Nama & NPM	Topik:	Tanggal:
1. **Abullah Hasyim Syauqi (G1F024019)** 2. **Sindi Putri utami (G1F024053)** 3. **Zaira Ayu Wandira (G1F024055)**	**IF dan SWITCH Java**	**19 september 2024**

Identifikasi Masalah:

1.1. Rekomendasikan langkah kerja dan flowchart susunan percabangan sesuai dengan data tersebut.

1.2. Desain susunan kode program untuk memeriksa nilai angka yang dimasukkan oleh pengguna ke dalam nilai abjad tertentu.

Analisis dan Argumentasi

1.1. langkah kerja sesuai dengan data

Langkah Kerja dan Flowchart Susunan Percabangan
Langkah Kerja:
Input Nilai Angka dari Pengguna:

Pengguna memasukkan nilai angka dalam rentang 0-100.
Percabangan If-Else:

Berdasarkan nilai yang dimasukkan, program memeriksa dan mengonversi nilai angka tersebut menjadi nilai abjad dan mutu:
Jika nilai $85 \leq X \leq 100$, maka A, nilai mutu = 4.0.
Jika nilai $80 \leq X < 85$, maka A-, nilai mutu = 3.75.
Jika nilai $75 \leq X < 80$, maka B+, nilai mutu = 3.5.
Jika nilai $70 \leq X < 75$, maka B, nilai mutu = 3.0.
Jika nilai $65 \leq X < 70$, maka B-, nilai mutu = 2.75.
Jika nilai $60 \leq X < 65$, maka C+, nilai mutu = 2.5.
Jika nilai < 60, maka C, nilai mutu = 2.0.
Output:

Program menampilkan nilai abjad dan nilai mutu yang sesuai dengan nilai angka yang dimasukkan pengguna.

1.2. Memeriksa nilai angka yang dimasukkan oleh pengguna ke dalam nilai abjad tertentu.

1. Struktur Kode

Kode ini adalah program sederhana yang mengonversi nilai angka menjadi nilai abjad berdasarkan rentang yang telah ditentukan. Program ini menggunakan kelas Scanner untuk menerima input dari pengguna.

- Input Nilai
  - Scanner: Program membuat objek Scanner untuk membaca input dari pengguna.
  - Pengambilan Input: Menggunakan nextDouble() untuk mengambil nilai yang dimasukkan pengguna.
- Penentuan Nilai Abjad
  - Percabangan if-else: Program menggunakan struktur percabangan untuk

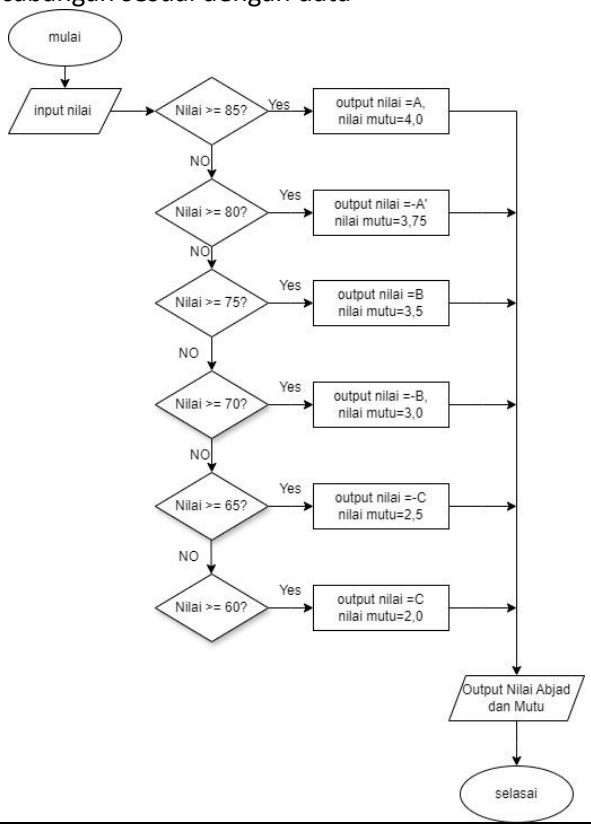
menentukan nilai abjad berdasarkan rentang nilai yang ditetapkan.

- Kondisi: Ada berbagai kondisi yang memeriksa rentang nilai:
  A untuk nilai 85-100
  A- untuk 80-84
  B+ untuk 75-79
  B untuk 70-74
  B- untuk 65-69
  C+ untuk 60-64
  C untuk 0-59
- Validasi: Jika nilai berada di luar rentang yang diharapkan, program mencetak pesan bahwa nilai tidak valid.

