**LAPORAN RESMI**

**MODUL I**

**KOTLIN DASAR**

**PEMROGRAMAN BERGERAK**

**NAMA : MUHAMMAD RAFI RAHMATULLAH**

**N.R.P : 210441100084**

**DOSEN : ACHMAD ZAIN NUR, S.kom,M.T**

**ASISTEN : MUHAMMAD IQBAL FIRMANSYAH**

**TGL PRAKTIKUM : 22 MARET 2024**

**Disetujui :**

**Maret 2024**

**Asisten**

**MUHAMMAD IQBAL FIRMANSYAH**

**210441100084**

****

**LABORATORIUM BISNIS INTELIJEN SISTEM**

**PRODI SISTEM INFORMASI**

**JURUSAN TEKNIK INFORMATIKA**

**FAKULTAS TEKNIK**

**UNIVERSITAS TRUNOJOYO MADURA**

# BAB I

# PENDAHULUAN

## Latar Belakang

Kotlin adalah bahasa pemrograman modern yang dirancang untuk berjalan pada mesin virtual Java (JVM), serta dapat dikompilasi menjadi kode sumber JavaScript atau kode sumber native. Dikembangkan oleh JetBrains, Kotlin menawarkan sintaksis yang ekspresif, aman, dan mudah dipahami, sehingga membuatnya menjadi pilihan yang menarik untuk pengembangan berbagai jenis aplikasi, mulai dari pengembangan aplikasi Android hingga pengembangan aplikasi web.

Sebagai bahasa yang interoperabel dengan Java, Kotlin memiliki dukungan yang luas dari komunitas pengembang, serta memiliki integrasi yang mulus dengan perangkat lunak dan framework yang ada dalam ekosistem Java. Hal ini membuatnya menjadi pilihan yang kuat bagi pengembang yang ingin meningkatkan produktivitas mereka tanpa harus meninggalkan investasi yang sudah ada dalam ekosistem Java.

Selain itu, keamanan merupakan aspek yang diutamakan dalam desain Kotlin. Kotlin menawarkan fitur-fitur seperti null safety, tipe data yang aman, dan interoperabilitas yang kuat dengan Java, sehingga membantu pengembang dalam mencegah kesalahan umum yang terjadi dalam pengembangan perangkat lunak. Dengan demikian, Kotlin tidak hanya memberikan produktivitas yang tinggi bagi pengembang, tetapi juga membantu meningkatkan kualitas dan keamanan dari aplikasi yang dikembangkan. Dalam ekosistem yang semakin kompleks dan berubah-ubah, Kotlin menjanjikan kesinambungan dan keandalan dalam pengembangan perangkat lunak untuk masa depan.

## Tujuan

1. Mahasiswa mampu memahami konsep dasar kotlin.
2. Mahasiswa mampu menggunakan pengetahuan dasar kotlin untuk mengembangkan aplikasi sederhana.
3. Mahasiswa mampu beradaptasi dengan bahasa kotlin

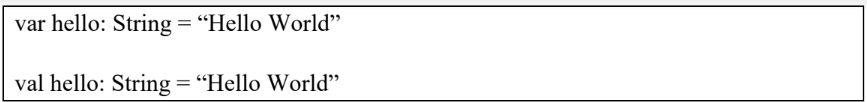
# BAB II

# DASAR TEORI

## Variabel dan Tipe Data

## C:\Users\syari\OneDrive\Pictures\Screenshots\Screenshot 2024-03-25 200910.png

Setelah kita liat gambar di atas, harusnya kita udah tau gasi apa itu variabel? Iya variable, tempat yang biasa kita gunakan untuk menyimpan berbagai macam tipe data dalam Bahasa Pemrograman, di Kotlin kita bisa mendeklarasikan sebuah variablesebagai berikut:



Anyway, ada yang notice ga penulisan var dan val di situ, kira-kira apa sih bedanya?

Setelah variabel, sekarang kita masuk ke tipe data yaa, sebelum masuk ke tipe data,coba liat gambar ini dan tebak kira-kira tipe data apa aja yang dipake di dalam gambar ini?



