Unit 1 IF (Latihan 1)

Nama & NPM	Topik:	Tanggal:
Dini Ramadona G1F024050	IF Dan SWITCH Java	21 September 2024

[1.1] Evaluasi penyebab kesalahan dan perbaiki kode tersebut!

Contoh 1: Salin dan tempel kode program berikut ke Eclipse.

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

Luaran Contoh 1:

```
Exception in thread "main" java.lang.Error: Unresolved compilation problems:
    nilai cannot be resolved to a variable
    masuk cannot be resolved
    nilai cannot be resolved to a variable

    at PercabanganIf.main(PercabanganIf.java:7)
```

Penyebab kesalahan pada kode diatas adalah:

1. Variabel nilai tidak dideklarasikan.
2. masuk.nextByte() seharusnya input.nextInt() karena Scanner diberi nama input.
3. Kondisi if menggunakan = (assignment) bukan == (perbandingan).

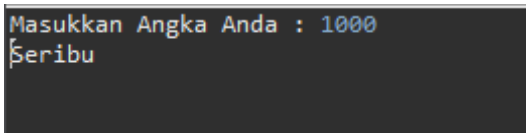
Perbaikan yang dilakukan:

1. Mendeklarasikan nilai sebagai int.
2. Menggunakan input.nextInt() untuk membaca input.
3. Mengubah kondisi if menjadi nilai == 1000.

Kode program yang sudah diperbaiki:

```
1  package com.dini;
2
3  import java.util.Scanner;
4
5  public class PercabanganIf {
6      public static void main(String[] args) {
7          Scanner input = new Scanner(System.in);
8          System.out.print("Masukkan Angka Anda : ");
9          int nilai = input.nextInt();
10
11         if (nilai == 1000) {
12             System.out.println("Seribu");
13         } else {
14             System.out.println("Nilai Bukan Seribu");
15         }
16     }
17 }
```

Kode luaran yang sudah diperbaiki:



[No.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.

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");
    }
    }
}
```

1.Struktur program:
Program ini adalah sebuah kelas Java bernama IfBersarang dalam package com.dini.
Menggunakan Scanner untuk input dari pengguna.
Memiliki metode main sebagai entry point.

2.Input:
Program meminta pengguna memasukkan 3 nilai: a. Nilai Tugas (nilaiT) b. Nilai Quiz (nilaiQ) c. Nilai UTS (nilaiU)

3.Logika if bersarang:
Terdapat 3 level if bersarang yang memeriksa nilai-nilai tersebut.
Kondisi untuk mendapatkan nilai A:
nilaiU harus >= 80 DAN
nilaiT harus >= 80 DAN
nilaiQ harus >= 80

4.Output:
Jika semua kondisi terpenuhi: "Anda mendapatkan nilai A"
Jika salah satu kondisi tidak terpenuhi: "Anda TIDAK mendapatkan nilai A"

5.Analisis:
Kode ini mengimplementasikan logika "AND" menggunakan if bersarang.

Struktur if bersarang membuat kode kurang efisien dan sulit dibaca.
Dapat disederhanakan menggunakan operator logika &&.

6.Saran perbaikan:

Kode program If Bersarang yang Dimodifikasi:

```java
package com.dini;

import java.util.Scanner;

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

        System.out.print("Masukkan Angka Tugas Anda : ");
        int nilaiT = input.nextInt();

        System.out.print("Masukkan Angka Quiz Anda : ");
        int nilaiQ = input.nextInt();

        System.out.print("Masukkan Angka UTS Anda : ");
        int nilaiU = input.nextInt();

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

Kode luaran If Bersarang yang Dimodifikasi:

```
Masukkan Angka Tugas Anda : 80
Masukkan Angka Quiz Anda : 90
Masukkan Angka UTS Anda : 100
Anda mendapatkan nilai A
```

Secara keseluruhan, kode ini berfungsi untuk menentukan apakah seorang siswa mendapatkan nilai A berdasarkan tiga komponen nilai, menggunakan struktur if bersarang yang bisa dioptimalkan untuk keterbacaan dan efisiensi yang lebih baik.

