Nama & NPM	Topik:	Tanggal:
Aditya Bagas Setiawan(G1F024051	IF	24/09/2024

[Nomor 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!

[Nomor 1] Analisis dan Argumentasi

1.1

```
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 penggun
        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");
        }    }
}
```

Pada contoh no 1 terdapat beberapa kesalahan yaitu

1. Pada line 7 “masuk.nextByte();” seharusnya menjadi “input.nextByte();”
2. tidak terdapat tipe data di line 7
3. penggunaan nexByte di line 7 seharusnya menggunakan nextInt karena nillai yang ingin ditampilkan adalah 1000
4. di line 9 hanya terdapat 1 sama dengan

Setelah diperbaiki kode tersebut menjadi

```java
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
        int nilai =  input.nextInt();  //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");
        }   }
}
```

1.2

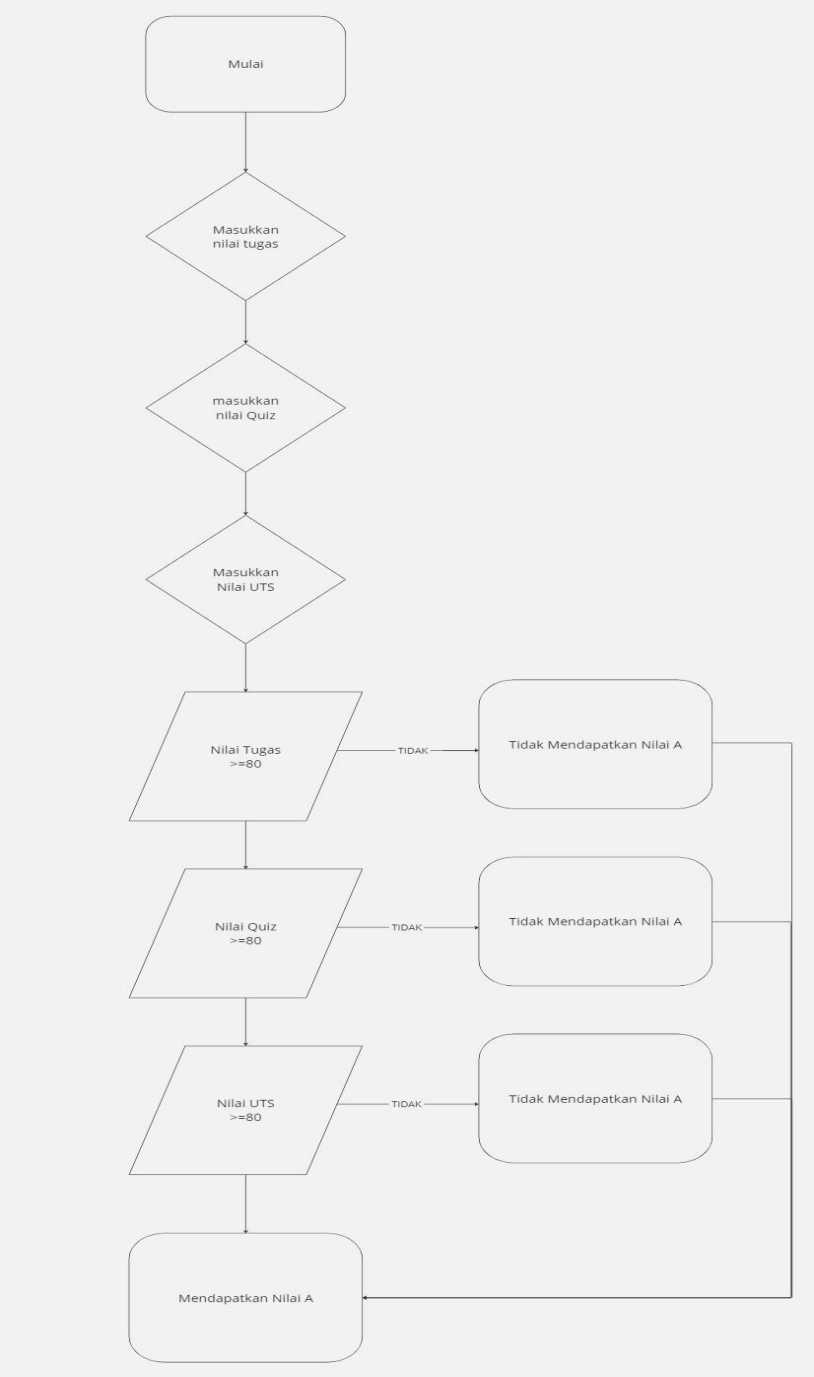
```java
import java.util.Scanner;

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 varU = new Scanner(System.in);
        System.out.print("Masukkan Angka Tugas Anda : ");
        int nilaiU =  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) {
            if(nilaiT >= 80) {
            System.out.println("Anda mendapatkan nilai A");
        }
       }
    }
         else{
         System.out.println("Anda TIDAK mendapatkan nilai A");
    }
    }
}
```