Kalo variabel adalah tempat yang kita gunakan untuk menyimpan berbagai macam tipedata, berarti tipe data adalah jenis data yang disimpen di dalam variabel, contohnya kaya gambar di bawah ini:



Gambar di atas kita sebut sebagai variabel, misalkan variabel lemari dan rakSepatu.Variable lemari punya tipe data pakaian dan rakSepatu punya tipe data sepatu.Jadi kalo disimpulkan tuh:

• Variabel : lemari dan rakSepatu

• Tipe Data : pakaian dan sepatu

Sampe sini paham kan apa bedanya variabel dan tipe data?

Anyway, di Kotlin kita bisa nulis berbagai macam tipe data sebagai berikut:



Kita bedah satu per satu kali, ya?

**Char**

Characters direpresentasikan menggunakan tipe Char. Tipe data Char hanya dapat kitagunakan untuk menyimpan karakter tunggal. Intinya sih, tipe data ini cuman bisa dipake buat nyimpen 1 karakter aja.

// define Char

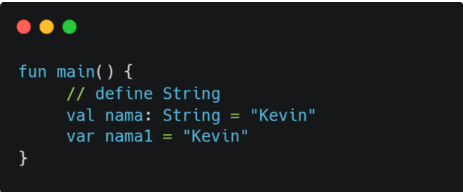
val hurufDepan: Char = 'A'

val hurufDepan1 = 'K'

**String**

String merupakan tipe data yang mirip dengan Char. Ia dapat digunakan untuk

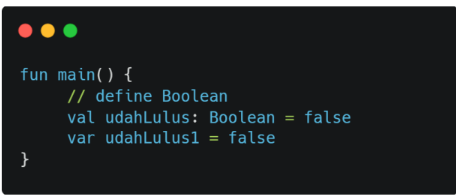
menyimpan nilai berupa teks. Perbedaannya, String bisa menampung beberapa karakter di dalamnya, jadi ga cuman 1 karakter aja kaya Char**.**

// define String

val nama: String = "Kevin"

val nama1 = "Kevin"

**Boolean**

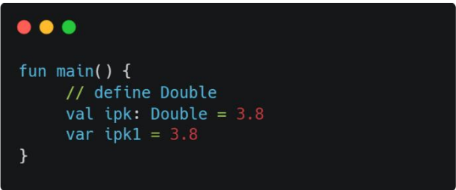
Boolean adalah sebuah tipe data yang hanya memiliki dua nilai, yaitu true dan false.

// define Boolean

val udahLulus: Boolean = false

val udahLulus1 = false

**Double**

Double adalah tipe data yang umumnya digunakan untuk menyimpan nilai desimal.Double digunakan untuk menyimpan nilai numerik pecahan sampai dengan maksimal 15-16 angka di belakang koma.

// define Double

val ipk: Double = 3.8

val ipk1 = 3.8

**Array**

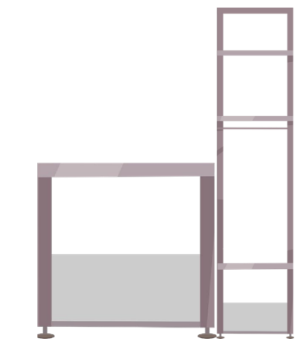
Array yakni tipe data yang memungkinkan kita untuk menyimpan beberapa objek di dalam sebuah variable.

// define Array

val matkulSmt7 = arrayOf("IMK",

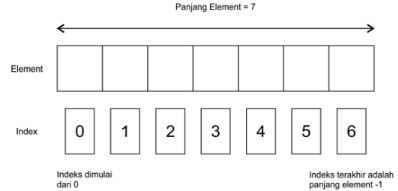
"Etika Profesi", "Skripsi")

By the way, array ini cukup unik karena berbeda dengan tipe data lain yang Cuma bisa menyimpan 1 jenis data, sedangkan array ini bisa menyimpan berbagai macam data dengan jumlah banyak, ilustrasinya sih gini:



Kalau kita mendeklarasikan variabel dengan tipe data int, kita itu kayak menyediakan lemari dengan 1 rak, kayak rak di atas ini. Tapi kalau kita mendeklarasikan variabel pakai tipe data array, kita tuh kayak beli lemari yang punya beberapa rak kayak gambar di atas ini. Jadi mending mana, array atau variabel biasa?

