|  |  |  |
| --- | --- | --- |
| **Nama & NPM** | **Topik:** | **Tanggal:** |
| **Randi Satria Anggarra**  **G1F024048** | **For dan while** | **10 oktober 2024** |
| **Unit 1 for** | | |
| 1.1. Analisis tujuan continue dan break pada Contoh 1:  continue digunakan untuk melompati iterasi saat ini dan melanjutkan ke iterasi berikutnya.  break digunakan untuk menghentikan seluruh perulangan.  Perubahan dan dampaknya:  a) for (int y = 0; y <= 15; y++):  Ini akan memperpanjang range perulangan, potensial menghasilkan output lebih banyak.  b) if (y % 2 == 0):  Ini akan mencetak angka genap, bukan ganjil seperti sebelumnya.  c) else if (y == 9):  Perulangan akan berhenti saat y mencapai 9, bukan 8.  1.2. Perubahan pada Contoh 2:  a) continue pertama;:  Akan melompati iterasi saat ini dari loop luar dan melanjutkan ke iterasi berikutnya.  b) break pertama;:  Akan menghentikan seluruh perulangan bersarang.  c) continue kedua;:  Akan berperilaku sama seperti break kedua; karena berada di akhir loop dalam.  1.3  Code:    Output:    Penjelasan:  Perulangan luar tetap sama, menghitung mundur dari tinggi ke 1.  Perulangan dalam diubah untuk mencetak bintang dari 1 sampai nilai t saat ini.  **Unit 2 while**  2.1. Perubahan break; menjadi continue; pada Contoh 4:  Dengan break;, program berhenti saat i mencapai 4.  Dengan continue;, program akan melompati iterasi saat i=4, tapi melanjutkan hingga i=6.  Kegunaan: break: Menghentikan seluruh perulangan.  continue: Melompati sisa kode dalam iterasi saat ini dan melanjutkan ke iterasi berikutnya.  2.2. Perubahan if (count % 3 == 0) menjadi if (count % 5 == 0) pada Contoh 5:  Ini akan mencetak kelipatan 5 (0, 5, 10, 15) alih-alih kelipatan 3.  Operator % (modulo) mengembalikan sisa pembagian. x % y == 0 berarti x habis dibagi y.  2.3. Perubahan while (count < 20) menjadi while (count < 0):  Dengan while, tidak ada output karena kondisi false dari awal.  Dengan do...while, akan ada satu iterasi (mencetak 0) sebelum loop berhenti.  Code:    Output:    Penjelasan:  Perbedaan utama: do...while selalu menjalankan blok kode setidaknya sekali sebelum memeriksa kondisi, sementara while memeriksa kondisi terlebih dahulu. | | |
| **KESIMPULAN**  Kesimpulannya, pemahaman mendalam tentang berbagai jenis perulangan dan pernyataan kontrolnya sangat penting dalam pemrograman Java. Ini memungkinkan pengembang untuk menulis kode yang efisien, mudah dibaca, dan sesuai dengan kebutuhan spesifik dari algoritma yang sedang diimplementasikan. | | |