// LATIHAN 1

// val nilai = 85

// val grade = when {

// nilai >= 90 -> 'A'

// nilai >= 80 -> 'B'

// nilai >= 70 -> 'C'

// nilai >= 60 -> 'D'

// else -> 'E'

// }

// println("Nilai: $nilai")

//

//LATIHAN 2

// val buku = 12000

// val pensil = 8500

// val pulpen = 20000

// val total = buku + pensil + pulpen

// println("$total")

//LATIHAN 3

// val angka = listOf(2, 7, 10, 13, 18, 21)

// val angkaGenap = angka.filter {it%2==0}

// println("Angka genap: $angkaGenap")

//LATIHAN 4

// val nama = listOf("Ana", "Bambang", "Cristbhynyn6", "Dewinh")

// val namaTerpanjang = nama.maxByOrNull {it.length}

// println("Nama dengan karakter terbanyak: $namaTerpanjang")

//LATIHAN 5

// val stok = listOf(10, 5)

// val total = stok.sum()

// println ("total stok barang $total")

// fun surveiKepuasan() {

// println("=== SURVEI KEPUASAN ===")

// // Map<namaSiswa, List<Int>>: [Kebersihan, Fasilitas, Pelayanan, Administrasi, Keamanan]

// val skor: Map<String, List<Int>> = mapOf(

// "Alya" to listOf(5, 4, 4, 4, 5),

// "Bima" to listOf(3, 3, 2, 3, 3),

// "Citra" to listOf(4, 5, 5, 4, 4),

// "Danu" to listOf(2, 3, 3, 2, 2),

// "Eka" to listOf(4, 4, 3, 4, 4)

// )

// val kategori = listOf("Kebersihan","Fasilitas","Pelayanan","Administrasi","Keamanan")

// // 1) Rata-rata & status

// fun status(avg: Double) = when {

// avg >= 4.0 -> "Sangat Puas"

// avg >= 3.0 -> "Puas"

// else -> "Tidak Puas"

// }

// val rataPerSiswa = skor.mapValues { (\_, v) -> v.average() }

// rataPerSiswa.forEach { (nama, avg) ->

// println("$nama -> rata2=${"%.2f".format(avg)} | ${status(avg)}")

// }

// // 2) Siswa dengan rata-rata tertinggi

// val (namaTop, avgTop) = rataPerSiswa.maxByOrNull { it.value }!!

// println("Rata-rata tertinggi: $namaTop (${String.format("%.2f", avgTop)})")

// // 3) Rata-rata tiap kategori

// val rataPerKategori = kategori.indices.associate { idx ->

// kategori[idx] to skor.values.map { it[idx] }.average()

// }

// println("Rata-rata per kategori:")

// rataPerKategori.forEach { (k, avg) ->

// println(" - $k: ${"%.2f".format(avg)}")

// }

// // 4) Semua siswa yang memberi < 3 pada kategori 'Pelayanan'

// val idxPelayanan = kategori.indexOf("Pelayanan")

// val siswaPelayananBuruk = skor.filter { (\_, v) -> v[idxPelayanan] < 3 }.keys

// println("Siswa skor < 3 pada 'Pelayanan': ${if (siswaPelayananBuruk.isEmpty()) "-" else siswaPelayananBuruk.joinToString()}")

// println()

// }

// data class Peminjaman(

// val namaSiswa: String,

// val judulDipinjam: List<String>,

// val sudahDikembalikan: Boolean

// )

// fun latihan7() {

// println("=== LATIHAN 7 — Peminjaman Buku ===")

// val data = listOf(

// Peminjaman("Alya", listOf("Kotlin Dasar", "Struktur Data", "Matematika Diskrit"), false),

// Peminjaman("Bima", listOf("Pemrograman Web", "Basis Data"), true),

// Peminjaman("Citra", listOf("Kotlin Dasar", "Algoritma"), false),

// Peminjaman("Danu", listOf("Sistem Operasi"), true),

// Peminjaman("Eka", listOf("Jaringan Komputer", "Kotlin Dasar", "UI/UX", "Statistika"), false)

// )

// val belumKembali = data.filter { !it.sudahDikembalikan }.map { it.namaSiswa }

// println("Belum mengembalikan: ${belumKembali.joinToString()}")