1.3

Ketiga kondisi di contoh 1.2 bisa diringkas menjadi 1,pilihan yang paling tepat pada empat pilihan tersebut adalah pilihan D

```java
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 varU = new Scanner(System.in);
        System.out.print("Masukkan Angka Tugas Anda : ");
        int nilaiU =  varT.nextByte();

        Scanner varQ = new Scanner(System.in);
        System.out.print("Masukkan Angka Quiz Anda  : ");
        int nilaiQ =  varQ.nextByte();

    if (nilaiU >= 80 && nilaiT >= 80 && nilaiQ >= 80){
        System.out.println("Anda mendapatkan nilai A");
    }
    else{
        System.out.println("Anda TIDAK mendapatkan nilai A");
    }

  }
}
```

1.4



[Nomor 1] Kesimpulan

Secara keseluruhan, penggunaan kondisi IF bersarang memberikan kontrol logika yang baik, namun penyederhanaan kondisi menjadi satu baris akan membuat kode lebih efisien dan mudah dibaca.

Nama & NPM	Topik:	Tanggal:
Aditya Bagas Setiawan(G1F024051)	SWITCH	24/09/2024

[Nomor 2] 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!

[Nomor 2] Analisis dan Argumentasi

2.1

Tidak terdapat kesalahan pada contoh 3,kode yang diberikan dapat berjalan dengan baik.

Setelah menghapus kode break 1 dan 2 berikut kode dan luaran yang dihasilkan:

```
1  import java.util.Scanner;
2
3  public class SwitchBersarang {
4      public static void main(String[] args) {
5          byte bulan;
6             int tahun = 2022;
7             int jumlahHari = 0;
8             System.out.print("Masukkan data bulan (dalam angka): ");
9             Scanner masukData = new Scanner(System.in);
10            bulan = masukData.nextByte();
11
12            switch (bulan) {
13
14 case 3: jumlahHari = 31; break;
15 case 4: jumlahHari = 30; break;
16 case 5: jumlahHari = 31; break;
17 case 6: jumlahHari = 30; break;
18 case 7: jumlahHari = 31; break;
19 case 8: jumlahHari = 31; break;
20 case 9: jumlahHari = 30; break;
21 case 10: jumlahHari = 31; break;
22 case 11: jumlahHari = 30; break;
23 case 12: jumlahHari = 31; break;
24                    default: System.out.println("Maaf bulan hanya sampai 12.");
25                    break;
26            }
27            System.out.println("Jumlah hari = " + jumlahHari);
28 }    }
```

```
Masukkan data bulan (dalam angka): 5
Jumlah hari = 31
```

break sangat penting dalam pernyataan switch untuk memastikan bahwa setelah case yang benar dieksekusi, aliran kontrol keluar dari switch dan tidak melanjutkan ke case-case berikutnya. Tanpa break, program akan melakukan fall-through (jatuh ke case berikutnya) yang sering kali tidak diinginkan, menghasilkan hasil yang salah atau tidak sesuai dengan logika.

2.2

Kode pada contoh 4 sudah benar namun belum efisien karena terdapat pengulangan kode di dalam bulan yang memiliki jumlah hari yang sama.

