LAPORAN PRAKTIKUM
PEMROGRAMAN BERBASIS MOBILE
PERTEMUAN KE-2

Disusun Oleh :

NAMA	**: Raden Isnawan Argi Aryasatya**
NIM	**: 195410257**
JURUSAN	**: Informatika**
JENJANG	**: S1**
KELAS	**: 5**

Laboratorium Terpadu

Sekolah Tinggi Manajemen Informatika Komputer

AKAKOM

YOGYAKARTA

2021

PERTEMUAN KE-2
(DASAR PEMROGRAMAN KOTLIN, FUNGSI DAN KELAS)

TUJUAN

1. Mahasiswa mampu mengimplementasikan dasar-dasar pemrograman dengan Kotlin
2. Mahasiswa mampu mengimplementasikan fungsi dan dipanggil dalam program

DASAR TEORI

Fungsi adalah blok pernyataan terkait yang bersama-sama melakukan tugas tertentu. Sebagai contoh katakanlah kita harus menulis tiga baris kode untuk menghasilkan rata-rata dua angka, jika kita membuat fungsi untuk menghasilkan rata-rata maka kita tidak perlu menulis tiga baris itu lagi dan lagi, kita bisa memanggil fungsi yang kita buat.
Ada dua jenis fungsi di Kotlin:

1. Fungsi pustaka standar
2. Fungsi yang didefinisikan pengguna

Sementara kelas adalah blok bangunan utama dari setiap bahasa pemrograman berorientasi objek. Semua objek adalah bagian dari kelas dan berbagi properti umum dan perilaku yang didefinisikan oleh kelas dalam bentuk data anggota dan fungsi anggota masing-masing. Kelas didefinisikan menggunakan kata kunci class di Kotlin.

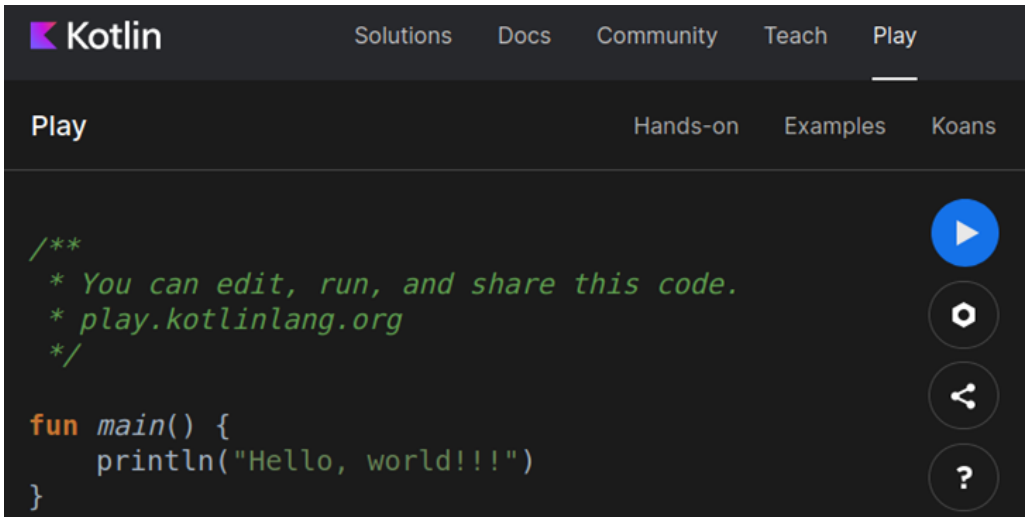
Lalu ada Konstruktor. Tujuan utama konstruktor adalah menginisialisasi properti kelas. Konstruktor dipanggil ketika kita membuat objek kelas.
Tipe Konstruktor:

1. Konstruktor Utama - Menginisialisasi properti kelas
2. Konstruktor Sekunder - Menginisialisasi properti kelas, kita dapat memiliki kode inisialisasi tambahan di dalam konstruktor sekunder.