// val judulUnik = data.flatMap { it.judulDipinjam }.map { it.trim() }.toSet()

// println("Judul unik: ${judulUnik.sorted().joinToString()}")

// val countLebih2 = data.count { it.judulDipinjam.size > 2 }

// println("Siswa meminjam > 2 buku: $countLebih2 orang")

// val target = "kotlin dasar"

// val listKotlinDasar = data.filter { pin ->

// pin.judulDipinjam.any { it.equals(target, ignoreCase = true) }

// }.map { it.namaSiswa }

// println("Siswa yang meminjam 'Kotlin Dasar': ${listKotlinDasar.joinToString()}")

// val totalBuku = data.sumOf { it.judulDipinjam.size }

// println("Total buku dipinjam: $totalBuku")

// println()

// }

// fun latihan8() {

// println("=== LATIHAN 8 — Kehadiran ===")

// // 10 siswa, 8 pertemuan (true=hadir, false=tidak)

// val hadir: Map<String, List<Boolean>> = mapOf(

// "Alya" to listOf(true, true, true, true, true, true, false, true),

// "Bima" to listOf(true, false, true, false, true, false, true, false),

// "Citra" to listOf(true, true, true, true, true, true, true, true),

// "Danu" to listOf(false, false, true, false, true, false, false, true),

// "Eka" to listOf(true, true, false, true, true, true, true, false),

// "Fahri" to listOf(true, true, true, false, false, true, true, true),

// "Gita" to listOf(false, false, false, false, true, true, false, false),

// "Hana" to listOf(true, true, true, true, false, true, true, true),

// "Ivan" to listOf(true, false, false, true, true, false, true, true),

// "Joko" to listOf(false, true, true, false, false, true, false, true)

// )

// val totalPertemuan = hadir.values.first().size

// val minimal = 6

// // 1) Jumlah kehadiran tiap siswa

// val jumlahHadir = hadir.mapValues { (\_, list) -> list.count { it } }

// println("Jumlah hadir per siswa:")

// jumlahHadir.forEach { (nama, jml) -> println(" - $nama: $jml/$totalPertemuan") }

// // 2) Siswa yang tidak memenuhi syarat minimal 6 kali hadir

// val tidakMemenuhi = jumlahHadir.filter { it.value < minimal }.keys

// println("Tidak memenuhi minimal $minimal hadir: ${if (tidakMemenuhi.isEmpty()) "-" else tidakMemenuhi.joinToString()}")

// // 3) Siswa dengan kehadiran tertinggi & terendah

// val (namaMax, maxH) = jumlahHadir.maxByOrNull { it.value }!!

// val (namaMin, minH) = jumlahHadir.minByOrNull { it.value }!!

// println("Kehadiran tertinggi: $namaMax ($maxH)")

// println("Kehadiran terendah: $namaMin ($minH)")

// // 4) Persentase rata-rata kehadiran kelas

// val totalHadirSemua = jumlahHadir.values.sum()

// val persenRataKelas = totalHadirSemua.toDouble() / (hadir.size \* totalPertemuan) \* 100.0

// println("Persentase rata-rata kehadiran kelas: ${"%.2f".format(persenRataKelas)}%")

// // 5) Pesan jika kehadiran < 75%

// hadir.forEach { (nama, list) ->

// val p = list.count { it }.toDouble() / totalPertemuan \* 100.0

// if (p < 75.0) {

// println(" - $nama: ${"%.1f".format(p)}% -> Tidak layak ikut ujian")

// }

// }

// println()

// }

// data class Booking(

// val namaSiswa: String,

// val hari: String, // contoh: "Senin", "Kamis", "Jumat"

// val jam: String, // contoh: "08.00–10.00"

// val lab: String, // contoh: "Lab AI"

// val disetujui: Boolean

// )

// fun latihan9() {

// println("=== LATIHAN 9 — Booking Lab ===")

// val bookings = listOf(

// Booking("Alya", "Senin", "08.00–10.00", "Lab AI", true),

// Booking("Bima", "Kamis", "08.00–10.00", "Lab AI", false),

// Booking("Citra", "Jumat", "10.00–12.00", "Lab Jaringan", true),

// Booking("Danu", "Jumat", "13.00–15.00", "Lab AI", true),

// Booking("Eka", "Rabu", "09.00–11.00", "Lab Multimedia", false),

// Booking("Alya", "Kamis", "08.00–10.00", "Lab AI", true) // Alya pesan lagi

// )