Terakhir, array juga memiliki urutan data atau yang biasa kita sebut sebagai Index. Index ini dimulai dari urutan 0, dan index terakhir berisi panjang elemen dikurangi satu. Jadi misal panjang elemennya 7, ya index terakhirnya 6.



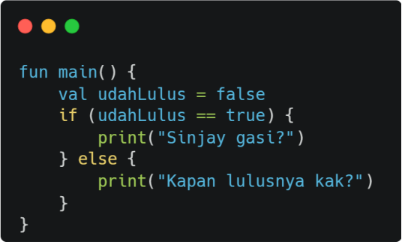
### **If Else (Penyeleksian Kondisi)**



Bayangin kamu mau nganterin nasi syukuran ke rumah tetangga yang jumlahnya 3,pasti kita bakal menentukan tetangga mana yang rumahnya bakal kita kunjungin duluan. Nah secara ga langsung temen-temen udah menggunakan konsep if else!

Jadi gampangnya sih, if else itu konsep pengambilan keputusan, di mana kita akan

menentukan mana yang akan kita taruh di urutan paling atas, hingga urutan paling

akhir. Nah untuk penulisan if else di kotlin tuh kaya gini:

val udahLulus = false

if (udahLulus == true) {

println("sinjay gasi?")

} else {

println("kapan lulus kak? hehe")

}

Anyway, ada versi lain dari if else, yaitu if – else if – else, bedanya apa sih kak dengan if else? Bedanya cuman di pengkondisiannya aja, jadi statement pengkondisiannya lebih banyak dari if else yang cuman 2 pilihan, contohnya kaya gini:

val semesterBerapa = 8

if (semesterBerapa == 6) {

println("jangan lupa kp kak")

} else if(semesterberapa == 7) {

println("kak udah ngambil skripsi?")

} else if (semesterBerapa == 8) {

println("kak gimana skripsinya?")

} else {

println("kapan lulus kak? hehe")

}

### **Looping**

Loop atau dalam bahasa Indonesia disebut putaran/perulangan adalah kondisi ketikakita akan menampilkan data yang banyak berulang kali. Bayangin ketika kita perlu mencetak kalimat 100x pada program. Bisa aja kita deklarasikan sampai 100x, tapi nanti jari kita yang jadi keriting :((

Kotlin punya 3 versi looping atau perulangan, yaitu:



Ada yang tau bedanya perulangan pertama, kedua dan ketiga (dibaca dari kiri ke

kanan)?

Perulangan pertama itu for loop, fungsinya untuk melakukan perulangan di mana kita udah tau pasti perulangannya bakal terjadi berapa kali, misalnya di kode tersebut tuh perulangannya terjadi 1 sampai 10.

Perulangan kedua itu while, fungsinya untuk melakukan perulangan di mana kita belum tau pasti perulangannya bakal terjadi berapa kali. Kaya bisa aja perulangan itu cuman terjadi 1x, tapi di sisi lain bisa aja perulangannya terjadi 1000x, semua tergantung dari kondisi yang dipake di dalam kurung while.

Perulangan ketiga itu do while, sama aja kaya while tapi bedanya dia ngelakuin perintah dulu baru ngecek kondisi yang ada di dalam kurung while. Sedangkan kalau while dia ngecek kondisi dulu baru ngejalanin perintah.

### **Function**

Function merupakan blok kode yang ditulis buat melakukan tugas tertentu. Pada tingkat yang lebih tinggi, suatu function, mengambil beberapa input tambahan yang disebut sebagai parameter, yang melakukan tugas tertentu pada input dan akhirnya bisa mengembalikan nilai.



Supaya familiar dengan Function, coba kamu bayangin sistem pencernaan manusia. Sistem pencernaan terdiri dari berbagai organ yang berpartisipasi dalam proses pencernaan. Setiap organ melakukan function spesifik, artinya mereka memiliki tugas yang berbeda-beda sesuai dengan kebutuhannya.

Anyway, blok function, yang merupakan bagian dari sistem, digunakan buat menangani tugas tertentu yang diperlukan untuk kelancaran function seluruh sistem. Function/method bisa memecah program menjadi sub-sub program, sehingga kita bisa bikin program lebih efisien.