[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)`

Kode program IF Bersarang:

```java
package com.dini;

import java.util.Scanner;

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

        System.out.print("Masukkan Angka Tugas Anda : ");
        int nilaiT = input.nextInt();
```

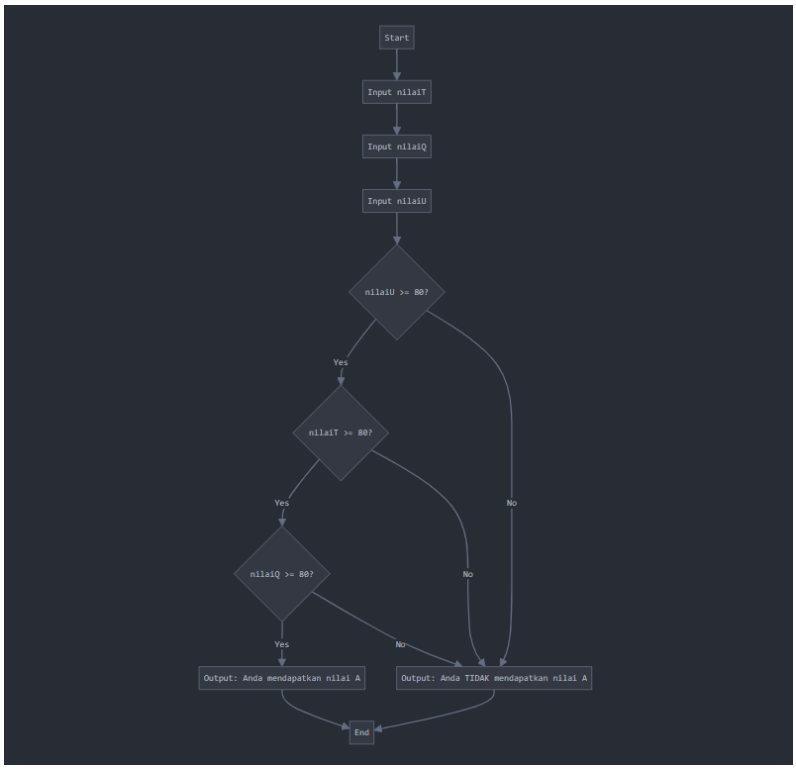
```
        System.out.print("Masukkan Angka Quiz Anda : ");
        int nilaiQ = input.nextInt();

        System.out.print("Masukkan Angka UTS Anda : ");
        int nilaiU = input.nextInt();

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

    }
}
```

Opsi yang paling tepat adalah: d. IF (nilaiU >= 80 && nilaiT >= 80 && nilaiQ >= 80)

Karena kita ingin semua nilai (UTS, Tugas, dan Quiz) harus lebih besar atau sama dengan 80 untuk mendapatkan nilai A.

[No 1.4] Uraikan gambar diagram flowchart dari Latihan 1.2!

Kali ini saya membuat Flowchart menggunakan Mermaid, Ini adalah diagram flowchart yang menggambarkan logika dari program IfBersarang yang telah dimodifikasi. Diagram ini menunjukkan alur program dari input nilai-nilai hingga output akhir, termasuk semua percabangan kondisi.



```
graph TD
    A[Start] --> B[Input nilaiT]
    B --> C[Input nilaiQ]
    C --> D[Input nilaiU]
    D --> E{nilaiU >= 80?}
    E -->|Yes| F{nilaiT >= 80?}
    E -->|No| G[Output: Anda TIDAK mendapatkan nilai A]
    F -->|Yes| H{nilaiQ >= 80?}
    F -->|No| G
    H -->|Yes| I[Output: Anda mendapatkan nilai A]
    H -->|No| G
    I --> J[End]
    G --> J
```

Flowchart ini menggambarkan proses pengambilan keputusan untuk menentukan apakah seorang siswa mendapatkan nilai A. Proses dimulai dengan input tiga nilai: nilaiT (nilai Tugas), nilaiQ (nilai Quiz), dan nilaiU (nilai UTS). Program kemudian memeriksa apakah setiap nilai memenuhi kriteria minimal 80. Jika semua nilai memenuhi kriteria, output akan menunjukkan bahwa siswa mendapatkan nilai A. Jika salah satu nilai tidak 80 makan tidak mendapatkan nilai A.

