**Template Lembar Kerja Individu dan Kelompok**

|  |  |  |
| --- | --- | --- |
| **Nama & NPM** | **Topik:** | **Tanggal:** |
| **Syifa Ariqah Pajriyanti**  **(G1F024009)** | **IF & SWITCH JAVA** | **23 September 2024** |
| **[Nomor Soal] Identifikasi Masalah:** | | |
| 1. Uraikan permasalahan dan variable   **Contoh 1:**  import java.util.Scanner; //memanggil impor package yang membaca masukan pengguna  public class PercabanganIf {     public static void main(String[] args) { Scanner masuk = new Scanner(System.in);  // membaca teks yang dimasukkan pengguna System.out.print("Masukkan Angka Anda : ");  //pengguna memasukkan data         int nilai =  masuk.nextByte();  //menyimpan masukan pengguna ke tipe data       if (nilai == 10) {  //percabangan yang memeriksa kondisi System.out.println("Sepuluh");  //baris kode yang dieksekusi bila benar         } else  {  //baris kode yang dieksekusi bila kondisi tidak terpenuhi dan salah     System.out.println("Nilai Bukan Sepuluh"); }    } }  **Luaran  Contoh 1:** Masukkan Angka Anda : 8 Nilai Bukan Sepuluh    **Contoh 2:**  import java.util.Scanner;  public class IfBersarang {     public static void main(String[] args) {         Scanner varT = new Scanner(System.in);         System.out.print("Masukkan Angka Tugas Anda : ");         int nilaiT =  varT.nextByte();              Scanner varQ = new Scanner(System.in);         System.out.print("Masukkan Angka Quiz Anda : ");         int nilaiQ =  varQ.nextByte();                     if (nilaiU >= 80) {         if(nilaiT >= 80) {             System.out.println("Anda mendapatkan nilai A");         }     }     else{         System.out.println("Anda TIDAK mendapatkan nilai A");     }     } }  **Luaran Contoh 2:** Masukkan Angka Tugas Anda : 70 Masukkan Angka Quiz Anda : 70 Masukkan Angka UTS Anda : 70 Anda TIDAK mendapatkan nilai A  **Latihan 1:**  1.1.    Bilangan genap merupakan bilangan yang habis dibagi 2. Bilangan ganjil adalah bilangan yang tidak habis dibagi 2. Analisa kode program yang tepat untuk menghitung masukan pengguna termasuk bilangan genap atau bilangan ganjil (lihat Contoh 1)?           (Petunjuk: hitung = nilai % 2 jika hitung = 0 maka bilangan genap, jika hitung = 1 maka bilangan ganjil)  Jawab :  import java.util.Scanner;  public class PercabanganIf {  public static void main(String[] args) {  Scanner masuk = new Scanner(System.in);  System.out.print("Masukkan Angka Anda : ");  int nilai = masuk.nextByte();  if (nilai % 2 == 0) {  System.out.println("Bilangan genap");  }  else {  System.out.println("Bilangan ganjil");  } }  }  1.2.    Cermati contoh 2, analisa kondisi pada IF bersarang!             Tambahkan satu kondisi IF dengan satu nilai input Quiz (nilaiQ). Jika nilai UTS, Tugas, dan Quiz lebih besar sama dengan 80 maka siswa mendapat nilai A.  Jawab:  mport java.util.Scanner;  public class IfBersarang {  public static void main(String[] args) {  Scanner varT = new Scanner(System.in);  System.out.print("Masukkan Angka Tugas Anda : ");  int nilaiT = varT.nextInt();  Scanner varQ = new Scanner(System.in);  System.out.print("Masukkan Angka Quiz Anda : ");  int nilaiQ = varQ.nextInt();  Scanner varU = new Scanner(System.in);  System.out.print("Masukkan Angka Quiz Anda : ");  int nilaiU = varQ.nextInt();  if (nilaiU >= 80 && nilaiT >= 80 && nilaiQ >= 80){  System.out.println("Anda mendapatkan nilai A");  } else{  System.out.println("Anda TIDAK mendapatkan nilai A");  }  }  }  1.3.    Apakah ketiga kondisi IF pada Contoh 1.2. dapat diringkas menjadi satu kondisi?           Periksa satu kondisi mana yang paling tepat menggantikan  ketiga kondisi itu!  a.  