Terus gimana kak cara buat function di Kotlin? Nah caranya gini...

fun main() {

biodata()

}

fun biodata() {

val nama = "Kevin"

val universitas = "Universitas

Trunojoyo"

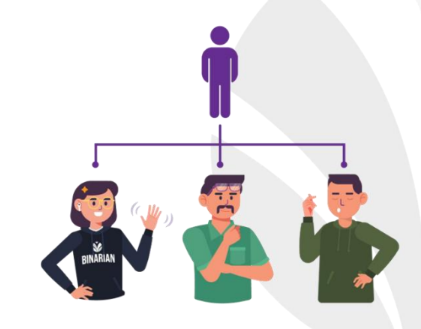
print(nama + " dari " +

universitas)

}

### **Class**

Manusia bisa kita ibaratkan sebagai sebuah CLASS. Manusia itu ada banyak karakter yang kita kenal. Misalnya, bestie kita ada Sabrina, Mas Gun, David.



Misal nih, dari 3 manusia itu, kita pilih sabrina untuk kita buatkan biodatanya. Nama dia Sabrina, perempuan dengan tinggi badan 160 cm dan berat badan 54 kg. Kalau di Kotlin, nama, jenis kelamin, tinggi badan, dan berat badan Sabrina ini

yang kita sebut sebagai PROPERTY. Dalam aktivitas kesehariannya, Sabrina ini demen banget jogging dan berenang. Nah, berbagai aktivitas yang dilakukan

sama Sabrina ini disebut sebagai Method/Function.

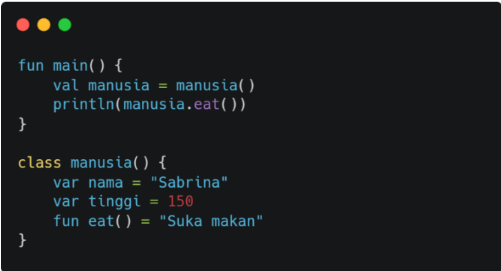
Jadi, apa itu class?

Class adalah blueprint (cetak biru/tempat) yang bisa kita gunakan untuk membuat

sebuah objek. Contohnya kaya di atas, kita bikin objek yang bernama sabrina. Nah di dalam class itu ada yang namanya property, kelas ini bisa menentukan property dan method apa aja yang dimiliki oleh sebuah class. Contoh dari property dan metode sudah dijelaskan di atas.

Nah sekarang kita bakal bahas cara mendeklarasikan class di dalam Kotlin.

fun main() {

val manusia = manusia()

println(manusia.eat())