Terakhir ada kelas data. Di Kotlin, Anda bisa membuat kelas data untuk menyimpan data. Alasan mengapa Anda ingin menandai kelas sebagai data adalah untuk memberi tahu kompiler bahwa Anda membuat kelas ini untuk menyimpan data, kompiler kemudian membuat beberapa fungsi secara otomatis untuk kelas data Anda yang akan sangat membantu dalam mengelola data.

PRAKTIK

1. Anda akan menggunakan compiler Kotlin secara online. Lewat browser buka url: play.kotlinlang.org dengan tampilan seperti berikut:



2. Modifikasilah kode yang ditampilkan sehingga menjadi seperti berikut.

```
fun main(args : Array<String>) {
    var num = 16
    println("Akar $num adalah: ${Math.sqrt(num.toDouble())}")
    val a = 12
    val b=15
    println("Nilai terbesar dar $a dan $b adalah: "+Math.max(a, b))
    println("4 pangkat 3 = ${Math.pow(4.0, 3.0)}")
}
```

Penjelasan: di nomor 3

3. Jalankan program yang ada dengan klik tombol lingkaran biru di sebelah kanan atas. Perhatikan hasil running program di bagian bawah jendela browser.

```
Akar 16 adalah: 4.0
Nilai terbesar dar 12 dan 15 adalah: 15
4 pangkat 3 = 64.0
```

Penjelasan:
Dengan variable num yang bernilai 16, kita menghitung akar dari 16 dengan fungsi ***Math.sqrt*** dan tipe datanya kita gunakan double dengan ***num.toDouble*** yang kemudian menghasilkan nilai 4.0. Kemudian kita deklarasikan val a yang bernilai 12 dan val b yang bernilai 15. Dengan kedua nilai dari kedua variable tersebut, kita tentukan mana nilai yang terbesar dengan fungsi ***Math.max(a, b)*** yang menghasilkan output 15. Terakhir, kita hitung 4 pangkat 3 dengan fungsi ***Math.pow*** yang menghasilkan output 64.0.

4. Buat program seperti berikut, dan perhatikan hasil running program.

```
//Created the function
fun jumlah(bilangan2: Array<Int>): Int
{
    var jml = 0
    for(bil in bilangan2){
        jml += bil
}
    return jml
}
fun main(args : Array<String>){
    val arrBil = arrayOf(10, 20, 30, 50)
    println("Jumlah bilangan: ${jumlah(arrBil)}")
}
```

```
Jumlah bilangan: 110
```

Penjelasan:
Pertama, kita buat fungsi bernama jumlah dengan menuliskan kode fun ***jumlah(bilangan2: Array<Int>): Int***. Di dalamnya, kita buat variable bernama jml yang kita beri nilai 0. Lalu kita lakukan perulangan dengan for untuk mengakses dan mengulang bil di bilangan2 (bil adalah seluruh data di bilangan 2) dengan kode for(bil in bilangan2). Kemudian kita ***tulis jml += bil*** yang artinya seluruh nilai di dalam bilangan2 akan ditambah. Lalu kita kembalikan nilai jml dengan return jml. Selanjutnya kita buat main function yang di dalamnya kita buat ***val arrBil = arrayOf(10, 20, 30, 50)***. Untuk menampilkan jumlah bilangan, kita cetak ***println("Jumlah bilangan: ${jumlah(arrBil)}")***.

5. Modifikasilah program di atas sehingga argumen/parameter fungsi bersifat variabel, seperti berikut. Perhatikan hasil program.

```
fun jumlah(vararg bil2: Int): Int
{
    var jml = 0
    bil2.forEach {bil ->
    jml += bil
}
    return jml
}
fun main(args : Array<String>){
    println("Jumlah bilangan: ${jumlah(10, 20, 30, 40)}")
}
```

```
Jumlah bilangan: 100
```

Penjelasan:
Di program ini kita membuat fungsi penjumlahan yang lebih efisien dan simple. Seperti tadi, kita buat variable bernama jml yang kita beri nilai 0 yaitu dengan menuliskan ***var jml = 0***. Perulangan yang kita gunakan sekarang adalah forEach. Untuk mengakses nilai pada parameter bil2, kita gunakan baris kode ***bil2.forEach***. lalu dengan variable bil kita bisa dapatkan seluruh data untuk kemudian kita proses perulangannya dengan metode penjumlahan yaitu dengan baris kode ***bil -> jml += bil***. Kemdudian kita ***return jml***. Selanjutnya kita buat main function yang di dalamnya langsung kita panggil fungsi jumlah beserta nilai-nilai parameter bil2 yaitu dengan menuliskan baris kode ***println("Jumlah bilangan: ${jumlah(10, 20, 30, 40)}")***