Kesimpulan:

1. Analisa

a) Kesimpulan Berdasarkan Permasalahan, Algoritma, dan Kode Program

Permasalahan:

1. Contoh 1: Kode tidak dapat berjalan karena kesalahan dalam mendeklarasikan variabel `nilai` dan menggunakan variabel `masuk` yang tidak didefinisikan. Selain itu, penggunaan operator = untuk perbandingan juga salah.
2. Contoh 2: Kode mengalami kesalahan karena variabel `nilaiU` tidak dideklarasikan, sehingga menyebabkan program tidak bisa berfungsi dengan baik.

Contoh 1:

1. Minta pengguna memasukkan angka.
2. Periksa apakah angka tersebut sama dengan 1000.
3. Tampilkan pesan sesuai hasil pemeriksaan.

Contoh 2:

4. Minta pengguna memasukkan nilai tugas dan nilai quiz.
5. Periksa nilai yang diberikan dan tentukan apakah pengguna mendapatkan nilai A.

Kode yang diberikan tidak lengkap atau memiliki kesalahan, sehingga tidak dapat dijalankan dengan baik.Kode perlu perbaikan untuk mendeklarasikan variabel dengan benar dan menggunakan operator perbandingan yang tepat.Dari analisis di atas, kedua contoh menunjukkan bahwa kesalahan umum dalam pemrograman, seperti deklarasi variabel dan penggunaan operator, dapat mengakibatkan program tidak dapat berfungsi. Perbaikan diperlukan agar program dapat berjalan sesuai yang diharapkan.

b) Dasar Alasan Pengambilan Keputusan

Alasan Pengambilan Keputusan:

1. Kebenaran Sintaksis: Menggunakan operator yang benar (misalnya, == untuk perbandingan) adalah penting agar kode dapat dieksekusi tanpa kesalahan.
2. Deklarasi Variabel: Semua variabel harus dideklarasikan sebelum digunakan. Hal ini menghindari kesalahan kompilasi yang dapat menghentikan program.

3. Keterbacaan Kode: Kode yang ditulis dengan jelas dan terstruktur lebih mudah dipahami dan dipelihara. Hal ini juga membantu dalam mendeteksi kesalahan lebih awal.
4. Pengujian Input: Memastikan bahwa program dapat menangani berbagai jenis input dari pengguna untuk meningkatkan keandalannya.

Dengan memperhatikan faktor-faktor tersebut, keputusan diambil untuk memperbaiki kode agar dapat berfungsi dengan baik dan memenuhi tujuan yang diinginkan.

Refleksi

Minggu ini, saya belajar tentang struktur kontrol if dan switch dalam pemrograman Java. Saya memahami bagaimana mengelola alur program berdasarkan kondisi tertentu dan bagaimana menggunakan pernyataan switch untuk situasi dengan banyak kemungkinan nilai. Memperbaiki kesalahan dalam kode memberikan wawasan tentang pentingnya ketelitian dalam sintaks dan logika. Saya juga belajar tentang efisiensi dalam penulisan kode dan cara menghindari pengulangan, yang sangat berguna dalam pengembangan perangkat lunak. Pengalaman ini meningkatkan kemampuan saya dalam pemrograman dan logika pemecahan masalah.

Unit 2 Switch (Latihan 2)

Nama & NPM	Topik:	Tanggal:
Dini Ramadona G1F024050	IF Dan SWITCH Java	21 September 2024

[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`!

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;
}   }
```

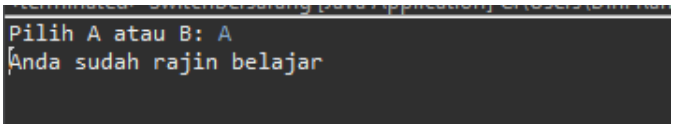
Penyebab kesalahan kode diatas adalah:

Kesalahan pada kode program contoh 3 yaitu menggunakan data sebelum mendeklarasikannya. Seharusnya menggunakan masukData untuk membaca input.Setelah switch(data), Harus menggunakan { untuk membuka blok pernyataan. Penggunaan : di sini salah.case A harus ditulis sebagai case 'A': (menambahkan tanda kutip).Dan harus menambahkan titik dua : setelah default.Perlu menambahkan kurung kurawal penutup } setelah blok switch.Berikut adalah kode program yang sudah diperbaiki:

Kode Program SwitchBersarang yang Diperbaiki

