|  |  |  |
| --- | --- | --- |
| **Nama & NPM** | **Topik:** | **Tanggal:** |
| **WINDA DWI SALISTIANNA**  **G1F024004** | **FOR DAN WHILE JAVA** | **30 SEPTEMBER 2024** |
| **UNIT 1 FOR** | | |
| **Latihan 1**  1.1.    Analisa tujuan penulisan kata kunci continue dan break pada Contoh 1!            Buat perubahan nilai angka pada variabel di            //Ubah 1 menjadi  for (int y = 0; y <= 15; y++) {  lalu running, periksa hasilnya           //Ubah 2 menjadi  if (y % 2 == 0)  lalu running, periksa hasilnya           //Ubah 3 menjadi else if (y == 9)  lalu running, periksa hasilnya           Analisa dampaknya perubahan ini terhadap luaran setelah running!  1.2.    Buat perubahan kode pada Contoh 2 di baris //Ubah1 menjadi           a. continue pertama;  lalu  running, periksa hasilnya           b. break pertama; lalu running, periksa hasilnya           c. continue kedua; lalu running, periksa hasilnya           Analisa perbedaan perubahan kode pada Ubah 1 untuk setiap poin (a), (b), dan (c)!  1.3.    Cermati kode contoh 3. Apabila ingin menghasilkan luaran berikut: Luaran: Masukan Input: 7 \*\*\*\*\*\*\* \*\*\*\*\*\* \*\*\*\*\* \*\*\*\* \*\*\* \*\* \*            Susunlah analisa kode untuk menghasilkan luaran tersebut!  1.4.      Analisa diagram flowchart dari Latihan 1.2 dan 1.3!  JAWABAN :  **1.1 Tujuan Penulisan continue dan break**  continue: Digunakan untuk melompati sisa iterasi saat kondisi tertentu terpenuhi, ketika angka ganjil.  break: Digunakan untuk menghentikan perulangan secara prematur, ketika angka mencapai 8.  Gambar 1.1 kode for  Penjelasan kode :  Kode pada gambar 1.1 adalah penggunaan `for` untuk mencetak angka ganjil dari 1 hingga 15. Di dalam loop, terdapat kondisi yang memeriksa apakah angka `y` genap menggunakan `if (y % 2 == 0)`. Jika kondisi ini benar, program akan melanjutkan ke iterasi berikutnya dengan `continue`, sehingga tidak mencetak angka genap. Jika `y` sama dengan 9, program akan menghentikan eksekusi loop menggunakan `break`. Hanya angka yang memenuhi kriteria sebagai angka ganjil (1, 3, 5, 7) yang akan dicetak ke layar. Hasilnya, output program adalah angka ganjil dari 1 sampai 7, dan angka 9 tidak akan dicetak.  Output:  Gambar 1.2 output  ubah 1: Memperluas rentang angka yang dicetak.  ubah 2 : Membalikkan kondisi, sehingga hanya bilangan ganjil yang dicetak.  ubah 3: Mengubah titik berhenti perulangan 9.  **1.2 Perubahan kode**  Sebuah gambar berisi teks, cuplikan layar, Font  Deskripsi dibuat secara otomatisa). Continue pertama  Sebuah gambar berisi teks, cuplikan layar, Font, nomor  Deskripsi dibuat secara otomatisGambar 1.3 codingan  Gambar 1.4 Output  Ketika i mencapai 2, continue pertama; akan menghentikan eksekusi loop kedua dan kembali ke iterasi berikutnya di loop pertama. Ketika i = 2, outputnya: i = 2; j = 1 Pada titik ini, loop kedua diulang kembali dari awal, dan i akan meningkat ke 3. Loop kedua tidak akan pernah mencetak nilai untuk j pada i = 2 lebih dari 1 karena continue menyebabkan eksekusi kembali ke loop pertama.  b). Break pertama  Gambar 1.5 break  Gambar 1.6 output  Ketika i mencapai 2, break pertama; akan menghentikan seluruh loop pertama, dan program akan keluar dari kedua loop. Output untuk i = 1 akan tetap sama:  i = 1; j = 1  i = 1; j = 2  Ketika i mencapai 2, program berhenti, jadi tidak ada output untuk i = 2, i = 3, atau i = 4.  