6. Tambahkan jumlah parameter fungsi jumlah ini pada saat dipanggil, menjadi: jumlah(10, 20, 30, 40, 50, 60). Perhatikan hasil program.

```
fun jumlah(vararg bil2: Int): Int
{
    var jml = 0
    bil2.forEach {bil ->
    jml += bil
}
    return jml
}
fun main(args : Array<String>){
    println("Jumlah bilangan: ${jumlah(10, 20, 30, 40, 50, 60)}")
}
```

```
Jumlah bilangan: 210
```

Penjelasan: yang terjadi adalah muncul hasil penjumlahan 10, 20, 30, 40, 50, 60 yaitu 210.

7. Buat program seperti berikut yang menggunakan fungsi inline: jumlah, dan perhatikan hasil running program.

```
fun main(args : Array<String>){
    val jumlah = {bil1: Int, bil2: Int -> bil1 + bil2}
    println("6 + 4 = ${jumlah(6,4)}")
}
```

```
6 + 4 = 10
```

Penjelasan:
Fungsi Inline (disebut juga fungsi lambda) dapat didefinisikan di dalam fungsi main(). Dalam program di atas, kita telah mendefinisikan fungsi inline jumlah() yang menerima dua argumen integer bil1 dan bil2 yang ditulis dengan kode ***val jumlah = {bil1: Int, bil2: Int -> bil1 + bil2}*** dan tipe hasil adalah integer yang ditampilkan dengan baris kode ***println("6 + 4 = ${jumlah(6,4)}")***.

8. Modifikasilah program di atas yang digunakan untuk mengalikan 3 buah bilangan.

```
fun main(args : Array<String>){
    val kali = {bil1: Int, bil2: Int, bil3: Int -> bil1 * bil2 * bil3}
    println("2 x 3 x 4 = ${kali(2,3,4)}")
}
```

```
2 x 3 x 4 = 24
```

Penjelasan:
kita mendefinisikan fungsi inline kali() yang menerima tiga argumen integer bil1,bil2, dan bil3 yang ditulis dengan kode ***val kali = {bil1: Int, bil2: Int, bil3: Int -> bil1 * bil2 * bil3}*** dan tipe hasil adalah integer yang ditampilkan dengan baris kode ***println("2 x 3 x 4 = ${kali(2,3,4)}")***

9. Jalankan program berikut dalam fungsi main(), perhatikan apa yang terjadi.

```
fun main(){
    val upperCase1: (String) -> String = { str: String -> str.toUpperCase() }
    val upperCase2: (String) -> String = { str -> str.toUpperCase() }
    val upperCase3 = { str: String -> str.toUpperCase() }
    val upperCase4: (String) -> String = { it.toUpperCase() }
    val upperCase5: (String) -> String = String::toUpperCase
    println(upperCase1("hello"))
    println(upperCase2("hello"))
    println(upperCase3("hello"))
    println(upperCase4("hello"))
    println(upperCase5("hello"))
}
```

```
HELLO
HELLO
HELLO
HELLO
HELLO
```

Penjelasan:
Di program ini, kita menggunakan fungsi ***toUpperCase()*** yang digunakan untuk mengubah nilai string ke nilai string yang terdiri dari huruf besar semua. Jadi, walaupun kita menulis kode untuk mencetak kata “hello”, yang muncul di output program adalah kata “HELLO” karena semua huruf kecil sudah dirubah menjadi huruf besar dengan fungsi ***toUpperCase()***.