2. Kelebihan
   - Keterbacaan: Kode ini mudah dibaca dan dipahami berkat penamaan variabel yang jelas dan struktur yang sederhana.
   - Penanganan Input: Program secara langsung meminta input dari pengguna, memberikan interaktivitas.
   - Validasi: Memastikan bahwa input yang dimasukkan pengguna berada dalam rentang yang valid.
3. Kekurangan
   - Tidak Menangani Kesalahan Input: Program tidak memiliki mekanisme untuk menangani input yang tidak valid (misalnya, jika pengguna memasukkan huruf atau karakter selain angka). Ini dapat menyebabkan pengecualian saat menjalankan program.
   - Kurangnya Opsi Ulang: Setelah output ditampilkan, tidak ada opsi untuk pengguna mengulang proses dengan nilai baru tanpa menjalankan kembali program.
   - Rentang Nilai: Program tidak mengatur batasan pada input, seperti memaksa pengguna untuk memasukkan nilai yang lebih dari atau sama dengan 0.

Penyusunan Algoritma dan Kode Program

1.1. flowchart susunan percabangan sesuai dengan data



1.2. program kode untuk memeriksa nilai angka yang dimasukkan oleh pengguna ke dalam nilai abjad tertentu.

```
import java.util.Scanner;

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

      // Input nilai angka
      System.out.print("Masukkan nilai Anda: ");
      double nilai = scanner.nextDouble();

      // Memeriksa dan menentukan nilai abjad
      if (nilai >= 85 && nilai <= 100) {
         System.out.println("Nilai Abjad: A");
      } else if (nilai >= 80 && nilai < 85) {
         System.out.println("Nilai Abjad: A-");
      } else if (nilai >= 75 && nilai < 80) {
         System.out.println("Nilai Abjad: B+");
      } else if (nilai >= 70 && nilai < 75) {
         System.out.println("Nilai Abjad: B");
      } else if (nilai >= 65 && nilai < 70) {
         System.out.println("Nilai Abjad: B-");
      } else if (nilai >= 60 && nilai < 65) {
         System.out.println("Nilai Abjad: C+");
      } else if (nilai >= 0 && nilai < 60) {
         System.out.println("Nilai Abjad: C");
      } else {
         System.out.println("Nilai tidak valid");
      }

      scanner.close();
   }
}
```

Kesimpulan

Program ini adalah cara yang sederhana untuk mengubah nilai angka menjadi nilai abjad. Kode ini mudah dibaca dan mengajak pengguna untuk memasukkan nilai.

Identifikasi Masalah:

Konstruksikan kode program dengan IF yang menghitung nilai IPK anda untuk data mata kuliah di semester 1 (gunakan data pada tabel dibawah).

Analisis dan Argumentasi

1. Struktur Kode
   Kode ini merupakan program yang menghitung Indeks Prestasi Kumulatif (IPK) untuk beberapa mahasiswa berdasarkan daftar mata kuliah, SKS, dan nilai abjad. Kode tersebut terstruktur dengan baik dan terdiri dari beberapa komponen penting:

- Kelas HitungIPK
  Fungsi konversiNilaiAbjad: Fungsi ini mengonversi nilai abjad (seperti A, A-, B+) menjadi nilai mutu (angka) menggunakan switch. Jika nilai yang diberikan tidak dikenali, fungsi ini

akan mencetak pesan kesalahan dan mengembalikan 0.0.

- main Method
  Data Mahasiswa: Menggunakan HashMap untuk menyimpan data mahasiswa, di mana kunci adalah nama mahasiswa dan nilainya adalah daftar objek MataKuliah. Loop untuk Menghitung IPK: Untuk setiap mahasiswa, program menghitung total SKS dan total mutu, kemudian menghitung IPK dengan rumus:

  IPK=Total SKS/Total Mutu

  •
  Output: IPK dari setiap mahasiswa ditampilkan dengan format yang rapi.
- Kelas MataKuliah
  Kelas ini berfungsi untuk menyimpan data mata kuliah, termasuk nama mata kuliah, SKS, dan nilai. Ini memperjelas pengelompokan data dan memudahkan akses informasi.

2. Kelebihan

- Keterbacaan: Kode cukup mudah dibaca dan dipahami, berkat penamaan variabel dan metode yang jelas.
- Penggunaan Struktur Data yang Efisien: Menggunakan HashMap untuk menyimpan data mahasiswa memungkinkan akses cepat berdasarkan nama.
- Modularitas: Fungsi konversi nilai abjad dipisahkan, membuat kode lebih modular dan mudah untuk diuji secara terpisah.
- Handling Input yang Baik: Program menangani nilai yang tidak dikenali dengan memberikan umpan balik kepada pengguna.

