|  |  |  |
| --- | --- | --- |
| **Nama & NPM** | **Topik:** | **Tanggal:** |
| **Ariansyah Putra Ramadhan**  **G1F024080** | **If dan Switch** | **10 Oktober 2024** |

**[No. 1] Identifikasi Masalah:**

1.1. Evaluasi penyebab kesalahan dan perbaiki kode tersebut!

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.

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)

1.4. Uraikan gambar diagram flowchart dari Latihan 1.2!

Contoh 1:

import java.util.Scanner; //memanggil impor package yang membaca masukan pengguna

public class PercabanganIf {  
    public static void main(String[] args) {  
        Scanner input = new Scanner(System.in);  // membaca teks yang dimasukkan pengguna  
        System.out.print("Masukkan Angka Anda : ");  //pengguna memasukkan data  
        nilai =  masuk.nextByte();  //menyimpan masukan pengguna ke tipe data  
          
        if (nilai = 1000) {  //percabangan yang memeriksa kondisi  
        System.out.println("Seribu");  //baris kode yang dieksekusi bila benar  
                }  
        else  {  //baris kode yang dieksekusi bila kondisi tidak terpenuhi dan salah  
            System.out.println("Nilai Bukan Seribu");  
        }    }  
}

**Contoh 2:** Salin dan tempel kode program berikut ke Eclipse.

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");  
    }  
    }  
}

**[No.1] Analisis dan Argumentasi**

1.1. Evaluasi penyebab kesalahan dan perbaiki kode tersebut!

Nama variabel: menggunakan nilai, tetapi tidak ada deklarasi untuk variabel tersebut. Nilai harus dideklarasikan sebelum di mulai

Metode Scanner: menggunakan masuk.nextByte(), tetapi nama objek masuk tidak dideklarasikan. Harus menggunakan objek input yang sudah di buat.

Operator penugasan vs operator perbandingan: Pada kondisi if, menggunakan = (operator penugasan) alih-alih == (operator perbandingan). Ini harus diperbaiki agar kondisi dapat dievaluasi dengan benar.

Berikut adalah kode yang telah diperbaiki:

import java.util.Scanner; // Memanggil impor package yang membaca masukan pengguna

public class PercabanganIf {

public static void main(String[] args) {

Scanner input = new Scanner(System.in); // Membaca teks yang dimasukkan pengguna

System.out.print("Masukkan Angka Anda : "); // Pengguna memasukkan data

byte nilai = input.nextByte(); // Menyimpan masukan pengguna ke tipe data byte

if (nilai == 1000) { // Percabangan yang memeriksa kondisi

System.out.println("Seribu"); // Baris kode yang dieksekusi bila benar

} else { // Baris kode yang dieksekusi bila kondisi tidak terpenuhi

System.out.println("Nilai Bukan Seribu");

}

input.close(); // Menutup scanner untuk mencegah kebocoran sumber daya

}

}

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.

