Template Lembar Kerja Individu dan Kelompok

Nama & NPM	Topik:	Tanggal:
Renni Ren Rofa.Pgb G1F021002	**Perulangan: FOR dan WHILE**	**10 Oktober 2024**

[No.1] Identifikasi Masalah:

1.1 Evaluasi penyebab kesalahan dan perbaiki kode pada Contoh 1!

Rekomendasikan kata kunci yang tepat diletakkan pada baris kode yang kosong 1 dan 2 untuk dapat menghasilkan luaran berikut:

Luaran contoh 1:

0
2
4
6

1.2 Cermati contoh kode 2 pada kode //baris kode kosong.

Rekomendasikan kode yang tepat menggunakan break atau continue terhadap pertama atau kedua agar menghasilkan luaran berikut:

Luaran Contoh 2:

i = 1; j = 1
i = 1; j = 2
i = 2; j = 1
i = 2; j = 2

1.3. Cermati kode contoh 3. Apabila ingin menghasilkan luaran berikut:

Luaran berbentuk piramida

Masukan Input: 7

```
    *
   ***
  *****
 *******
*********
***********
*************
```

Rekomendasikan kode untuk menghasilkan luaran tersebut!

1.4. Analisa diagram flowchart dari Latihan 1.2 dan 1.3!

[No.1] Analisis dan Argumentasi

Solusi : melibatkan perbaikan sintaks pada kedua contoh program:

- Pada Contoh 1, perbaikan dilakukan dengan mendeklarasikan variabel kontrol y dengan benar dan menggunakan tipe data int agar kondisi modulus dapat dievaluasi.
- Pada Contoh 2, aliran perulangan diperbaiki dengan menambahkan perintah continue di dalam perulangan bersarang.

Solusi ini memastikan bahwa program berjalan tanpa kesalahan kompilasi dan memberikan keluaran yang diharapkan. Dengan memperbaiki struktur perulangan dan mengontrol aliran dengan continue, program menghasilkan keluaran yang diinginkan**.**

[No.1] Penyusunan Algoritma dan pembahasan

1.1 Perbaikan contoh 1

- Kesalahan: Pada contoh ini, masalah terjadi pada baris perulangan for karena urutan dan logika penulisan yang salah.
- Perbaikan kode:

```
public class ContohFor {
```

```
    public static void main(String[] args) {
      for (int y = 0; y <= 15; y++) { // Perbaikan inisialisasi variabel
        if (y % 2 == 0) {          // kondisi 1 (bilangan genap)
          System.out.println(y + " ");
        } else if (y == 8) {       // kondisi 2 (tidak ada aksi untuk 8)
          continue;               // baris kode kosong 2
        }
      }
    }
  }
```

1.2 Perbaikan contoh 2

- Kesalahan: Tidak ada kesalahan sintaks, namun potongan kode yang hilang setelah kondisi if (i == 2) perlu diisi untuk mengontrol aliran program. Untuk hasil yang sesuai, gunakan continue.
- Rekomendasi kode :

```
public class ForBersarang {
  public static void main(String[] args) {
    pertama:
    for( int i = 1; i < 5; i++) {
      kedua:
      for(int j = 1; j < 3; j++) {
        System.out.println("i = " + i + "; j = " + j);
      }
      if (i == 2) {
        continue pertama;   // kode yang hilang
      }
    }
  }
}
```

1.3 Perbaikan contoh 3

- Kesalahan: Untuk membuat piramida yang sesuai, kamu perlu mengatur jumlah spasi sebelum mencetak tanda bintang agar tampilan rapi dan membentuk piramida.
- Rekoemndasi kode:

```
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 = 1; t <= tinggi; t++) {
      // Loop untuk mencetak spasi
      for (int s = tinggi; s > t; s--) {
        System.out.print(" ");
      }
      // Loop untuk mencetak bintang
      for (int b = 1; b <= (2 * t - 1); b++) {
        System.out.print("*");
      }
      System.out.println(); // Membuat baris baru
    }
```

```
        }
    }
```

[No.1] Kesimpulan

Masalah utama adalah kesalahan sintaks dan aliran kontrol pada perulangan. Solusi yang diterapkan memperbaiki kedua masalah ini, sehingga program menghasilkan keluaran yang diinginkan.Penggunaan continue untuk mengatur perulangan dalam contoh bersarang, dan perbaikan tipe data untuk memastikan kondisi logika dapat dieksekusi dengan benar.