10. Buat program seperti berikut, dan perhatikan hasil running program.

(program di halaman selanjutnya)

```
fun main(args: Array<String>) {
    func("BeginnersBook", ::demo)
}
    fun func(str: String, myfunc: (String) -> Unit) {
    print("Welcome to Kotlin tutorial at ")
    myfunc(str)
}
    fun demo(str: String) {
    println(str)
}
```

```
Welcome to Kotlin tutorial at BeginnersBook
```

Penjelasan:
Fungsi orde tinggi (higher-order function) dapat memiliki fungsi lain sebagai parameter atau mengembalikan fungsi atau dapat melakukan keduanya. Di dalam program ini, kita belajar bagaimana kita melewatkan suatu fungsi ke fungsi lain. Kita juga akan melihat bagaimana suatu fungsi mengembalikan fungsi lainnya. Dalam contoh program di atas, kita melewatkan fungsi demo() ke fungsi func lainnya (). Untuk meneruskan fungsi sebagai parameter ke fungsi lain, kita gunakan operator :: di depan fungsi seperti yang ditunjukkan pada baris kode ***func("BeginnersBook", ::demo).*** Di fungsi selanjutnya yang bernama func, kita beri perintah untuk menampilkan ***print("Welcome to Kotlin tutorial at ")***. Di function demo, saat kita melakukan ***println(str)***, yang tertampilkan di output program adalah gabungan dari kedua string dari kedua fungsi yang kita buat tadi.

11. Buat program seperti berikut, dan perhatikan hasil running program.

```
fun main(args: Array<String>) {
    val sum = func(10)
    println("10 + 20: ${sum(20)}")
}

fun func(num: Int): (Int) -> Int = {num2 -> num2 + num}
```

```
10 + 20: 30
```

Penjelasan:
Di dalam fungsi main, kita buat variable yaitu ***val sum = func(10)*** kemudian kita tuliskan ***println("10 + 20: ${sum(20)}")*** untuk menampilkan hasil penjumlahan. Kemudian kita buat fungsi baru bernama func dengan menuliskan fun ***func(num: Int): (Int) -> Int = {num2 -> num2 + num}***. Lalu kenapa hasilnya 30? Karena variable sum mempunyai nilai 10 yang kemudian ditambahkan dengan sum(20).

12. Buat program seperti berikut, dan perhatikan hasil running program.

```
fun main(args: Array<String>) {

    //creating the object of class Student
    val mhs = Mahasiswa("Susi Susanti", 23)
```

```
    println("Nama : ${mhs.nama}")
    println("Umur : ${mhs.umur}")
}

class Mahasiswa(val nama: String, var umur: Int) {
//This is my class. For now I am leaving it empty
}
```

```
Nama : Susi Susanti
Umur : 23
```

Penjelasan:
Di program ini kita membuat objek mhs milik kelas Mahasiswa yang kita tulis dengan kode ***val mhs = Mahasiswa("Susi Susanti", 23)***. Untuk menampilkan nilai dari parameter milik objek tersebut, kita tulis ***println("Nama : ${mhs.nama}")*** dan ***println("Umur : ${mhs.umur}")***. Selanjutnya kita mendeklarasikan konstruktor ***(val nama: String, var umur: Int)*** sebagai bagian dari header kelas. Ini adalah konstruktor utama kita yang menginisialisasi properti nama dan umur (anggota data) dari kelas Mahasiswa.

13. Tambahkan objek Mahasiswa lagi dengan nama mhs2 dan berilah data nama dan umur, kemudian tampilkan.

```
fun main(args: Array<String>) {

    //creating the object of class Student
    val mhs = Mahasiswa("Susi Susanti", 23)
    val mhs2 = Mahasiswa("Raden Isnawan", 21)

    println("Nama : ${mhs.nama}")
    println("Umur : ${mhs.umur}")
    println("")
    println("Nama : ${mhs2.nama}")
    println("Umur : ${mhs2.umur}")
}

class Mahasiswa(val nama: String, var umur: Int) {
//This is my class. For now I am leaving it empty
}
```

```
Nama : Susi Susanti
Umur : 23

Nama : Raden Isnawan
Umur : 21
```

Penjelasan:
objek kedua adalah mhs2 yang kita tulis dengan ***val mhs2 = Mahasiswa("Raden Isnawan", 21)*** dan kita tampilkan pada output program dengan kode ***println("Nama : ${mhs2.nama}") dan println("Umur : ${mhs2.umur}"***