```
1  package com.dini;
2
3  import java.util.Scanner;
4
5  public class SwitchBersarang {
6      public static void main(String[] args) {
7          Scanner masukData = new Scanner(System.in);
8          // mengambil input
9          System.out.print("Pilih A atau B: ");
10         char data = masukData.next().charAt(0);
11
12         switch(data) {
13             case 'A':
14                 System.out.print("Anda sudah rajin belajar");
15                 break; // baris 1
16             case 'B':
17                 System.out.print("Anda perlu kurangi main game");
18                 break; // baris 2
19             default:
20                 System.out.print("Pilihan anda diluar A atau B");
21                 break;
22         }
23     }
24 }
```

Kode program luaran SwitchBersarang yang Diperbaiki



Perbaikan yang dilakukan:

1. Mengubah char data = data.next().charAt(0); menjadi char data = masukData.next().charAt(0);
2. Menambahkan kurung kurawal { setelah switch(data)
3. Mengubah case A menjadi case 'A' (menggunakan tanda kutip tunggal)
4. Memperbaiki indentasi dan sintaks

Sekarang, mari kita bahas penghapusan break:

- Jika Anda menghapus break pada baris 1, eksekusi akan "fall through" ke case 'B', sehingga kedua pesan akan dicetak.
- Jika Anda menghapus break pada baris 2, tidak akan ada efek yang terlihat karena ini adalah case terakhir sebelum default.

Kesimpulan:

break pada switch digunakan untuk menghentikan eksekusi switch setelah case yang sesuai ditemukan. Tanpa break, eksekusi akan berlanjut ke case berikutnya.

[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!

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);
}    }
```

Kode tersebut sudah cukup efisien, tetapi ada beberapa pengulangan informasi, terutama untuk jumlah hari yang sama (31 atau 30). Kita bisa mengoptimalkannya seperti kode program dibawah:

Kode Program Switch Bulan yang Dioptimalkan:

```
1  package com.dini;
2
3  import java.util.Scanner;
4
5  public class SwitchBulan {
6      public static void main(String[] args) {
7          byte bulan;
8          int tahun = 2022;
9          int jumlahHari;
10
11         System.out.print("Masukkan data bulan (dalam angka): ");
12         Scanner masukData = new Scanner(System.in);
13         bulan = masukData.nextByte();
14
15         switch (bulan) {
16             case 4: case 6: case 9: case 11:
17                 jumlahHari = 30;
18                 break;
19             case 2:
20                 jumlahHari = (tahun % 4 == 0) ? 29 : 28;
21                 break;
22             case 1: case 3: case 5: case 7: case 8: case 10: case 12:
23                 jumlahHari = 31;
24                 break;
25             default:
26                 jumlahHari = 0;
27                 System.out.println("Maaf bulan hanya sampai 12.");
28                 break;
29         }
30
31         if (jumlahHari != 0) {
32             System.out.println("Jumlah hari = " + jumlahHari);
33         }
34     }
35 }
```

Kode Program Luaran Switch Bulan yang Dioptimalkan:

```
Masukkan data bulan (dalam angka): 10
Jumlah hari = 31
```

[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` !

Kode Program Switch Contoh 3 Diubah ke If

```
1  package com.dini;
2
3  import java.util.Scanner;
4
5  public class IfBersarangBulan {
6      public static void main(String[] args) {
7          Scanner masukData = new Scanner(System.in);
8          // mengambil input
9          System.out.print("Pilih A atau B: ");
10         char data = masukData.next().charAt(0);
11
12         if (data == 'A') {
13             System.out.print("Anda sudah rajin belajar");
14         } else if (data == 'B') {
15             System.out.print("Anda perlu kurangi main game");
16         } else {
17             System.out.print("Pilihan anda diluar A atau B");
18         }
19     }
20 }
```

Kode Program Luaran Switch Contoh 3 Diubah ke If

```
Pilih A atau B: A
Anda sudah rajin belajar
```

