang disebut subpackage.

5.Akses Modifier: Anggota package secara default dapat diakses dari mana pun dalam package tersebut.

6.Package-Level Functions dan Properti: Anda dapat menempatkan fungsi dan properti di level package.

7.Impor Package: Menggunakan pernyataan import untuk mengakses kelas atau fungsi dari package lain.

8.Wildcards Import: Mengimpor seluruh isi package dengan menggunakan wildcard .

9.Konvensi Nama Package: Biasanya menggunakan urutan terbalik dari domain Anda.

10.Package Private: Kotlin mendukung aksesibilitas package-private dengan mendeklarasikan anggota sebagai internal

B. Studi Kasus Aplikasi Todolist

100 / 100

1. Membuat Entity

100 / 100

Catatan

1.Definisi: Kelas Todolist adalah representasi dari sebuah entitas daftar tugas (to-do list).

2.Properti: Properti utama dari kelas ini adalah todo, yang merupakan teks dari tugas yang perlu dilakukan.

3.Konstruktor: Kelas memiliki dua konstruktor. Konstruktor pertama adalah konstruktor default yang menginisialisasi todo dengan string kosong. Konstruktor kedua memungkinkan inisialisasi todo dengan nilai yang diberikan.

4.Getter dan Setter: Kelas memiliki metode getTodo() untuk mendapatkan nilai dari properti todo, dan metode setTodo() untuk mengatur nilai properti todo.

5.Aksesibilitas Properti: Properti todo ditandai sebagai private, yang berarti hanya dapat diakses dari dalam kelas itu sendiri.

6.Keterkaitan dengan To-Do List: Kelas ini dirancang untuk menyimpan informasi tentang sebuah tugas dalam daftar to-do list, dan menyediakan metode untuk mengakses dan mengubah tugas tersebut..

2. Membuat Repository

100 / 100

Catatan

1. TodolistRepository:
- Interface yang mendefinisikan operasi dasar yang dapat dilakukan pada repositori to-do list.
- Metode yang didefinisikan termasuk getAll() untuk mendapatkan semua entri to-do list, add() untuk menambahkan entri baru, remove() untuk menghapus entri berdasarkan nomor, update() untuk memperbarui entri yang ada, search() untuk mencari entri berdasarkan kata kunci, dan sort() untuk mengurutkan entri.

2. TodolistRepositoryImpl:
- Implementasi dari TodolistRepository.
- Menggunakan ArrayList untuk menyimpan entri to-do list.
- Mengimplementasikan semua metode yang didefinisikan dalam TodolistRepository.
- Metode sort() mengurutkan entri to-do list berdasarkan kata kunci "a" untuk ascending dan "d" untuk descending. Jika kata kunci tidak valid, akan dicetak pesan kesalahan.

3. Membuat Service

100 / 100

Catatan

1. TodolistService:
- Interface yang mendefinisikan layanan atau operasi yang dapat dilakukan pada to-do list.
- Metode yang didefinisikan termasuk showTodoList() untuk menampilkan seluruh to-do list, addTodoList() untuk menambahkan entri baru, removeTodoList() untuk menghapus entri berdasarkan nomor, updateTodoList() untuk memperbarui entri yang ada, searchTodoList() untuk mencari entri berdasarkan kata kunci, dan sortTodoList() untuk mengurutkan entri.

2. TodolistServiceImpl:
- Implementasi dari TodolistService.
- Menggunakan objek TodolistRepository untuk berinteraksi dengan data to-do list.
- Mengimplementasikan semua metode yang didefinisikan dalam TodolistService.
- Metode sortTodoList() menggunakan repositori untuk melakukan pengurutan to-do list berdasarkan kata kunci "a" untuk ascending dan "d" untuk descending, dan mencetak pesan sesuai dengan hasilnya.

4. Membuat Util

100 / 100

Catatan

1. InputUtil:
- Kelas utilitas untuk membantu dalam menerima input dari pengguna.
- Mendefinisikan sebuah metode input yang dapat dipanggil secara langsung tanpa perlu menginstansiasi objek kelas.
- Metode input menerima sebuah string info sebagai parameter, mencetak informasi tersebut ke layar, dan mengembalikan input yang diberikan oleh pengguna.
- Jika input yang diberikan oleh pengguna adalah null, metode akan melemparkan NullPointerException.

5. Membuat View

100 / 100

Catatan

1. TodolistView:
- Kelas yang bertanggung jawab untuk menampilkan antarmuka pengguna untuk aplikasi to-do list.
- Memiliki metode showTodoList() yang memulai loop untuk menampilkan daftar to-do list dan menu, serta mengelola input pengguna untuk menambah, menghapus, mengubah, mencari, atau mengurutkan entri to-do list.
- Memiliki metode privat addTodoList(), removeTodoList(), updateTodoList(), searchTodoList(), dan sortTodoList() untuk menangani tindakan pengguna sesuai dengan menu yang dipilih.
- Menggunakan kelas InputUtil untuk menerima input dari pengguna.
- Menggunakan objek TodolistService untuk berinteraksi dengan logika bisnis yang terkait dengan to-do list.

6. App

100 / 100

Catatan

1. Isi: app.jar berisi bytecode dari semua file Kotlin yang Anda kompilasikan, termasuk kelas App, Todolist, TodolistRepository, TodolistRepositoryImpl, TodolistService, TodolistServiceImpl, InputUtil, dan TodolistView.

2. Fungsi: app.jar dapat dijalankan menggunakan perintah java -jar app.jar untuk menjalankan aplikasi yang terkandung di dalamnya.

3. Ketergantungan: Jika ada ketergantungan eksternal dalam proyek Anda, seperti perpustakaan pihak ketiga, ketergantungan tersebut tidak akan disertakan dalam app.jar. Anda perlu memastikan bahwa semua ketergantungan diperlukan tersedia saat menjalankan aplikasi.

4. Kesalahan Kompilasi: Jika ada kesalahan kompilasi selama proses, pesan kesalahan akan ditampilkan di konsol. Anda perlu memperbaiki kesalahan tersebut sebelum Anda dapat berhasil mengompilasi dan menjalankan aplikasi.

Code Auto Grader

78.03 / 80

1. Aplikasi Todolist

97.54 / 100

Test Case	Status	Score
TC1	PASS	20 / 20
TC2	PASS	20 / 20
TC3	PASS	20 / 20
TC4	PASS	20 / 20
TC5	PASS	20 / 20
Total Score		100 / 100