Berikut ini adalah penulisan kode yang lebih efisien:

```java
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 8: case 10: case 12:
                jumlahHari = 31;  // Bulan yang memiliki 31 hari
                break;
            case 4: case 6: case 9: case 11:
                jumlahHari = 30;  // Bulan yang memiliki 30 hari
                break;
            case 2:
                if (tahun % 4 == 0) {
                    jumlahHari = 29;  // Februari pada tahun kabisat
                } else {
                    jumlahHari = 28;  // Februari pada tahun biasa
                }
                break;
            default:
                System.out.println("Maaf bulan hanya sampai 12.");
                break;
        }

        if (bulan >= 1 && bulan <= 12) {
            System.out.println("Jumlah hari = " + jumlahHari);
        }

        masukData.close();
    }
}
```

2.3

Kesalahan yang terdapat pada contoh 3 antara lain:
-Kesalahan penulisan variabel data Di baris char data = data.next().charAt(0);
-kesalahan sintaks pada switch seharusnya diakhiri {, bukan :
-Case A tidak dalam tanda kutip

Berikut ini adalah kode yang sudah diperbaiki:

```java
import java.util.Scanner;

public class SwitchBersarang {
    public static void main(String[] args) {
        Scanner masukData = new Scanner(System.in);

        // Mengambil input dari pengguna
        System.out.print("Pilih A atau B: ");
        char data = masukData.next().charAt(0);

        // Switch untuk memilih berdasarkan input karakter
        switch (data) {
            case 'A': // Menggunakan tanda kutip untuk karakter
                System.out.print("Anda sudah rajin belajar");
                break; // Mengakhiri case A
            case 'B':
                System.out.print("Anda perlu kurangi main game");
                break; // Mengakhiri case B
            default:
                System.out.print("Pilihan anda diluar A atau B");
                break; // Mengakhiri default
        }

        masukData.close();
    }
}
```

Kode program tersebut bisa diubah menjadi perintah if karena memungkinkan kita untuk memeriksa kondisi yang sama dengan SWITCH

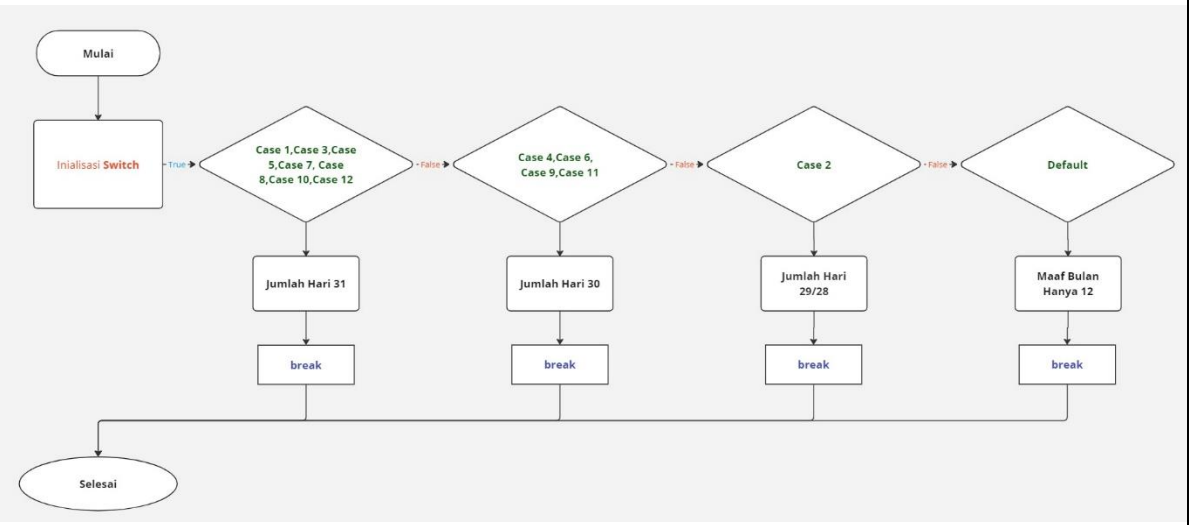
Berikut adalah kode program menggunakan if:

```java
import java.util.Scanner;

public class IfBersarang {
    public static void main(String[] args) {
        Scanner masukData = new Scanner(System.in);

        // Mengambil input dari pengguna
        System.out.print("Pilih A atau B: ");
        char data = masukData.next().charAt(0);

        // If-else untuk memeriksa input karakter
        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");
        }

        masukData.close();
    }
}
```