IF (nilaiU >= 80 || nilaiT >= 80 || nilaiQ >= 80)           b.    IF (nilaiU >= 80 || nilaiT >= 80 && nilaiQ >= 80)           c.    IF (nilaiU >= 80 && nilaiT >= 80 || nilaiQ >= 80)           d.    IF (nilaiU >= 80 && nilaiT >= 80 && nilaiQ >= 80)  jawab: Kondisi yang tepat adalah IF (nilaiU >= 80 && nilaiT >= 80 && nilaiQ >= 80)  1.4.    Uraikan gambar diagram flowchart dari Latihan 1.2!     1. Rincikan sumber informasi yang relevan (buku / webpage)  * **Video Materi 1 tentang IF  –**[**https://www.youtube.com/watch?v=G0dfdAFa9iM**](https://www.youtube.com/watch?v=G0dfdAFa9iM) | | |
| **[Nomor 1] Analisis dan Argumentasi** | | |
| 1. Uraikan rancangan solusi yang diusulkan.    1. rancangan solusi yang diusulkan adalah   **Tujuan Program**:   * Menghitung dan menentukan apakah seorang siswa mendapatkan nilai A berdasarkan nilai tugas, kuis, dan ujian. Nilai A diberikan jika semua nilai (tugas, kuis, dan ujian) >= 80.   **Input**:   * Tiga angka yang mewakili: * Nilai tugas (nilaiT) * Nilai kuis (nilaiQ) * Nilai ujian (nilaiU) * Semua input harus berupa bilangan bulat.   **Proses**:   * Menggunakan kelas Scanner untuk membaca input dari pengguna. * Mengambil input nilai tugas, kuis, dan ujian satu per satu. * Memeriksa apakah semua nilai yang dimasukkan memenuhi kriteria (>= 80). * Menggunakan struktur kontrol if untuk menentukan output berdasarkan kriteria tersebut.   **Output**:   * Jika semua nilai >= 80, tampilkan pesan: "Anda mendapatkan nilai A". * Jika salah satu nilai < 80, tampilkan pesan: "Anda TIDAK mendapatkan nilai A".   **Langkah-langkah Implementasi**:   * Import kelas Scanner untuk menangani input. * Buat objek Scanner untuk membaca input. * Tampilkan prompt untuk memasukkan nilai tugas, kuis, dan ujian. * Baca dan simpan nilai ke dalam variabel. * Gunakan pernyataan if untuk memeriksa kriteria kelulusan. * Tampilkan hasil berdasarkan evaluasi. * Tutup objek Scanner setelah selesai.   1. rancangan solusi yang diusulkan adalah : * Penggabungan Scanner: Hanya perlu satu objek Scanner untuk mengambil input. Ini mengurangi redundansi dan membuat kode lebih bersih. * Validasi Input: Tambahkan validasi untuk memastikan bahwa pengguna hanya memasukkan nilai yang valid (misalnya, angka dalam rentang tertentu). * Pesan yang Lebih Jelas: Perbaiki pesan input untuk nilaiU agar tidak membingungkan. * Penanganan Kesalahan: Tambahkan penanganan kesalahan untuk mengatasi input yang tidak valid dan mencegah program crash. * Fungsi Terpisah: Pertimbangkan untuk memisahkan logika evaluasi nilai ke dalam metode terpisah untuk meningkatkan keterbacaan dan pemeliharaan kode.  1. Analisis solusi, kaitkan dengan permasalahan.    1. Solusi yang diusulkan sangat relevan dengan permasalahan yang ada dan menawarkan pendekatan yang efisien, jelas, dan mudah dipahami untuk menentukan apakah seorang siswa mendapatkan nilai A. Meskipun solusi sudah baik, ada ruang untuk perbaikan, terutama dalam hal validasi input untuk meningkatkan pengalaman pengguna. Dengan analisis ini, jelas bahwa solusi yang diusulkan tidak hanya memenuhi kebutuhan saat ini, tetapi juga memberikan dasar yang kuat untuk pengembangan di masa depan.    2. solusi yang diusulkan, program menjadi lebih robust, user-friendly, dan mudah dikelola. Kode yang bersih dan terstruktur meningkatkan efisiensi pengembangan dan pemeliharaan di masa depan. Analisis ini menunjukkan bahwa pendekatan yang lebih baik dalam pengelolaan input, penanganan kesalahan, dan struktur kode dapat secara signifikan mengatasi permasalahan yang ada, menjadikan aplikasi lebih handal dan profesional. | | |
| **[Nomor 1] Penyusunan Algoritma dan Kode Program** | | |