}

class manusia() {

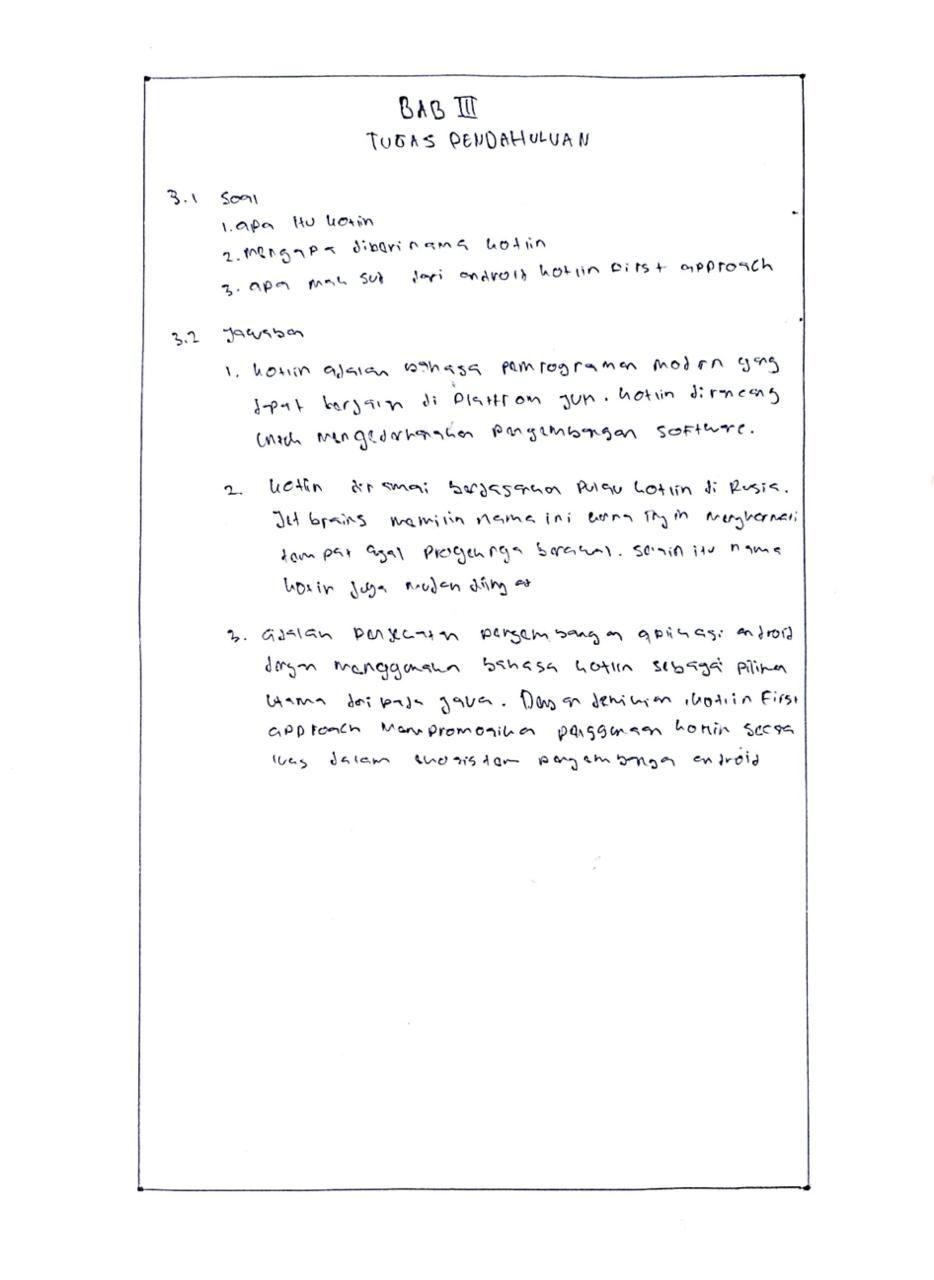
var nama = "Sabrina"

var tinggi = 158

fun eat() = "Suka makan"

}

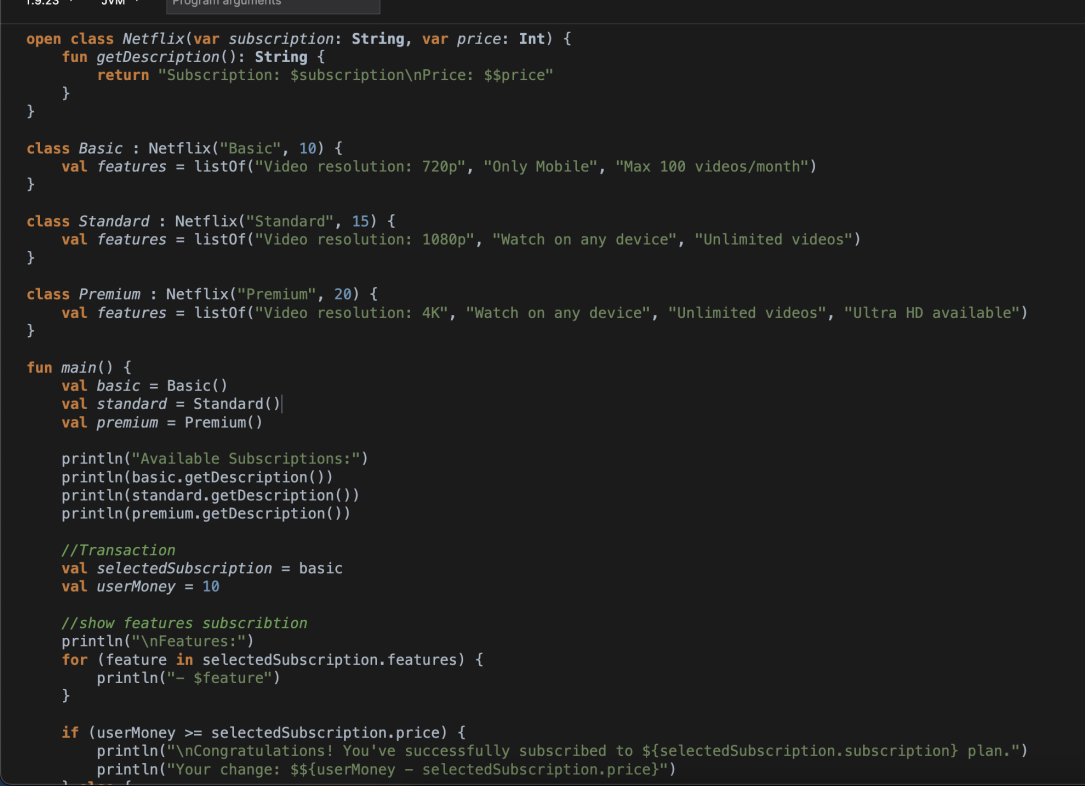
Di sini kita mendeklarasikan class manusia, class manusia memiliki property nama dan tinggi, selain itu dia juga memiliki method eat yang bisa kita panggil jika kita butuhkan.