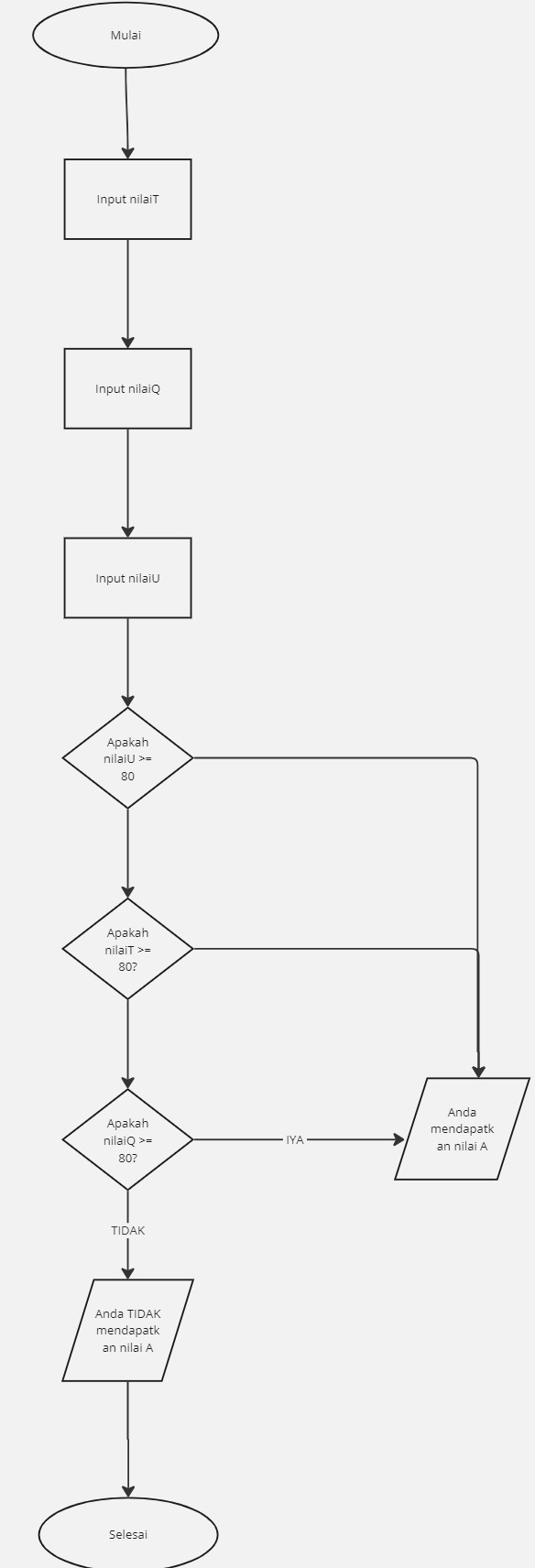
1.3 Ketiga kondisi if pada program tersebut dapat diringkas menjadi satu kondisi. Dari logika yang ada, untuk mendapatkan nilai A, semua nilai (UTS, Tugas, dan Quiz) harus lebih besar atau sama dengan 80.

Oleh karena itu, jawaban yang tepat adalah

d. IF (nilaiU >= 80 && nilaiT >= 80 && nilaiQ >= 80)

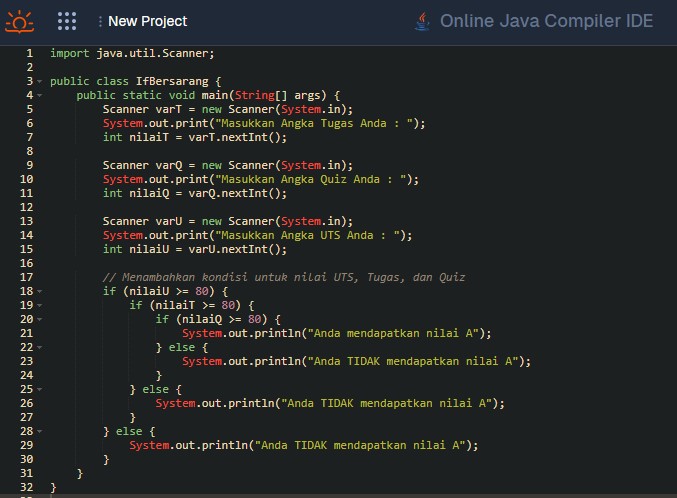
Jika ketiga syarat tersebut terpenuhi, maka Anda akan mendapatkan nilai A. Jika salah satu tidak memenuhi syarat, maka Anda tidak akan mendapatkan nilai A.

1.4 Gambar Flowchart



**[No.1 ] Penyusunan Algoritma dan Kode Program**

1. Algoritma
2. Mulai program
3. Input Nilai Tugas
4. Input Nilai Quiz
5. Input Nilai UTS
6. Cek Nilai
7. Selesai
8. Kode program dan luaran



1. Screenshot/ Capture potongan kode dan hasil luaran

Beri komentar pada kode yang di Screenshot

1. Analisa luaran yang dihasilkan   
   Program Java di atas melakukan analisis nilai yang dimasukkan pengguna untuk menentukan apakah pengguna mendapatkan nilai A berdasarkan tiga komponen: Tugas, Quiz, dan UTS.