14. Tambahkan nilai default pada konstruktor. Kemudian buatlah objek Mahasiswa (misal mhs3) tanpa data nama dan umur, dan tampilkan hasilnya.

```
fun main(args: Array<String>) {

    //creating the object of class Student
    val mhs = Mahasiswa("Susi Susanti", 23)
    val mhs2 = Mahasiswa("Raden Isnawan", 21)
    val mhs3 = Mahasiswa()

    println("Nama : ${mhs.nama}")
    println("Umur : ${mhs.umur}")
    println("")
    println("Nama : ${mhs2.nama}")
    println("Umur : ${mhs2.umur}")
    println("")
    println("Nama : ${mhs3.nama}")
    println("Umur : ${mhs3.umur}")
}
class Mahasiswa(val nama: String = "Mahasiswa", var umur: Int = 20) {
//This is my class. For now I am leaving it empty
}
```

```
Nama : Susi Susanti
Umur : 23

Nama : Raden Isnawan
Umur : 21

Nama : Mahasiswa
Umur : 20
```

Penjelasan:
Di program ini, kita memanfaatkan nilai default pada parameter konstruktor class Mahasiswa. Variable nama kita beri nilai "Mahasiswa" dan variable umur kita beri nilai 20 yang dilakukan dengan menuliskan ***class Mahasiswa(val nama: String = "Mahasiswa", var umur: Int = 20)***. Dengan begitu, saat kita membuat sebuah objek tanpa nilai parameter, maka otomatis objek tersebut akan menampung dan menampilkan nilai default yang ada di konstruktor. Objek yang kita buat supaya bisa menampung nilai default konstruktor adalah ***val mhs3 = Mahasiswa()***. Dan untuk menampilkan output nilai objek tersebut kita tulis kode ***println("Nama : ${mhs3.nama}")*** dan ***println("Umur : ${mhs3.umur}")***.

15. Buat program seperti berikut, dan perhatikan hasil running program.

```
fun main(args: Array<String>) {
    val mhs = Mahasiswa("Susi Susanti", 23)
    val mhs2 = Mahasiswa("Haryanto", 22)
    val mhs3 = Mahasiswa()
}
class Mahasiswa(val nama: String = "Mahasiswa", var umur: Int = 99) {
val namaMhs: String
var umurMhs: Int
init{
    if(nama == "Mahasiswa"){
        namaMhs = "Kosong"
```

```
        umurMhs = 0
    } else {
        namaMhs = nama.toUpperCase()
        umurMhs = umur
    }
        println("Nama : $namaMhs")
        println("Umur : $umurMhs")
    }
}
```

```
Nama : SUSI SUSANTI
Umur : 23
Nama : HARYANTO
Umur : 22
Nama : Kosong
Umur : 0
```

Penjelasan:
Di dalam program di atas, kita menambahkan kode initializer tambahan di dalam konstruktor suatu kelas. Blok initializer kita tulis di dalam konstruktor menggunakan init. Dalam blok ini kita memiliki logika inisialisasi tambahan. Di bawah konstrukor ***class Mahasiswa(val nama: String = "Mahasiswa", var umur: Int = 99)***, kita deklarasikan dua variable yaitu val namaMhs: String dan var umurMhs: Int. lalu kita awali initializer dengan keyword ***init***. Di dalamnya kita buat seleksi kondisi yaitu ***if(nama == "Mahasiswa")*** yang artinya "jika nilai dari nama adalah 'Mahasiswa', maka.." lalu program menampilkan output namaMhs berupa kata "Kosong" dan umurMhs bernilai 0. Dengan kata lain, objek mhs3 memiliki nama "Kosong" dan umur 0. Untuk objek mhs dan mhs2 diseleksi dengan kondisi else yaitu ***namaMhs = nama.toUpperCase()*** yang mengubah nama menjadi huruf besar semua dan ***umurMhs = umur*** untuk menampilkan umur yang tadi dimasukkan saat pembuatan objek. Terakhir, untuk menampilkan output kita tulis ***println("Nama : $namaMhs")*** dan ***println("Umur : $umurMhs")***