// // 2) Pemesanan yang belum disetujui

// val pending = bookings.filter { !it.disetujui }

// println("Belum disetujui:")

// pending.forEach { println(" - ${it.namaSiswa} | ${it.hari} ${it.jam} @ ${it.lab}") }

// // 3) Total pemesanan per laboratorium

// val totalPerLab = bookings.groupingBy { it.lab }.eachCount()

// println("Total pemesanan per lab:")

// totalPerLab.forEach { (lab, tot) -> println(" - $lab: $tot") }

// // 4) Daftar siswa yang memesan pada hari Jumat

// val siswaJumat = bookings.filter { it.hari.equals("Jumat", ignoreCase = true) }.map { it.namaSiswa }.toSet()

// println("Siswa yang booking hari Jumat: ${siswaJumat.joinToString()}")

// // 5) Fungsi cek apakah "Lab AI" dipakai di jam tertentu pada hari tertentu

// fun labAIDipakai(hari: String, jam: String): Boolean =

// bookings.any { it.lab.equals("Lab AI", true) && it.hari.equals(hari, true) && it.jam == jam }

// println("Cek Lab AI Kamis 08.00–10.00: ${if (labAIDipakai("Kamis","08.00–10.00")) "TERPAKAI" else "KOSONG"}")

// println("Cek Lab AI Selasa 08.00–10.00: ${if (labAIDipakai("Selasa","08.00–10.00")) "TERPAKAI" else "KOSONG"}")

// // 6) Siswa yang memesan lebih dari satu kali

// val multiBooker = bookings.groupBy { it.namaSiswa }.filter { (\_, v) -> v.size > 1 }.keys

// println("Siswa pesan > 1 kali: ${if (multiBooker.isEmpty()) "-" else multiBooker.joinToString()}")

// println()

// }

// fun latihan10() {

// println("=== LATIHAN 10 — Bahasa Favorit ===")

// // Map<nama, List<bahasa> > (3 pilihan)

// val pilihan: Map<String, List<String>> = mapOf(

// "Alya" to listOf("Kotlin", "Python", "C++"),

// "Bima" to listOf("Java", "C", "Python"),

// "Citra" to listOf("Python", "Kotlin", "C#"),

// "Danu" to listOf("C", "C++", "Java"),

// "Eka" to listOf("Kotlin", "Java", "Python"),

// "Fahri" to listOf("C#", "Python", "Kotlin")

// )

// // Normalisasi (trim)

// fun norm(s: String) = s.trim()

// // 1) Semua bahasa yang pernah dipilih (unik)

// val semuaBahasa = pilihan.values.flatten().map(::norm).toSet().sorted()

// println("Semua bahasa yang pernah dipilih: ${semuaBahasa.joinToString()}")

// // 2) Jumlah pemilih tiap bahasa, urut desc

// val hitungBahasa = pilihan.values.flatten().map(::norm).groupingBy { it }.eachCount()

// .toList().sortedByDescending { it.second }

// println("Jumlah pemilih per bahasa (terbanyak -> tersedikit):")

// hitungBahasa.forEach { (bhs, n) -> println(" - $bhs: $n") }

// // 3) Siswa yang memilih "Kotlin" sebagai pilihan pertama

// val kotlinPertama = pilihan.filter { (\_, v) -> v.isNotEmpty() && norm(v[0]).equals("Kotlin", true) }.keys

// println("Kotlin sebagai pilihan pertama: ${kotlinPertama.joinToString()}")

// // 4) Berapa siswa yang memilih "Python" di posisi manapun

// val countPython = pilihan.count { (\_, v) -> v.any { it.equals("Python", true) } }

// println("Jumlah siswa yang memilih 'Python' di posisi manapun: $countPython")

// // 5) Bahasa yang tidak dipilih sama sekali dari list target

// val target = listOf("Kotlin","Java","C++","Python","C","C#")

// val tidakDipilih = target.filter { t -> hitungBahasa.none { it.first.equals(t, true) } }

// println("Bahasa yang tidak dipilih sama sekali: ${if (tidakDipilih.isEmpty()) "-" else tidakDipilih.joinToString()}")

// println()

// }