| 1. Rancang desain solusi atau algoritma    1. Algoritma :  * Mulai * Deklarasi variable * Inisialisasi scenner * Input nilai * Cek kriteria * Selesai   1. algoritma: * mulai * inisialisasi * ambil input * validasi input * evaluasi nilai * selesai  1. Tuliskan kode program dan luaran 2. Beri komentar pada kode   1.1   * Scanner: Digunakan untuk membaca input dari pengguna. * nextByte(): Membaca input sebagai byte, yang memiliki batas nilai. Anda mungkin ingin mempertimbangkan untuk menggunakan nextInt() jika Anda ingin menerima nilai yang lebih besar. * Modulus (%): Operator ini digunakan untuk mendapatkan sisa dari pembagian. Dalam hal ini, digunakan untuk menentukan apakah angka genap atau ganjil.   1.2   * Impor dan Deklarasi: Mengimpor kelas Scanner untuk mendapatkan input dari pengguna dan mendeklarasikan kelas utama. * Inisialisasi Scanner: Membuat objek Scanner untuk membaca input. * Pengambilan Input: Menjelaskan setiap langkah pengambilan input dari pengguna untuk nilai tugas dan quiz, serta memanggil metode untuk validasi. * Validasi Input: Menggambarkan proses validasi input dalam metode getValidInput, termasuk penanganan kesalahan. * Evaluasi Nilai: Menggambarkan logika dalam metode evaluateGrade untuk menentukan dan menampilkan hasil berdasarkan kriteria nilai.  1. Uraikan luaran yang dihasilkan   1.1 Masukkan Angka Anda : 10  Bilangan genap  1.2 Masukkan Angka Tugas Anda : 90  Masukkan Angka Quiz Anda : 89  Masukkan Angka Quiz Anda : 98  Anda mendapatkan nilai A   1. Screenshot/ Capture potongan kode dan hasil luaran | | |
| **[Nomor 1] Kesimpulan** | | |
| Evaluasi  Apa konsekuensi dari skenario pemprograman ini?   * Batasan Tipe Data: Menggunakan nextByte() membatasi nilai yang dapat diterima (dari -128 hingga 127). Jika pengguna memasukkan nilai di luar batas ini, program akan melemparkan InputMismatchException. * Penanganan Input: Kode ini tidak memiliki penanganan kesalahan (error handling). Jika pengguna memasukkan input yang tidak valid (seperti huruf atau simbol), program akan gagal. * Keterbatasan Fungsionalitas: Program hanya membedakan antara bilangan genap dan ganjil. Fungsionalitas yang lebih kompleks mungkin diperlukan untuk aplikasi nyata. * Kinerja: Untuk penggunaan yang lebih besar, pendekatan ini mungkin tidak efisien. Misalnya, dalam aplikasi yang memerlukan pemrosesan banyak angka sekaligus. | | |
|  | | |

**Template Lembar Kerja Individu dan Kelompok**

|  |  |  |
| --- | --- | --- |
| **Nama & NPM** | **Topik:** | **Tanggal:** |
| **Syifa Ariqah Pajriyanti**  **(G1F024009)** | **IF & SWITCH JAVA** | **23 September 2024** |
| **[Nomor 2] Identifikasi Masalah:** | | |
| 1. Uraikan permasalahan dan variable   **Contoh 3:**  import java.util.Scanner;  public class SwitchBersarang {     public static void main(String[] args) {         Scanner masukData = new Scanner(System.in);         // mengambil input         System.out.print("Pilih A atau B :  ");         char data = masukData.next().charAt(0);          switch(data) {          case 'A':              System.out.print("Anda sudah rajin belajar");             break; // baris 1         case 'B':              System.out.print("  Anda perlu kurangi main game");             break; // baris 2         default:             System.out.print("  Pilihan anda diluar A atau B ");  }      }     }  **Luaran Contoh 3:**  Pilih A atau B :  A Anda sudah rajin belajar    **Contoh 4:**  import java.util.Scanner;  public class SwitchBersarang {      public static void main(String[] args) {              byte bulan;             int tahun = 2022;             int jumlahHari = 0;             System.out.print("Masukkan data bulan (dalam