16. Buat program seperti berikut, dan perhatikan hasil running program.

```
data class Mahasiswa(val nama: String, val umur: Int)
fun main(args: Array<String>) {
    val mhs = Mahasiswa("Susi Susanti", 23)
    val mhs2 = Mahasiswa("Agus Budianto", 25)
    println("Nama Mahasiswa: ${mhs.nama}")
    println("Umur Mahasisa: ${mhs.umur}")
    println("Nama Mahasiswa: ${mhs2.nama}")
    println("Umur Mahasisa: ${mhs2.umur}") }
```

```
Nama Mahasiswa: Susi Susanti
Umur Mahasisa: 23
Nama Mahasiswa: Agus Budianto
Umur Mahasisa: 25
```

Penjelasan:
Program ini merupakan penerapan data class. Data class merupakan class khusus yang digunakan untuk menampung data. Pada kotlin data class ditandai dengan keyword data yaitu ***data class Mahasiswa(val nama: String, val umur: Int)***. Untuk membuat objek dan mengubah property kita tuliskan baris kode ***val mhs = Mahasiswa("Susi Susanti", 23)*** dan ***val mhs2 = Mahasiswa("Agus Budianto", 25)***. Terakhir kita lakukan destructuring yang digunakan untuk mengambil value dari properti dari class dan dipecah menjadi variabel

tersendiri yaitu dengan ***println("Nama Mahasiswa: ${mhs.nama}")*** dan ***println("Umur Mahasisa: ${mhs.umur}")***. Hal tersebut juga berlaku untuk mhs2.

17. Modifikasilah program di atas sehingga menjadi seperti berikut, dan perhatikan hasil running program.

```
data class Mahasiswa(val nama: String, val umur: Int)
fun main(args: Array<String>) {
    val mhs = Mahasiswa("Susi Susanti", 23)
    val mhs2 = Mahasiswa("Susi Susanti", 23)
    val mhs3 = Mahasiswa("Agus Budianto", 25)

    if (mhs.equals(mhs2) == true)
        println("mhs sama dengan mhs2.")
    else
        println("mhs tidak sama dengan mhs2.")
    if (mhs.equals(mhs3) == true)
        println("mhs sama dengan mhs3.")
    else
        println("mhs tidak sama dengan mhs3.")

println("Hashcode dari mhs: ${mhs.hashCode()}")
println("Hashcode dari mhs2: ${mhs2.hashCode()}")
println("Hashcode dari mhs3: ${mhs3.hashCode()}")
}
```

```
mhs sama dengan mhs2.
mhs tidak sama dengan mhs3.
Hashcode dari mhs: 454548524
Hashcode dari mhs2: 454548524
Hashcode dari mhs3: 1448582333
```

Penjelasan:
Di dalam data class tersebut, kita beri seleksi kondisi. Yang pertama adalah ***if (mhs.equals(mhs2) == true)*** yang artinya "jika nilai mhs sama dengan nilai mhs2, maka…" lalu jika terpenuhi maka program menampilkan ***println("mhs sama dengan mhs2.")***. Jika tidak terpenuhi maka program menampilkan pesan ***println("mhs tidak sama dengan mhs2.")***. Lalu kita buat kondisi if kedua yaitu adalah ***if (mhs.equals(mhs3) == true)*** yang artinya "jika nilai mhs sama dengan nilai mhs3, maka…" lalu jika terpenuhi maka program menampilkan ***println("mhs sama dengan mhs3.")***. Jika tidak terpenuhi maka program menampilkan pesan ***println("mhs tidak sama dengan mhs3.")***. terakhir, kita tampilkan hashcode dari setiap objek dengan fungsi hashCode().