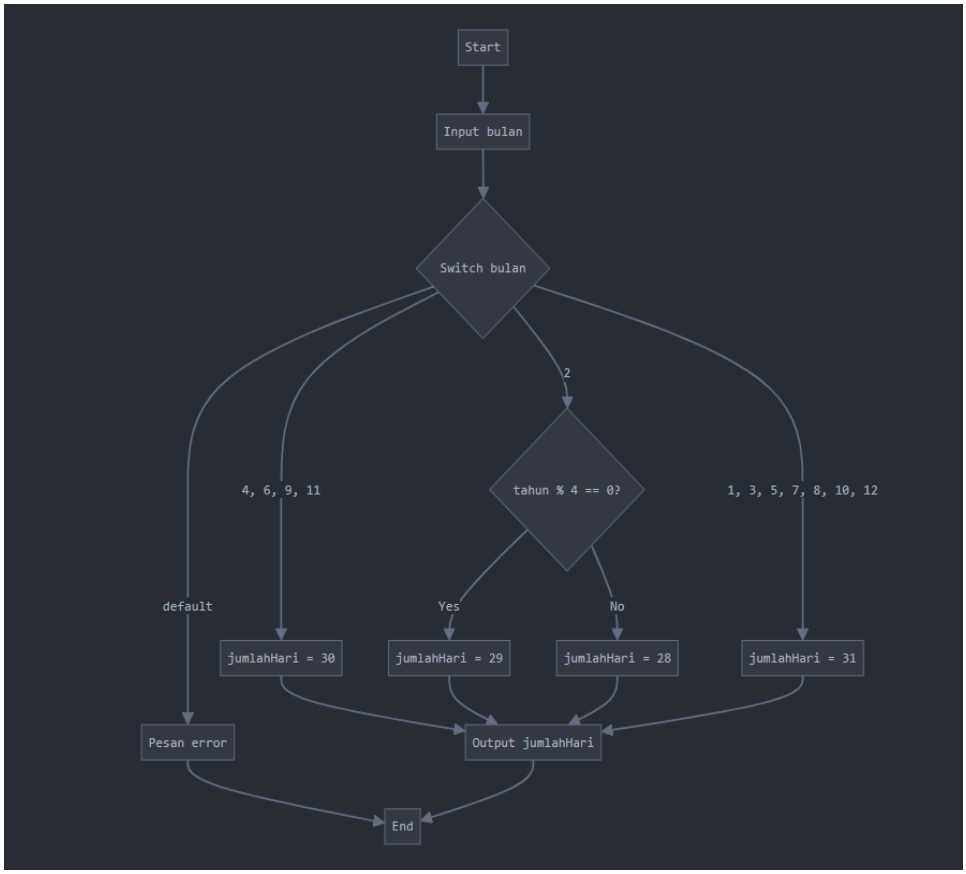
Kesimpulan perbandingan IF dan SWITCH:

- IF lebih fleksibel dan dapat menangani berbagai jenis kondisi.
- SWITCH lebih efisien untuk menangani banyak kemungkinan nilai dari satu variabel.
- IF cocok untuk kondisi yang kompleks atau rentang nilai.
- SWITCH lebih mudah dibaca untuk kasus dengan banyak pilihan yang setara.

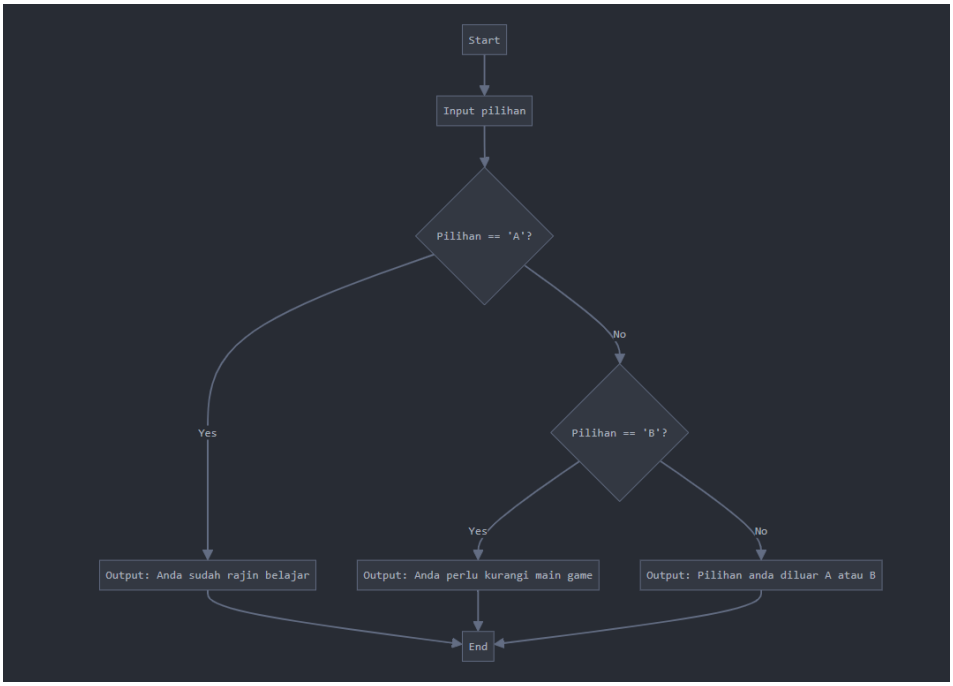
[2.4] Desain gambar flowchart dari Latihan 2.2. dan Latihan 2.3!