angka): ");             Scanner masukData = new Scanner(System.in);             bulan = masukData.nextByte();                          switch (bulan) {                 case 1: case 3:   //baris 1                     jumlahHari = 31;                     break;                 case 4: //baris 2                     jumlahHari = 30;                     break;                 case 2:                     if (tahun % 4 == 0)                         jumlahHari = 29;                     else                         jumlahHari = 28;                     break;                 default:                     System.out.println("Maaf bulan hanya sampai 12.");                     break;             }             System.out.println("Jumlah hari = " + jumlahHari);     } }  **Luaran Contoh 4:** Masukkan data bulan (dalam angka): 7 Jumlah hari = 31  **Latihan 2:**  2.1.    Cermati kode pada Contoh 3.            Hapuslah kode break; pada //baris 1, eksekusi kembali.            Kemudian hapuslah kode break; pada //baris 2, eksekusi kembali.           Analisis perbedaan hasil luaran ketika kode menggunakan break, ketika kode break baris 1 dihapus, dan ketika kode break baris 2 dihapus!            Analisa kegunaan baris kode dengan break dan kata kunci default!  Jawab:  Jika break pada baris 2 dihapus: memasukkan 'A', program akan menampilkan "Anda sudah rajin belajar" dan kemudian juga akan mengeksekusi pernyataan pada case 'B', karena tidak ada break yang menghentikan eksekusi. Output: “Anda sudah rajin belajarAnda perlu kurangi main game”  Ketika break pada baris 2 Dihapus: Jika pengguna memasukkan 'B', program akan menampilkan "Anda perlu kurangi main game" dan kemudian juga akan mengeksekusi pernyataan pada default, karena tidak ada break yang menghentikan eksekusi. Output:” Anda perlu kurangi main gamePilihan anda diluar A atau B”  Kegunaan break:  Dengan menggunakan, kita dapat mengontrol alur eksekusi dan memastikan hanya satu case yang dijalankan berdasarkan input pengguna.  Kegunaan default:  Default menangani situasi di mana nilai input tidak cocok dengan salah satu case yang ada. Ini berfungsi sebagai fallback untuk memberikan pesan kesalahan atau tindakan alternatif. Penggunaan default menjamin bahwa ada respons yang diberikan meskipun input tidak sesuai dengan yang diharapkan (dalam hal ini, bukan 'A' atau 'B').  2.2.    Cermati kode pada Contoh 4 yang menampilkan jumlah hari sesuai dengan bulannya. Namun kode tersebut baru sampai bulan ke-4. Tambahkan sampai bulan ke-12 pada baris ke-1 dan baris ke-2.  import java.util.Scanner;  public class SwitchBersarang {  public static void main(String[] args) {  byte bulan;  int tahun = 2022;  int jumlahHari = 0;  System.out.print("Masukkan data bulan (dalam angka): ");  Scanner masukData = new Scanner(System.in);  bulan = masukData.nextByte();    switch (bulan) {  case 1: case 3: case 5: case 7: case 9: case 11:  jumlahHari = 31;  break;  case 4: case 6: case 8: case 10: case 12:  jumlahHari = 30;  break;  case 2:  if (tahun % 4 == 0)  jumlahHari = 29;  else  jumlahHari = 28;  break;  default:  System.out.println("Maaf bulan hanya sampai 12.");  break;  }  System.out.println("Jumlah hari = " + jumlahHari);  }  }  2.3.    Cermati permasalahan yang dipecahkan pada Contoh 3.             Apakah masalah ini bisa diubah menjadi perintah IF?            Jika bisa, rincikan analisa Anda!            Bandingkan masalah yang dapat diselesaikan percabangan dengan IF atau SWITCH !  Jawab: ya, masalah ini dapat diubah menjadi perintah if.  