**[No.1] Kesimpulan**

1. **Evaluasi**

Program di atas adalah sebuah progrm Java sederhana yang mengevaluasi nilai siswa berdasarkan input nilai tugas, quiz, dan UTS. Berikut adalah evaluasi dari program tersebut:

Kelebihan:

1. Sederhana dan Mudah Dipahami: Struktur kode yang menggunakan if bersarang (nested if) membuat logika penentuan nilai A menjadi jelas.
2. Input dari Pengguna: Menggunakan Scanner untuk mendapatkan input dari pengguna memungkinkan interaktivitas.
3. Kondisi yang Jelas: Memeriksa nilai UTS, tugas, dan quiz secara bertahap memberikan cara yang terstruktur untuk menentukan apakah siswa mendapatkan nilai A..

|  |  |  |
| --- | --- | --- |
| **Nama & NPM** | **Topik:** | **Tanggal:** |
| **Ariansyah Putra Ramadhan**  **G1F024080** | **If dan Switch** | **10 Oktober 2024** |

**[No. 1] Identifikasi Masalah:**

2.1. Cermati kode pada Contoh 3.   
       Evaluasi penyebab kesalahan dan perbaiki kode tersebut!  
       Hapuslah kode break; pada //baris 1, lalu eksekusi kembali.   
       Kemudian hapuslah kode break; pada //baris 2, lalu eksekusi kembali.  
       Simpulkan kegunaan break pada switch!

2.2. Cermati kode pada Contoh 4. Evaluasi apakah penulisan kode tersebut sudah efisien?   
       Apakah ada penulisan informasi yang diulangi?   
       Jika ada, rekomendasikan penulisan yang lebih tepat!

2.3.  Cermati permasalahan yang dipecahkan pada Contoh 3.   
        Apakah masalah ini bisa diubah menjadi perintah IF?   
        Jika bisa, rekomendasikan bentuk perintah IF dari Contoh 3!   
        Simpulkan perbandingan masalah yang dapat diselesaikan percabangan dengan IF atau SWITCH !

2.4.  Desain gambar flowchart dari Latihan 2.2. dan Latihan 2.3!

**Contoh 3:**Salin dan tempel kode program berikut ke Eclipse.

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 = data.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 ");   
            break;  
}    }

**Luaran Contoh 3:**  
Exception in thread "main" java.lang.Error: Unresolved compilation problems:   
    Cannot invoke next() on the primitive type char  
    Syntax error on token ":", { expected  
    Syntax error, insert ":: IdentifierOrNew" to complete ReferenceExpression  
    Syntax error, insert ":" to complete SwitchLabel  
    A cannot be resolved to a variable  
    Syntax error on token "default", } expected

    at SwitchBersarang.main(SwitchBersarang.java:8)

**Contoh 4:**Salin dan tempel kode program berikut ke Eclipse.

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: jumlahHari = 31; break;  
            case 2: if (tahun % 4 == 0) {  jumlahHari = 29; }  
                    else { jumlahHari = 28; }  
                    break;  
case 3: jumlahHari = 31; break;  
case 4: jumlahHari = 30; break;  
case 5: jumlahHari = 31; break;  
case 6: jumlahHari = 30; break;  
case 7: jumlahHari = 31; break;  
case 8: jumlahHari = 31; break;  
case 9: jumlahHari = 30; break;  
case 10: jumlahHari = 31; break;  
case 11: jumlahHari = 30; break;  
case 12: jumlahHari = 31; break;  
                    default: System.out.println("Maaf bulan hanya sampai 12.");  
                    break;  
            }  
            System.out.println("Jumlah hari = " + jumlahHari);  
}    }

**[No.1] Analisis dan Argumentasi**

2.1

Kode yang sudah di perbaiki

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) { // Mengganti ':' menjadi '{'

case 'A': // Menambahkan tanda kutip pada '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");

break;

}

}

}

Perbaikan yang Dilakukan

1. Mengganti tanda : pada switch(data) menjadi {.
2. Menambahkan tanda kutip pada case A menjadi case 'A'.
3. Menambahkan tanda : setelah default untuk memperbaiki sintaks.
4. Mengubah data menjadi masukData pada saat membaca input.

Fungsi break dalam switch adalah untuk menghentikan eksekusi dari kasus yang cocok dan keluar dari blok switch. Tanpa break, program akan terus menjalankan semua pernyataan berikutnya dalam switch, bahkan jika mereka tidak cocok dengan nilai yang diberikan. Ini dikenal sebagai "fall-through".

2.2 Evaluasi Kode

Efisiensi:

Kode menggunakan struktur switch yang jelas, tetapi ada pengulangan informasi untuk jumlah hari setiap bulan. Ini bisa disederhanakan untuk meningkatkan efisiensi dan meminimalkan potensi kesalahan.