Latihan 2

[No. 2] Identifikasi Masalah:

1) Uraikan permasalahan

2.1 Ubahlah baris kode pada Contoh 4

//Ubah1 menjadi if(i % 3 == 0){ ◊ running, periksa hasilnya

//Ubah2 menjadi continue; ◊ running, periksa hasilnya

Evaluasi perbandingan luaran sebelum dan setelah diubah! Simpulkan maksud dari perubahan tersebut!

2.2. Cermati Contoh 5. Periksa luaran, bila ketika di eksekusi, jumlah yang diulang = 0! Evaluasi luaran, bila kode diubah menjadi do … while dengan masukan sama jumlah yang diulang = 0.

Simpulkan perbedaan while dan do … while!

2.3. Bila diketahui pernyataan pseudocode berikut:

[1] inisiasi idPelajaran
[2] inisiasi nilai pelajaran
[3] inisiasi nilai rata-rata
[4] Minta pengguna untuk menuliskan jumlah pelajaran
[5] Ketika idPelajaran lebih kecil dari jumlah pelajaran
[6] Minta pengguna untuk menuliskan nilai pelajaran
[7] Hitung nilai rata-rata = (nilai pelajaran + nilai rata-rata) / 2
[8] Tambah satu ke idPelajaran
[9] Tampilkan nilai rata-rata

Rekomendasikan kode untuk menyelesaikan Pseudocode tersebut!

[No.2] Analisis dan Argumentasi

- Contoh 1: Perbaikan logika for dengan mengubah urutan parameter perulangan, memastikan tipe data konsisten, serta menambahkan kondisi yang tepat di dalam blok if-else.
- Contoh 2: Menyisipkan kontrol aliran menggunakan continue atau break di dalam loop bersarang untuk memenuhi output yang diinginkan.
- Contoh 3: Mengubah struktur piramida bintang agar menggunakan perulangan bersarang dengan spasi dan tanda bintang.

[No.2] Penyusunan Algoritma dan Kode Program

2.1 Ubah Baris Kode pada Contoh 4

Penjelasan:

- Sebelum diubah: Program akan mencetak semua nilai i yang sesuai dengan logika perulangan tanpa ada kondisi tambahan atau interupsi dalam proses iterasi.
- Setelah diubah:
  - Ubah1 (if i % 3 == 0): Hanya bilangan yang habis dibagi 3 (i % 3 == 0) yang akan memenuhi kondisi dan melakukan perintah di dalam if.

- Ubah2 (continue): Ketika kondisi if terpenuhi, continue digunakan untuk melewatkan sisa iterasi saat ini dan melanjutkan ke iterasi berikutnya tanpa mengeksekusi sisa kode dalam blok perulangan.

Evaluasi Perbandingan Luaran:

- Sebelum diubah: Program mungkin mencetak semua nilai i sesuai dengan kondisi perulangan.
- Setelah diubah: Program akan melewati setiap nilai i yang habis dibagi 3, dan hanya melanjutkan iterasi ke nilai berikutnya, tanpa mencetak nilai i tersebut.

Kesimpulan Maksud dari Perubahan:
Perubahan tersebut membuat program tidak memproses nilai i yang habis dibagi 3 (i % 3 == 0), karena perintah continue akan menghentikan eksekusi iterasi saat kondisi ini terpenuhi, dan langsung beralih ke iterasi berikutnya.