18. Buat program seperti berikut, dan perhatikan hasil running program.

```
data class Mahasiswa(val nama: String, val umur: Int)
fun main(args: Array<String>) {
    val mhs = Mahasiswa("Susi Susanti", 23)
    // mengkopi umur dari objek mhs
    val mhs2 = mhs.copy(nama = "Lusiana")
println("Nama ${mhs.nama}, Umur ${mhs.umur}")
println("Nama ${mhs2.nama}, Umur ${mhs2.umur}")
```

```
}
```

```
Nama Susi Susanti, Umur 23
Nama Lusiana, Umur 23
```

Penjelasan:
Di program ini, kita menggunakan fungsi copy untuk menyalin nilai dari suatu objek. Objek pertama kita buat dengan menuliskan ***val mhs = Mahasiswa("Susi Susanti", 23)***. Kemudian objek kedua kita buat dengan menuliskan ***val mhs2 = mhs.copy(nama = "Lusiana")***. Fungsi copy() tersebut berguna untuk menyalin semua nilai milik mhs. Tetapi, nilai nama pada mhs2 kita ubah menjadi "Lusiana" dengan baris kode ***nama = "Lusiana"***.

19. Tambahkan program di atas dengan kode berikut pada bagian paling bawah.

```
data class Mahasiswa(val nama: String, val umur: Int)
fun main(args: Array<String>) {

    val mhs = Mahasiswa("Susi Susanti", 23)
    // mengkopi umur dari objek mhs
    val mhs2 = mhs.copy(nama = "Lusiana")

println("Nama ${mhs.nama}, Umur ${mhs.umur}")
println("Nama ${mhs2.nama}, Umur ${mhs2.umur}")

val nama = mhs.component1()
val umur = mhs.component2()
println("Nama $nama, Umur $umur")
}
```

```
Nama Susi Susanti, Umur 23
Nama Lusiana, Umur 23
Nama Susi Susanti, Umur 23
```

Penjelasan:
Di program ini kita menambahkan beberapa baris kode untuk mengambil nilai nama dan umur dari objek mhs yaitu:
val nama = mhs.component1()
val umur = mhs.component2()

LATIHAN

1. Tulis fungsi untuk menghitung jarak dua buah titik t1(x1, y1) dan t2(x2, y2). Panggilah fungsi ini dalam fungsi main() dengan t1(2,3) dan t2(8,7), serta t1(5,3) dan t2(-8, -4).

```
fun jarak(x1:Int, y1:Int, x2:Int, y2:Int): Double
{
    var hasil = Math.sqrt((x2-x1)*(x2-x1)*1.0 + (y2-y1)*(y2-y1))
    return hasil
}
fun main(args : Array<String>){
    println("Jarak dari t1(2,3) dan t2(8,7): ${jarak(2, 3, 8, 7)}")
    println("Jarak dari t1(5,3) dan t2(-8,-4): ${jarak(5, 3, -8, -4)}")
```

```
}
```

```
Jarak dari t1(2,3) dan t2(8,7): 7.211102550927978
Jarak dari t1(5,3) dan t2(-8,-4): 14.7648230602334
```

Penjelasan:
Pada program di atas, kita menggunakan rumus jarak yang bisa kita dapatkan dari teorema Pythagoras. Rumus yang kita gunakan adalah seperti gambar di bawah berikut:

$$\sqrt{(x_2-x_1)^2+(y_2-y_1)^2}$$

Artinya, jika x1, y1=2, 3 dan x2, y2=8, 7 maka $(x2-x1)^2=(8-2)^2 = 36$ dan $(y2-y1)^2= (7-3)^2 = 16$. sekarang 36 + 16 =52, dan 52 adalah akar kuadrat dari 7.2. Jadi jarak antara (2,3) dan (8,7) adalah 7.211102550927978.

jika x1, y1=5, 3 dan x2, y2=-8, -4 maka $(x2-x1)^2=(-8-5)^2 = 169$ dan $(y2-y1)^2= (-4-3)^2 = 49$. sekarang 169 + 49 = 218, dan 218 adalah akar kuadrat dari 14.7. Jadi jarak antara titik (5,3) dan (-8,-4) adalah 14.7648230602.

