Nama : Reyhandhika Zikri Prijadi

Nim : 254107020219

Kelas/ absen : TI_1G/ 26

Pertanyaan

1. Apakah fungsi tanpa parameter selalu harus bertipe void?
Jawab : Tidak, tidak harus. Fungsi tanpa parameter bisa saja mengembalikan nilai (return value).

- **Fungsi void** digunakan ketika fungsi tersebut hanya melakukan sebuah **aksi** atau prosedur tanpa perlu memberikan hasil perhitungan kembali ke pemanggilnya. Contohnya adalah fungsi **Menu()** yang hanya menampilkan teks ke layar.
- **Fungsi dengan tipe kembalian** (seperti **int**, **String**, **double**) digunakan ketika fungsi tersebut bertujuan untuk **menghasilkan atau menghitung sebuah nilai**.

2. Apakah daftar menu pada program kafe dapat ditampilkan tanpa menggunakan fungsi Menu()? Modifikasi kode program tersebut untuk dapat menampilkan daftar menu tanpa menggunakan fungsi!

```
    Run | Debug
    public static void main(String[] args) {

        System.out.println(x: "==== MENU RESTO KAFE ====");
        System.out.println(x: "1. KOPI HITAM - Rp 15.000");
        System.out.println(x: "2. CAPPUCCINO - Rp 20.000");
        System.out.println(x: "4. TEH TARIK - Rp 12.000");
        System.out.println(x: "3. LATTE - Rp 22.000");
        System.out.println(x: "5. ROTI BAKAR - Rp 10.000");
        System.out.println(x: "6. MIE GORENG - Rp 18.000");
        System.out.println(x: "========================");
        System.out.println(x: "Silahkan pilih menu yang anda inginkan");

    }
}
```

Jawab: Ya, bisa. Kita bisa memindahkan semua perintah **System.out.println** yang ada di dalam fungsi **Menu()** langsung ke dalam fungsi **main**.

3. Jelaskan keuntungan menggunakan fungsi Menu() dibandingkan menulis semua perintah penampilan menu langsung di dalam fungsi main.
Penggunaan fungsi **Menu()** memberikan beberapa keuntungan signifikan dalam hal pengembangan perangkat lunak:

1. **Reusability (Dapat Digunakan Kembali):** Jika di kemudian hari kita perlu menampilkan kembali daftar menu (misalnya, setelah pelanggan selesai memesan), kita cukup memanggil **Menu();** dengan satu baris kode. Tanpa fungsi, kita harus menyalin kembali semua baris **System.out.println**, yang membuat kode tidak efisien dan rentan kesalahan.
2. **Readability (Lebih Mudah Dibaca):** Fungsi **main** menjadi lebih bersih dan rapi. Pemanggilan **Menu();** jauh lebih mudah dipahami maknanya daripada blok beberapa baris perintah **System.out.println**. Ini membuat alur logika utama program di **main** menjadi lebih jernih.

3. **Maintainability (Lebih Mudah Dirawat):** Jika ada perubahan pada menu (misalnya, ada penambahan item atau perubahan harga), kita hanya perlu melakukan perubahan kode di **satu tempat**, yaitu di dalam fungsi **Menu()**. Jika tanpa fungsi, kita harus mencari dan mengganti di setiap tempat kita menuliskan menu tersebut, yang sangat tidak praktis

4. Uraikan secara sngkat alur eksekusi program ketika fungsi Menu() dipanggil dari main

(mulai dari program dijalankan sampai daftar menu tampil di layar).

Jawaban: Alur eksekusi program dapat diuraikan sebagai berikut:

1. Program dijalankan, dan program Java (JVM) mulai mengeksekusi perintah dari dalam **public static void main(String[] args)**.
2. Eksekusi berjalan secara berurutan, baris per baris di dalam fungsi **main**.
3. Ketika program menemui baris **Menu();**, alur eksekusi akan **melompat** dari **main** menuju ke badan fungsi **Menu()**.
4. Seluruh perintah yang ada di dalam fungsi **Menu()** (dalam hal ini, semua **System.out.println**) dieksekusi satu per satu hingga selesai.
5. Setelah semua perintah di dalam **Menu()** selesai, alur eksekusi **kembali** ke fungsi **main**, tepat pada baris yang berada tepat setelah pemanggilan **Menu();**.
6. Program kemudian melanjutkan eksekusi baris berikutnya di **main** hingga program selesai.

Pertanyaan

1. Apakah kegunaan parameter di dalam fungsi?
Jawaban: Kegunaan utama parameter adalah untuk menjadikan sebuah fungsi lebih **dinamis dan fleksibel**. Parameter berfungsi sebagai **masukan (input)** bagi sebuah fungsi, yang memungkinkan fungsi yang sama untuk menghasilkan output atau perilaku yang berbeda, tergantung pada nilai yang diberikan saat pemanggilan. Tanpa parameter, sebuah fungsi akan selalu melakukan hal yang persis sama setiap kali dipanggil.