2.2. Evaluasi Perbedaan while dan do … while

Penjelasan :

Periksa Luaran pada Contoh 5:

- Ketika jumlah perulangan = 0 dalam struktur while, perulangan tidak akan dieksekusi karena kondisi dievaluasi terlebih dahulu. Jadi, jika kondisi jumlah == 0, perulangan tidak terjadi.

Evaluasi jika Kode Diubah Menjadi do … while:

- Dalam struktur do ... while, perulangan selalu dijalankan minimal sekali sebelum memeriksa kondisi. Oleh karena itu, jika jumlah perulangan = 0, program tetap akan mengeksekusi blok kode sekali sebelum memeriksa kondisi jumlah == 0.

Kesimpulan Perbedaan while dan do … while:

- while: Mengevaluasi kondisi terlebih dahulu. Jika kondisi false, perulangan tidak akan dieksekusi sama sekali.
- do ... while: Mengeksekusi blok kode setidaknya sekali, kemudian memeriksa kondisi. Jika kondisi false, perulangan akan berhenti setelah satu iterasi.

2.3 Rekomendasi Kode untuk Pseudocode

Code:

```
import java.util.Scanner;

public class HitungRataRata {
   public static void main(String[] args) {
      // Inisiasi variabel
      int idPelajaran = 0;
      double nilaiPelajaran, rataRata = 0;

      // Membuat scanner untuk input
      Scanner input = new Scanner(System.in);

      // Meminta pengguna menuliskan jumlah pelajaran
      System.out.print("Masukkan jumlah pelajaran: ");
      int jumlahPelajaran = input.nextInt();

      // Perulangan untuk menghitung rata-rata
      while (idPelajaran < jumlahPelajaran) {
         System.out.print("Masukkan nilai pelajaran ke-" + (idPelajaran + 1) + ": ");
         nilaiPelajaran = input.nextDouble();
```

```
            // Hitung nilai rata-rata (metode kumulatif)
            rataRata = (nilaiPelajaran + rataRata) / 2;

            // Tambah idPelajaran
            idPelajaran++;
        }

        // Tampilkan hasil rata-rata
        System.out.println("Nilai rata-rata adalah: " + rataRata);

        // Tutup scanner
        input.close();
    }
}
```

Luaran :
Masukkan jumlah pelajaran: 3
Masukkan nilai pelajaran ke-1: 80
Masukkan nilai pelajaran ke-2: 90
Masukkan nilai pelajaran ke-3: 85
Nilai rata-rata adalah: 86.25

Penjelasan :

- Inisiasi: Variabel idPelajaran, nilaiPelajaran, dan rataRata diinisiasi sebelum perulangan dimulai.
- Input Jumlah Pelajaran: Pengguna diminta memasukkan jumlah pelajaran.
- Perulangan: Menggunakan perulangan while, setiap kali idPelajaran kurang dari jumlahPelajaran, program meminta nilai dari pengguna dan menghitung rata-rata secara kumulatif.
- Perhitungan Rata-Rata: Nilai rata-rata dihitung dengan cara (nilai saat ini + rata-rata sebelumnya) dibagi dua.
- Penambahan Id: Setelah setiap iterasi, nilai idPelajaran ditambah 1.

  Simpulan: Kode ini menyelesaikan pseudocode dengan mengimplementasikan input dari pengguna, menghitung nilai rata-rata secara kumulatif, dan menampilkan hasil akhir.

[No.2] Kesimpulan

Dari praktek latihan yang ada kita semua mendapatkan pengetahuan tentang penggunaan operator logika dan kontrol aliran dalam perulangan sangat penting untuk menghindari iterasi yang tidak diperlukan.

Konstruk Hubungan Variabel: Variabel loop seperti i dan j dalam loop bersarang memungkinkan kita untuk mengatur kombinasi perulangan dan membuat hasil yang sesuai dengan skenario spesifik yang diinginkan.