Pengulangan Informasi:

Jumlah hari untuk bulan-bulan tertentu diulang dalam setiap kasus case di dalam switch. Ini membuat kode lebih panjang dan lebih sulit untuk diperbarui jika ada perubahan.

Rekomendasi Penulisan yang Lebih Tepat :

Program di atas menggunakan array untuk menyimpan jumlah hari dalam setiap bulan, sehingga kode menjadi lebih ringkas dan mudah dipelihara

2.3 Masalah yang dipecahkan dalam program tersebut adalah untuk memberikan respon berdasarkan input pengguna, yaitu karakter 'A' atau 'B'. Jika input tidak sesuai dengan pilihan yang ada, program akan memberi tahu bahwa pilihan tersebut tidak valid.

Pengubahan ke Bentuk IF

Program tersebut bisa diubah menjadi perintah IF. Berikut adalah bentuk alternatif menggunakan IF:

import java.util.Scanner;

public class IfBersarang {

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);

if (data == 'A') {

System.out.print("Anda sudah rajin belajar");

} else if (data == 'B') {

System.out.print("Anda perlu kurangi main game");

} else {

System.out.print("Pilihan anda diluar A atau B");

}

}

}

Perbandingan Penggunaan IF dan SWITCH

Kemudahan Pemahaman:

SWITCH lebih mudah dibaca ketika ada banyak pilihan yang harus dibandingkan.

IF dapat menjadi lebih kompleks jika ada banyak kondisi.

Kinerja:

Untuk jumlah kondisi yang besar, SWITCH dapat memiliki kinerja yang lebih baik, karena dapat dioptimalkan oleh compiler.

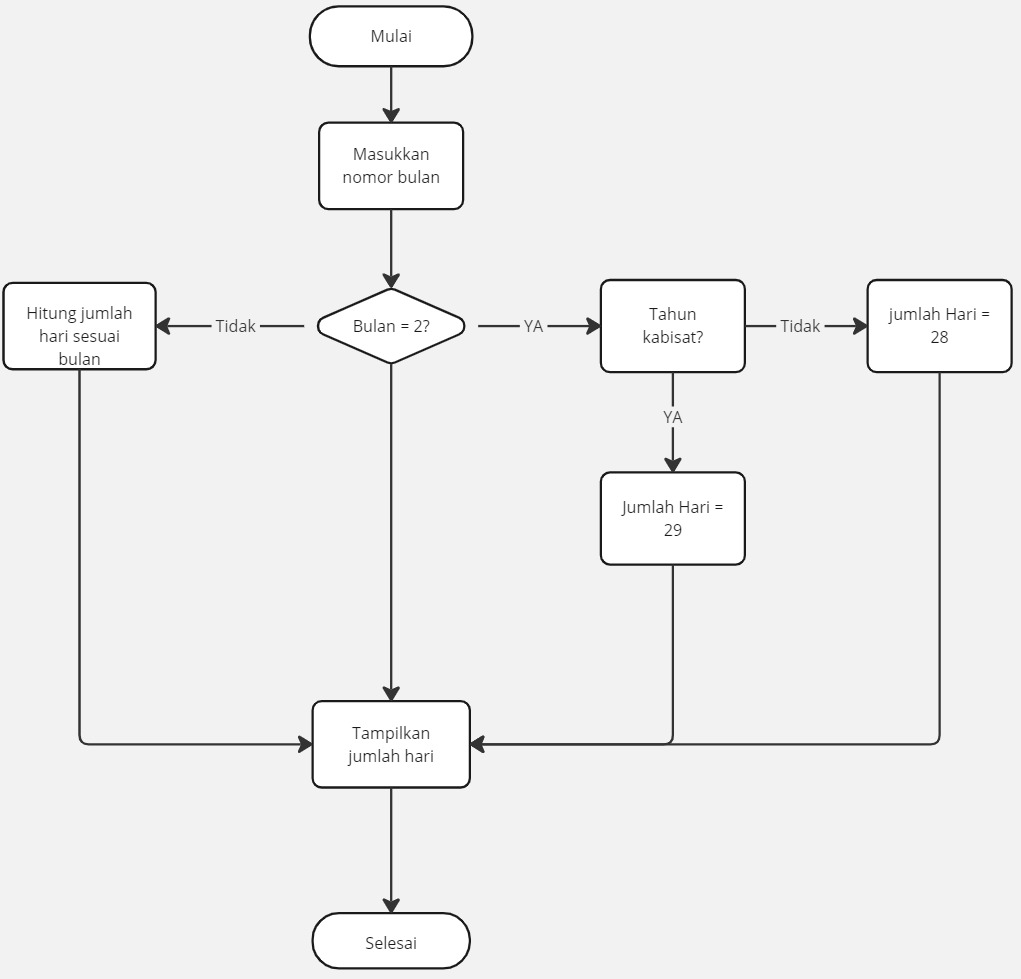
Namun, untuk sedikit pilihan, keduanya umumnya memiliki kinerja yang serupa.