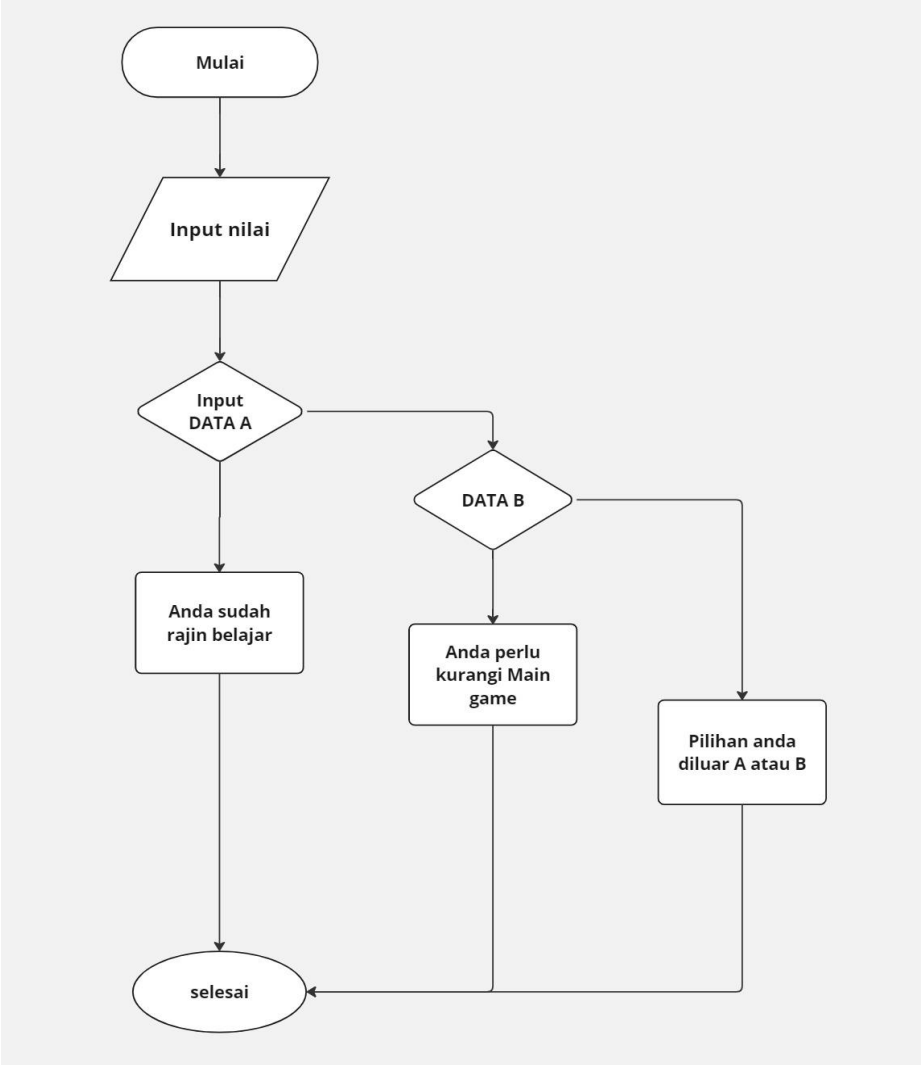
Kesimpulannya
-Switch sangat ideal digunakan untuk kondisi diskrit yang melibatkan pengecekan nilai-nilai tetap (seperti karakter 'A', 'B', atau bilangan). Dalam hal ini, switch memberikan struktur yang lebih ringkas dan lebih mudah dibaca.

- If else lebih fleksibel, cocok untuk kondisi yang lebih kompleks, seperti rentang nilai atau kondisi kombinasi. Untuk kasus sederhana seperti contoh ini, if-else juga bisa digunakan, namun lebih berguna saat logika yang lebih kompleks diperlukan.

2.4
A.flowchart Latihan 2.2



B.Flowchart Latihan 2.3



[Nomor 2] Kesimpulan

switch lebih ringkas dan efisien untuk kasus-kasus dengan pilihan diskrit tetap, sementara if-else lebih fleksibel dan dapat digunakan untuk kondisi yang lebih kompleks. Efisiensi Kode: Pengulangan informasi dalam kode dapat dihindari dengan menggabungkan kondisi yang memiliki hasil sama, baik dalam switch atau if-else. Pentingnya break dalam switch: break mengakhiri eksekusi pada case tertentu dan mencegah eksekusi berlanjut ke blok case lain yang tidak relevan.