Sebuah gambar berisi teks, cuplikan layar, Font, nomor  Deskripsi dibuat secara otomatisc). Continue kedua  Gambar 1.7 kode  Sebuah gambar berisi teks, cuplikan layar, Font, nomor  Deskripsi dibuat secara otomatis    Gambar 1.8 output  Ketika i mencapai 2, continue kedua; akan menghentikan eksekusi untuk iterasi j saat itu dan melanjutkan ke iterasi berikutnya di loop kedua. Untuk i = 1, outputnya tetap sama:  i = 1; j = 1  i = 1; j = 2  Untuk i = 2, hanya akan mencetak j = 1, dan saat mencapai j = 2, akan kembali ke awal loop kedua untuk iterasi selanjutnya, tetapi karena i sudah 2, j akan tetap berjalan ke 2.  **1.3 code \***  Source code :  import java.util.Scanner;  public class ForBersarang {  public static void main(String[] args) {  // Instance Input Scanner  Scanner input = new Scanner(System.in);  System.out.print("Masukan Input: ");  int tinggi = input.nextInt(); // Mendapatkan Input Dari User  for (int t = tinggi; t >= 1; t--) {  // Menghitung Jumlah Spasi per Baris  for (int s = t; s < tinggi; s++) {  System.out.print(" "); // Mencetak spasi untuk penempatan bintang  }  // Menghitung Jumlah Bintang per Baris  for (int b = 1; b <= t; b++) {  System.out.print("\*"); // Mencetak bintang  }  System.out.println(); // Membuat Baris Baru  }  input.close(); // Menutup Scanner  }  }  Luaran  Masukan Input: 7  \*\*\*\*\*\*\*  \*\*\*\*\*\*  \*\*\*\*\*  \*\*\*\*  \*\*\*  \*\*  \*  **1.4.      Analisa diagram flowchart dari Latihan 1.2 dan 1.3!**  Flowchart 1.2 for bersarang  Mulai  false  If ( i == 2)  Continue pertama  Break pertama    true  Continue kedua  increment  Benar  Selesai  Flowchart 1.3 for bersarang  Mulai  false  Masukan Input  for (int t = tinggi; t >= 1; t--)  for (int s = t; s < tinggi; s++)    true  for (int b = 1; b <= t; b++)  increment  Benar  Selesai | | |
| **UNIT 2 WHILE** | | |
| **Latihan 2**  2.1.  Buat perubahan nilai angka pada variabel di Contoh 4         //Ubah 1 menjadi continue;  lalu running, periksa hasilnya         Analisa dampaknya perubahan terhadap luaran setelah running dan uraikan kegunaan break dan continue!  2.2. Buat perubahan nilai angka pada variabel di Contoh 5        //Ubah2 menjadi if (count % 5 == 0)  lalu running, periksa hasilnya        Analisa dampaknya perubahan terhadap luaran setelah running dan uraikan kegunaan % untuk angka yang berbeda         pada perintah tersebut!  2.3. Buat perubahan nilai angka pada variabel di         //Ubah1 menjadi while (count < 0) {    lalu running, periksa hasilnya        Ubahlah baris kode while pada Contoh 5 menjadi do … while dengan persyaratan yang sama  while (count < 0).   Bandingkan hasil luaran antara menggunakan while dan do … while!  2.4.  Analisa diagram flowchart dari Latihan 2.1, Contoh 5, dan Latihan 2.3!  **JAWABAN**  **2.1 perubahan nilai**  Break :  Sebuah gambar berisi teks, cuplikan layar, Font, tampilan  Deskripsi dibuat secara otomatis    Gambar 2.1 code whilee  Sebuah gambar berisi teks, cuplikan layar, Font, garis  Deskripsi dibuat secara otomatis  Gambar 2.2 Output    Sebuah gambar berisi teks, cuplikan layar, Font, nomor  Deskripsi dibuat secara otomatisContinue :  Sebuah gambar berisi teks, cuplikan layar, Font, tampilan  Deskripsi dibuat secara otomatisGambar 2.3 code  Gambar 2.4 output  **Dampak perubahan break**: Menghentikan eksekusi loop sepenuhnya. Ketika i mencapai 4, loop berhenti, sehingga tidak ada angka yang dicetak setelah 3. **continue**: Menghentikan iterasi dan melanjutkan ke iterasi berikutnya. Ketika i mencapai 4, continue akan membuat loop melompat ke langkah selanjutnya, yang berarti program akan mencetak 4 dan melanjutkan mencetak 5 dan 6.  **Kegunaan break**: berguna untuk menghentikan loop lebih awal berdasarkan kondisi tertentu. melanjutkan iterasi lebih lanjut. **Sedangkan continue**: berguna untuk melewatkan eksekusi sisa kode dalam iterasi saat ini dan melanjutkan ke iterasi berikutnya.  **2.2 Perubahan nilai**  **Sebuah gambar berisi teks, cuplikan layar, Font, nomor  Deskripsi dibuat secara otomatis**  Gambar 2.5 While bersarang  Sebuah gambar berisi teks, cuplikan layar, Font, tampilan  Deskripsi dibuat secara otomatis  Gambar 2.6 Output  **Kondisi** if (count % 5 == 0) memeriksa apakah count dapat dibagi oleh 5 tanpa sisa. Jika iya, maka count dicetak ke konsol.  count % 5 == 0: Menghasilkan true untuk kelipatan 5 (0, 5, 10, 15, ...).  **2.3 Perubahan nilai**  Sebuah gambar berisi teks, cuplikan layar, Font, tampilan  Deskripsi dibuat secara otomatisWhile:  Gambar 2.7 Code dan Output  Hasil running gambar 2.7 tidak ada Karena count diinisialisasi dengan 0 dan kondisi while (count < 0) tidak terpenuhi, **tidak ada output** yang akan dihasilkan.  do...while :  Sebuah gambar berisi teks, cuplikan layar, software, Font  Deskripsi dibuat secara otomatis  Gambar 2.8 Code dan Output  Gambar 2.8 menghasilkan output 0 karena blok do dieksekusi sekali sebelum memeriksa kondisi, meskipun kondisi tidak terpenuhi untuk iterasi berikutnya.  **Perbandingan while dan do...while**  **While** :Hasil running tidak ada Karena count diinisialisasi dengan 0 dan kondisi while (count < 0) tidak terpenuhi, **tidak ada output** yang akan dihasilkan.  **do...while**: Menghasilkan output 0 karena blok do dieksekusi sekali sebelum memeriksa kondisi, meskipun kondisi tidak terpenuhi untuk iterasi berikutnya.  **2.4 diagram flowchart dari Latihan 2.1, Contoh 5, dan Latihan 2.3!**  2.1 flowchart 2.1  Mulai  Continue  While (i <=6)  If (i == 4)  Incerment  Selesai  Mulai  Count  If (Count % 3 == 0)  Incerment  While (count < 0)  True  False  Selesai | | |
| **Kesimpulan** | | |
| Kesimpulan tentang materi ini adalah bahwa kedua struktur pengulangan ini memiliki fungsi yang penting dalam mengontrol alur program, memungkinkan eksekusi kode secara berulang berdasarkan kondisi tertentu. For sering digunakan ketika jumlah iterasi sudah diketahui sebelumnya, seperti saat menghitung atau mencetak elemen dalam rentang tertentu. Sementara itu, while lebih fleksibel dan cocok digunakan ketika jumlah iterasi tidak dapat ditentukan sebelumnya, mengandalkan kondisi yang harus dipenuhi. Keduanya juga mendukung penggunaan kata kunci break dan continue, yang memberikan kontrol tambahan terhadap proses pengulangan, seperti menghentikan loop secara prematur atau melompati iterasi tertentu. Memahami cara kerja dan penerapan kedua struktur ini sangat penting untuk mengembangkan algoritma yang efisien dan efektif dalam pemrograman. | | |