Fleksibilitas:

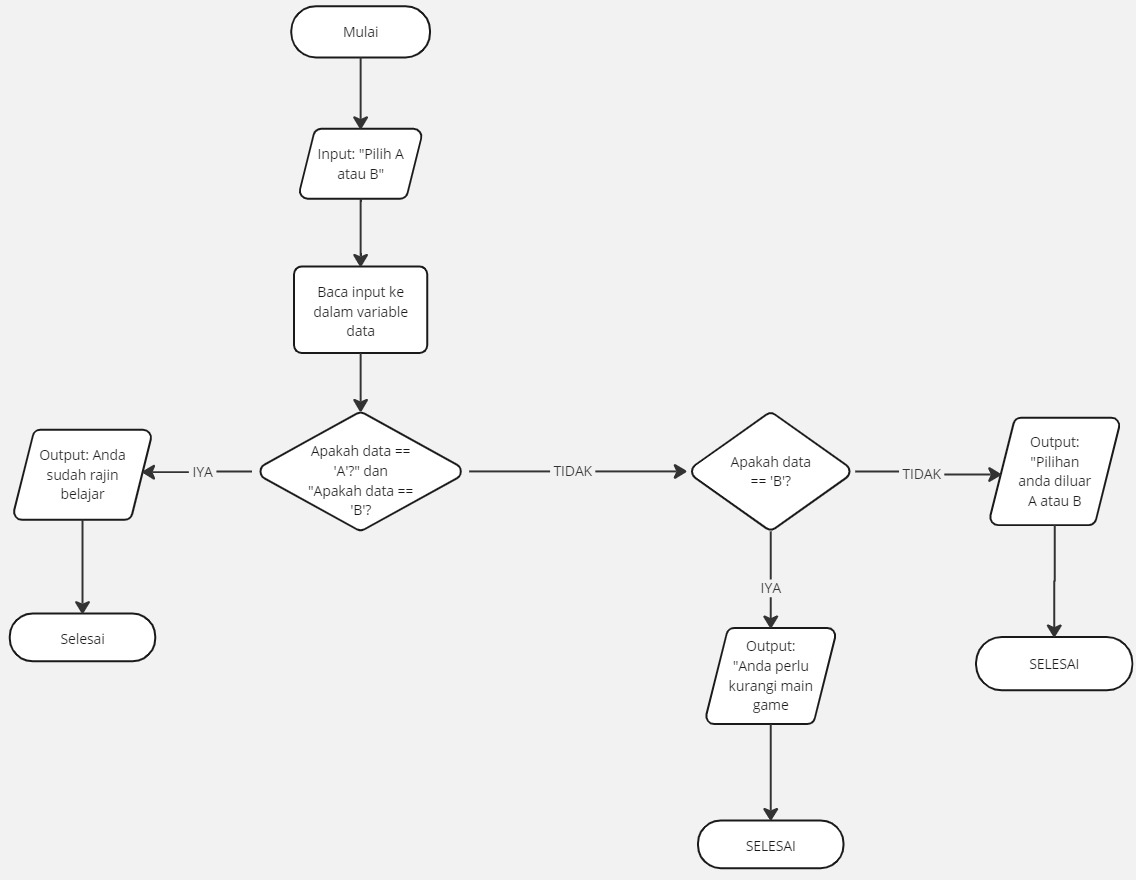
IF lebih fleksibel karena dapat menangani berbagai kondisi yang lebih kompleks (misalnya, range atau kombinasi).

SWITCH biasanya terbatas pada nilai konstan dan tidak dapat menangani kondisi kompleks dengan mudah.

2.4 Flowchart 2.2:



Flowchart 2.3:



[