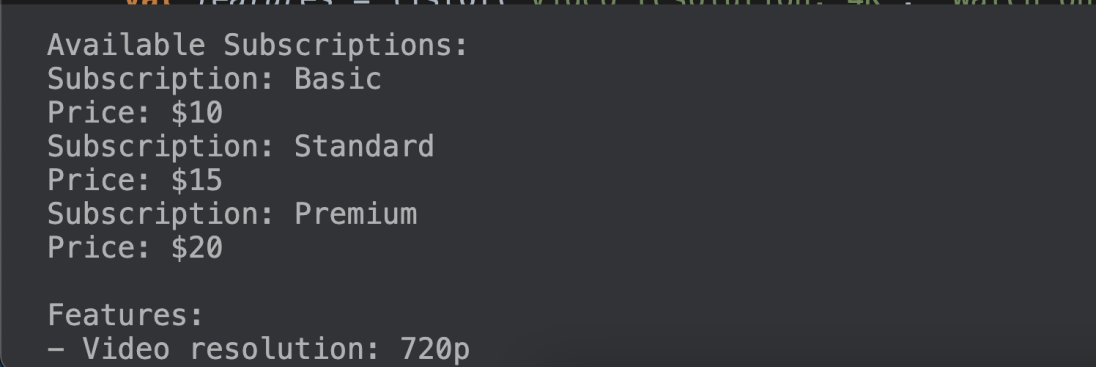
# BAB IV

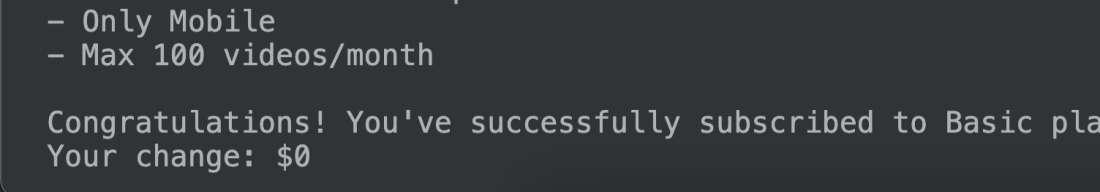
# IMPLEMENTASI

## Source Code

****

## Hasil

****

****

## Penjelasan Code

Kode tersebut adalah program sederhana dalam bahasa Kotlin yang mensimulasikan berlangganan layanan Netflix dengan berbagai jenis paket langganan. Berikut adalah penjelasan singkat tentang struktur dan fungsionalitas kode:

1. Kelas Netflix (Netflix class):

* Kelas ini merupakan kelas terbuka (open class) yang memiliki properti subscription (langganan) dan price (harga).
* Memiliki fungsi `getDescription()` yang mengembalikan deskripsi langganan dan harga dalam format string.