Kali ini saya akan membuat dua flowchart terpisah menggunakan Mermaid.

Flowchart Switch Bulan Latihan 2.2



Flowchart If Bersarang Latihan 2.3



Ini adalah flowchart untuk kedua kasus tersebut. Flowchart pertama menggambarkan logika switch untuk menentukan jumlah hari dalam bulan, sementara flowchart kedua menggambarkan logika if-else untuk memilih pesan berdasarkan input pengguna

Kesimpulan

Evaluasi:

a) Konsekuensi/Dampak dari Kode Program yang Dibuat

1. Kesalahan Eksekusi: Jika tidak ada break di dalam switch, maka program akan melakukan fall-through, yaitu melanjutkan eksekusi ke case berikutnya meskipun tidak cocok. Ini dapat menghasilkan output yang tidak diinginkan.
2. Kesalahan Input: Jika pengguna memasukkan karakter yang tidak sesuai (misalnya, selain 'A' atau 'B'), program akan memberikan umpan balik yang tidak jelas jika tidak ditangani dengan baik. Hal ini dapat membuat pengguna bingung.
3. Keterbacaan Kode: Struktur program yang baik dengan penggunaan switch atau if yang tepat membuat kode lebih mudah dipahami dan dikelola.

b) Evaluasi Input Program, Proses Perhitungan, dan Luaran yang Dihasilkan

1. Input Program:
   Program menerima input karakter ('A' atau 'B'). Input yang valid perlu ditangani, sedangkan input yang tidak valid harus diberikan umpan balik yang jelas.Sebaiknya program memvalidasi input untuk memastikan bahwa hanya karakter yang diharapkan yang diterima.
2. Proses Perhitungan:
   Dalam contoh kedua (jumlah hari dalam bulan), program melakukan perhitungan berdasarkan tahun kabisat. Penanganan tahun kabisat sudah tepat, tetapi bisa lebih efisien menggunakan array untuk menyimpan jumlah hari di setiap bulan.Logika program harus memastikan bahwa input bulan antara 1 hingga 12; jika tidak, perlu diberikan peringatan.
3. Luaran yang Dihasilkan:

Luaran dari program berupa pesan yang sesuai dengan input pengguna. Misalnya, jika pengguna memilih 'A', program mengeluarkan pesan bahwa pengguna rajin belajar. Jika memilih 'B', program menyarankan untuk mengurangi bermain game.

Luaran yang dihasilkan harus jelas dan informatif, sehingga pengguna memahami hasil dari pilihan yang mereka buat.

Rekomendasi

- Melakukan validasi input untuk memastikan bahwa pengguna memasukkan karakter yang diharapkan.
- Menyederhanakan logika perhitungan hari dalam bulan dengan menggunakan array, agar tidak terjadi redundansi.
- Menyediakan umpan balik yang lebih baik untuk input yang tidak valid, agar pengalaman pengguna menjadi lebih baik.

Refleksi

Minggu ini, saya belajar tentang struktur kontrol if dan switch dalam pemrograman Java. Saya memahami bagaimana mengelola alur program berdasarkan kondisi tertentu dan bagaimana menggunakan pernyataan switch untuk situasi dengan banyak kemungkinan nilai. Memperbaiki kesalahan dalam kode memberikan wawasan tentang pentingnya ketelitian dalam sintaks dan logika. Saya juga belajar tentang efisiensi dalam penulisan kode dan cara menghindari pengulangan, yang sangat berguna dalam pengembangan perangkat lunak. Pengalaman ini meningkatkan kemampuan saya dalam pemrograman dan logika pemecahan masalah.