Membandingkan permasalahan yang dapat diselesaikan   * Struktur: * switch: Lebih terstruktur untuk banyak kasus yang bergantung pada nilai dari satu variabel. Memungkinkan pengelompokan dan lebih mudah dibaca ketika ada banyak pilihan. * if: Fleksibel dan dapat menangani kondisi kompleks, termasuk penggunaan operator logika seperti && dan ||. * Penggunaan: * switch: Lebih efisien untuk banyak kondisi berdasarkan satu variabel, biasanya lebih cepat karena dapat dioptimalkan menjadi tabel pencarian oleh compiler. * if: Berguna untuk kondisi yang melibatkan perbandingan atau logika yang lebih kompleks, seperti rentang nilai. * Kemudahan Pembacaan: * switch: Menawarkan kejelasan ketika memiliki banyak kasus yang harus dihadapi berdasarkan satu variabel. Sering kali, lebih mudah dipahami dalam konteks pilihan yang sederhana. * if: Meskipun mungkin lebih panjang untuk ditulis, bisa lebih jelas untuk kondisi kompleks yang memerlukan logika tambahan.   2.4.    Buatlah dokumentasi gambar flowchart dari Latihan 2.1. dan Latihan 2.2!  2.1    2.2   1. Rincikan sumber informasi yang relevan (buku / webpage)  ****Video Materi 2 tentang SWITCH –****[https://www.youtube.com/watch?v=RB4nz4xkisM](http://https/www.youtube.com/watch?v=RB4nz4xkisM) | | |
| **[Nomor 2] Analisis dan Argumentasi** | | |
| 1. Uraikan rancangan solusi yang diusulkan.   Rancangan solusi untuk program yang menghitung jumlah hari dalam sebuah bulan melibatkan beberapa langkah. Program ini meminta pengguna untuk memasukkan bulan dalam angka (1 sampai 12) dan menggunakan tahun default 2022. Langkah pertama adalah mengimpor kelas Scanner untuk mengambil input. Variabel byte bulan, int tahun, dan int jumlahHari dideklarasikan. Program kemudian membaca input bulan dan menggunakan struktur switch untuk menentukan jumlah hari: bulan dengan 31 hari (Januari, Maret, Mei, Juli, Agustus, Oktober, Desember), bulan dengan 30 hari (April, Juni, September, November), dan untuk Februari, program memeriksa tahun kabisat. Jika bulan tidak valid, pesan kesalahan ditampilkan.   1. Analisis solusi, kaitkan dengan permasalahan.   solusi yang diusulkan efektif dan efisien untuk permasalahan menghitung jumlah hari dalam bulan. Namun, perlu ada penambahan validasi input dan penanganan tahun kabisat yang lebih akurat untuk meningkatkan keandalan program. Dengan perbaikan ini, program akan menjadi lebih robust dan dapat diandalkan. | | |
| **[Nomor 2] Penyusunan Algoritma dan Kode Program** | | |
| 1. Rancang desain solusi atau algoritma  * Mulai * Deklarasi variable * Tampilkan pesan * Baca input * Switch(bulan) * Tampilkan hasil * Akhiri  1. Tuliskan kode program dan luaran 2. Beri komentar pada kode  * Impor adalah menjelaskan penggunaan kelas Scanner untuk mengambil input. * Deklarasi Variabel, memberi tahu tentang setiap variabel yang digunakan dan tujuannya. * Input Pengguna untuk menjelaskan langkah meminta pengguna untuk memasukkan bulan. * Struktur Switch untuk enguraikan cara menggunakan switch untuk menentukan jumlah hari berdasarkan bulan, termasuk penanganan tahun kabisat untuk bulan Februari dan penanganan kasus default untuk input tidak valid. * Output: tampilan hasil   .   1. Uraikan luaran yang dihasilkan   Masukkan data bulan (dalam angka): 9  Jumlah hari = 31   1. Screenshot/ Capture potongan kode dan hasil luaran | | |
| **[Nomor 2] Kesimpulan** | | |
| 1. Evaluasi   Apa konsekuensi dari skenario pemprograman ini?  Konsekuensi dari skenario pemrograman ini menunjukkan pentingnya melakukan validasi input, menangani kasus khusus dengan lebih baik, dan mempertimbangkan pengalaman pengguna. Meskipun program ini sudah cukup sederhana dan berfungsi, perbaikan dan penanganan kekurangan tersebut akan meningkatkan keandalan dan kegunaan aplikasi secara keseluruhan. | | |
|  | | |