2. Kelas-kelas turunan (Subclasses):

* `Basic`, `Standard`, dan `Premium` adalah kelas turunan dari kelas `Netflix`.
* Setiap kelas turunan menginisialisasi properti subscription dan price dengan nilai yang sesuai.
* Setiap kelas turunan memiliki properti `features` yang berisi daftar fitur-fitur yang disertakan dalam paket langganan tersebut.

3. Fungsi `main():

* Pada bagian ini, dilakukan inisialisasi tiga objek: `basic`, `standard`, dan `premium` dengan masing-masing merupakan instance dari kelas `Basic`, `Standard`, dan `Premium`.
* Kemudian, deskripsi dari setiap langganan ditampilkan menggunakan fungsi `getDescription()`.
* Dilakukan simulasi transaksi dimana user memilih paket langganan (`selectedSubscription`) dan memiliki uang sejumlah `userMoney`.
* Fitur-fitur dari paket langganan yang dipilih ditampilkan.
* Kemudian dilakukan pengecekan apakah uang yang dimiliki user cukup untuk berlangganan paket tersebut. Jika cukup, program memberikan konfirmasi berlangganan beserta kembalian uangnya. Jika tidak cukup, program memberikan pesan bahwa uangnya tidak mencukupi.

Jadi, secara keseluruhan, kode tersebut adalah simulasi sederhana dari proses berlangganan di Netflix, di mana pengguna memilih paket langganan, melihat fitur-fiturnya, dan melakukan pembelian berdasarkan uang yang dimilikinya.

# BAB V

# PENUTUP

## Analisa

Dari hasil praktikum, praktikan menganalisa bahwa jframe ini sangat memudahkan dalam pembuatan suatu program dan dapat mendesain tampilan sesuai dengan keinginan. Dalam pembuatan jframe ini menggunakan basis swig. Menggunakan bahasa pemograman java dan aplikasi netbeans. Swing GUI builder ini membantu programmer untuk membangun aplikasi dekstop secara visual dan bukan hanya sekedar text-based code. Dengan melakukan drag-and-drop komponen swing ke top level containernya, sebuah aplikasi GUI sudah dapat dibangun. Didalam jframe ini , banyak komponen yang tersedia untuk memudahkan pengguna membuat desain seperti, frame, label, button, combo box, table, radio button, button group dan menu.

hasil praktikum ini memberikan wawasan penting tentang penggunaan komponen dasar GUI dalam pengembangan aplikasi. Praktikan telah belajar bagaimana menciptakan antarmuka pengguna yang ramah pengguna, efisien, dan konsisten, serta bagaimana memisahkan tampilan dari logika aplikasi. Dengan pemahaman ini, praktikan dapat lebih efektif dalam merancang dan mengembangkan aplikasi dengan antarmuka pengguna yang menarik dan berfungsi dengan baik.

1. **Kesimpulan**

Kesimpulan dari latihan di atas adalah bahwa kode tersebut adalah implementasi sederhana dari simulasi proses berlangganan layanan Netflix menggunakan bahasa pemrograman Kotlin. Dalam kode tersebut, terdapat hierarki kelas dengan kelas induk `Netflix` dan tiga kelas turunannya (`Basic`, `Standard`, dan `Premium`), masing-masing mewakili jenis paket langganan yang berbeda.

Setiap kelas turunan memiliki properti unik yang mewakili fitur-fitur dari paket langganan tersebut. Selain itu, terdapat fungsi `getDescription()` yang mengembalikan deskripsi lengkap dari langganan beserta harganya.

Melalui fungsi `main()`, pengguna dapat memilih paket langganan, melihat fitur-fiturnya, dan melakukan simulasi transaksi berlangganan. Program memberikan feedback berdasarkan uang yang dimiliki pengguna, apakah cukup untuk berlangganan atau tidak.

Dengan demikian, kode tersebut memberikan pemahaman yang jelas tentang bagaimana konsep pewarisan (inheritance) dan penggunaan kelas dalam Kotlin, serta memberikan gambaran tentang bagaimana proses pengguna dapat berinteraksi dengan aplikasi melalui tindakan-tindakan seperti memilih paket dan melakukan transaksi.