2. Buatlah fungsi inline dengan nama pangkat, dengan contoh pemanggilan seperti berikut.

```
fun main(args : Array<String>){
    val pangkat = {bil1: Int, bil2: Int -> Math.pow(bil1*1.00,bil2*1.00)}
    println("4 pangkat 3 = ${pangkat(4,3)}")
}
```

```
4 pangkat 3 = 64.0
```

Penjelasan:
Di program ini, kita menggunakan salah satu fungsi pustaka standar di Kotlin yaitu Math.pow() yang berguna untuk menghitung operasi pangkat. Untuk melakukan hal tersebut, kita deklarasikan variabel bernama pangkat, kita tentukan tipe data nya, dan kita terapkan Math.pow() pada parameter milik variabel pangkat. Hal tersebut bisa kita wujudkan dengan menuliskan ***val pangkat = {bil1: Int, bil2: Int -> Math.pow(bil1*1.00,bil2*1.00)}***. Terakhir, kita tampilkan hasil dengan menuliskan ***println("4 pangkat 3 = ${pangkat(4,3)}")***.

3. Buat kelas Barang dengan properti: nama, harga, jumlah, dan diskon. Berilah nilai default pada konstruktornya. Tambahkan fungsi(method):
   - tampil() untuk menampilkan semua propertinya
   - hitungTotal() untuk menghitung harga x jumlah x diskon

Buatlah fungsi main() untuk mengakses kelas Barang

```
fun tampil(){
    val properti = Barang()
    println("Nama : ${properti.nama}")
    println("Harga : ${properti.harga}")
    println("Jumlah : ${properti.jumlah}")
    println("Diskon : ${properti.diskon}")
}

fun hitungTotal(harga:Int=750000, jumlah:Int=2, diskon:Int=20): Double{
    var hasil = harga * jumlah * 1.00 * (diskon * 0.01)
```

```
    return hasil
}

fun main(args: Array<String>){
    tampil()
    println("Total diskon: ${hitungTotal()}")
}

class Barang(val nama:String="RAM 8GB",val harga:Int=750000,
             val jumlah:Int=2,val diskon:Int=20){
}
```

```
Nama : RAM 8GB
Harga : 750000
Jumlah : 2
Diskon : 20
Total diskon: 300000.0
```

Penjelasan:
Pertama kita membuat fungsi tampil() yang di dalamnya kita membuat objek bernama property. Kemudian, kita tampilkan nilai default pada parameter konstruktor Barang dengan menuliskan ***println("Nama : ${properti.nama}")***. Kode tersebut berlaku juga bagi parameter lain yang akan ditampilkan seperti harga, jumlah, dan diskon.

Lalu kita buat fungsi bernama hitungTotal untuk menghitung total diskon. Kita juga beri nilai default pada fungsi tersebut. Langsung saja kita tuliskan ***fun hitungTotal(harga:Int=750000, jumlah:Int=2, diskon:Int=20): Double***. Di dalamnya ada variabel hasil untuk menghitung total diskon yaitu ***var hasil = harga * jumlah * 1.00 * (diskon * 0.01)***. Kemudian kita proses dan return perhitungannya dengan menuliskan ***return hasil***.

Terakhir, kita buat konstruktor beserta nilai default pada parameternya supaya nilai tersebut bisa digunakan di fungsi tampil() yang tadi kita sudah buat di awal program. Langsung saja kita tulis baris kode ***class Barang(val nama:String="RAM 8GB",val harga:Int=750000,val jumlah:Int=2,val diskon:Int=20){}***

KESIMPULAN

Di pertemuan ke-2 ini, saya berhasil membuat laporan berdasarkan modul 2 dan saya dapat memenuhi beberapa tujuan dari modul 2 yaitu mampu mengimplementasikan dasar-dasar pemrograman dengan Kotlin dan juga mampu mengimplementasikan fungsi dan dipanggil dalam program. Di praktikum pertemuan ke-2 ini, saya mendapat banyak ilmu terutama mengenai fungsi, konstruktor, dan kelas data. Saya mampu membuat fungsi daftar pustaka maupun fungsi yang saya definisikan sendiri, mampu membuat konstruktor beserta nilai default nya, dan mampu membuat data class untuk menampung beberapa properti dala program. Di bagian latihan, saya juga belajar membuat beberapa fungsi yang dipadukan dengan rumus-rumus matematika seperti rumus teorema Pythagoras, dan rumus matematika untuk menghitung total diskon. Saya mendapat banyak ilmu di pertemuan kali ini.

Terima Kasih