3. Kekurangan

- Nilai Tidak Valid: Jika nilai abjad tidak valid, program akan mengembalikan 0.0, yang dapat mempengaruhi perhitungan IPK. Sebaiknya ditambahkan logika untuk menghentikan perhitungan atau memberikan opsi untuk mengulangi input.
- Penggunaan List.of: Metode ini membuat daftar tidak dapat dimodifikasi. Ini baik untuk menghindari perubahan tidak sengaja, tetapi jika ada kebutuhan untuk memodifikasi data, mungkin perlu menggunakan ArrayList.
- Tidak Ada Validasi Input dari Pengguna: Program ini tidak memiliki mekanisme untuk meminta input dari pengguna secara dinamis; semua data ditetapkan secara statis.

Penyusunan Algoritma dan Kode Program

```
import java.util.HashMap;
import java.util.List;
import java.util.ArrayList;

public class HitungIPK {

   // Fungsi untuk mengkonversi nilai abjad ke nilai mutu
   public static double konversiNilaiAbjad(String nilaiAbjad) {
      switch (nilaiAbjad) {
         case "A":
            return 4.0;
         case "A-":
            return 3.75;
         case "B+":
            return 3.5;
         case "B":
            return 3.0;
```

```
        case "B-":
            return 2.75;
        case "C+":
            return 2.5;
        default:
            System.out.println("Nilai '" + nilaiAbjad + "' tidak dikenali!");
            return 0.0; // Nilai tidak valid
    }
}

public static void main(String[] args) {
    // Data IPK untuk 3 orang, masing-masing berisi daftar mata kuliah, sks, dan nilai abjad
    HashMap<String, List<MataKuliah>> dataMahasiswa = new HashMap<>();

    dataMahasiswa.put("Hasyim", List.of(
            new MataKuliah("Pengantar Teknologi Informasi", 2, "A-"),
            new MataKuliah("Sistem Digital", 3, "B"),
            new MataKuliah("Komputer dan Pemrograman", 3, "B+"),
            new MataKuliah("Pengantar Sistem Multimedia", 2, "A")
    ));

    dataMahasiswa.put("Sindi", List.of(
            new MataKuliah("Pengantar Teknologi Informasi", 2, "B+"),
            new MataKuliah("Sistem Digital", 3, "A-"),
            new MataKuliah("Komputer dan Pemrograman", 3, "A"),
            new MataKuliah("Pengantar Sistem Multimedia", 2, "B")
    ));

    dataMahasiswa.put("Zaira", List.of(
            new MataKuliah("Pengantar Teknologi Informasi", 2, "C+"),
            new MataKuliah("Sistem Digital", 3, "B+"),
            new MataKuliah("Komputer dan Pemrograman", 3, "A-"),
            new MataKuliah("Pengantar Sistem Multimedia", 2, "B-")
    ));

    // Loop untuk menghitung dan menampilkan IPK setiap mahasiswa
    for (String namaMahasiswa : dataMahasiswa.keySet()) {
        List<MataKuliah> mataKuliahList = dataMahasiswa.get(namaMahasiswa);
        double totalSKS = 0;
        double totalMutu = 0;

        for (MataKuliah mk : mataKuliahList) {
            double sks = mk.getSks();
            double nilaiMutu = konversiNilaiAbjad(mk.getNilai());

            totalSKS += sks;
            totalMutu += sks * nilaiMutu;
        }

        double ipk = totalSKS > 0 ? totalMutu / totalSKS : 0;
        System.out.printf("IPK %s: %.2f%n", namaMahasiswa, ipk);
    }
}
```

```
    // Kelas MataKuliah untuk menyimpan data mata kuliah
    static class MataKuliah {
        private String nama;
        private double sks;
        private String nilai;

        public MataKuliah(String nama, double sks, String nilai) {
            this.nama = nama;
            this.sks = sks;
            this.nilai = nilai;
        }

        public double getSks() {
            return sks;
        }

        public String getNilai() {
            return nilai;
        }
    }
}
```

Kesimpulan

Program ini merupakan implementasi yang baik untuk menghitung IPK berdasarkan nilai abjad, dengan struktur yang jelas dan fungsi yang terpisah. Dengan beberapa perbaikan, terutama dalam hal penanganan input dan validasi, program ini dapat menjadi lebih robust dan user-friendly.

Refleksi

Minggu ini belajar banyak tentang bagaimana cara mengolah data dan menggunakan percabangan dalam pemrograman. Tantangan utamanya adalah membuat kode yang bisa menangani berbagai jenis input dan menghitung IPK dengan benar. Pengalaman ini membantu untuk memahami konsep dasar pemrograman.