2. Jelaskan mengapa pada percobaan ini fungsi Menu() menggunakan parameter

namaPelanggan dan isMember?
Jawaban: Penggunaan kedua parameter ini memiliki tujuan yang jelas dalam konteks bisnis:

- **namaPelanggan (bertipe String):** Parameter ini digunakan untuk **personalisasi** layanan. Dengan mengetahui nama pelanggan, program dapat menampilkan sapaan yang lebih personal seperti "Selamat datang, Budi!", yang dapat meningkatkan pengalaman pengguna.
- **isMember (bertipe boolean):** Parameter ini digunakan untuk **menerapkan logika bisnis yang bersyarat**. Program perlu mengetahui status keanggotaan pelanggan untuk memberikan penawaran khusus (seperti informasi diskon atau promo) yang hanya berlaku bagi member, bukan pelanggan biasa.

3. Apakah parameter sama dengan variabel? Jelaskan.
Jawaban: Keduanya mirip karena merupakan tempat penyimpanan data, namun memiliki perbedaan fundamental dalam **cara mendapatkan nilai** dan **ruang lingkup (scope)**.

4. Jelaskan bagaimana cara kerja parameter isMember pada fungsi Menu(). Apa

perbedaan output ketika isMember bernilai true dan ketika false?
Jawaban: Parameter **isMember** bekerja sebagai sebuah **flag (penanda) bertipe boolean** yang digunakan untuk mengendalikan alur program melalui struktur percabangan **if-else** di dalam fungsi **Menu()**.

- **Ketika isMember bernilai true:** Kondisi **if (isMember)** akan bernilai benar. Program akan menjalankan blok kode di dalam **if**, yang biasanya menampilkan pesan khusus untuk member, seperti "Anda adalah member, dapatkan diskon spesial!".
- **Ketika isMember bernilai false:** Kondisi **if (isMember)** akan bernilai salah. Program akan melewati blok **if** dan menjalankan blok kode di dalam **else** (jika ada), yang mungkin berisi ajakan untuk menjadi member atau tidak menampilkan pesan promosi sama sekali.

5. Apa yang akan terjadi jika memanggil fungsi Menu() tanpa menyertakan parameter

namaPelanggan dan isMember?
Jawaban: Program akan mengalami **kesalahan saat kompilasi (Compile-Time Error)**. Kompiler akan menampilkan pesan kesalahan yang menyatakan bahwa metode tersebut tidak dapat diterapkan pada argumen yang diberikan. Ini terjadi karena *signature* fungsi **Menu(String, boolean)** adalah sebuah kontrak yang wajib dipenuhi saat pemanggilan; memberikan argumen yang tidak sesuai jumlah atau tipenya akan melanggar kontrak tersebut.

6. Modifikasi kode di atas dengan menambahkan parameter baru kodePromo (String).

Jika kodePromo adalah "DISKON50", tampilkan berikan diskon 50%. Jika kodePromo

adalah "DISKON30", tampilkan berikan diskon 30%. Jika tidak ada kode promo yang

berlaku, tampilkan kode invalid.

```
}
public static void Menu(String namaPelanggan, boolean isMember, String kodePromo) {
    System.out.println("Selamat datang, " + namaPelanggan + "!");

    if (isMember) {
        System.out.println(x: "Anda adalah member, dapatkan diskon 10% untuk pembelian!");
    }

    if (kodePromo.equals(anObject: "DISKON50")) {
        System.out.println(x: "Selamat! Anda mendapatkan diskon 50%!");
    } else if (kodePromo.equals(anObject: "DISKON30")) {
        System.out.println(x: "Selamat! Anda mendapatkan diskon 30%!");
    } else {
        System.out.println(x: "Kode promo invalid.");
    }

    System.out.println(x: "==== MENU RESTO KAFE ====");
```

7. Berdasarkan fungsi Menu() di atas, jika nama pelanggan adalah "Budi", pelanggan

tersebut member, dan menggunakan kode promo "DISKON30", tuliskan satu baris

perintah pemanggilan fungsi menu yang benar.

```
public static void main(String[] args) {
    Menu(namaPelanggan: "Budi", isMember: true, kodePromo: "DISKON30");
```

8. Menurut Anda, apakah penggunaan parameter namaPelanggan dan isMember

pada fungsi Menu() membuat program lebih mudah dibaca dan dikembangkan

dibandingkan jika nilai-nilai tersebut ditulis langsung di dalam fungsi tanpa parameter?

Jelaskan alasan Anda.
Jawaban: Ya, tentu saja. Penggunaan parameter membuat program jauh lebih mudah dibaca dan dikembangkan karena beberapa alasan:

- **Keterbacaan (Readability):** Pemanggilan **Menu("Budi", true, "DISKON30");** sangat eksplisit. Siapa pun yang membaca kode dapat langsung memahami konteksnya: ini adalah pemanggilan menu untuk pelanggan "Budi", yang seorang member, dengan menggunakan kode promo "DISKON30".
- **Pengembangan (Developability):** Jika di masa depan ada perubahan logika (misalnya, ada tipe member baru dengan promo berbeda), kita hanya perlu memodifikasi logika **if-else** di dalam fungsi **Menu**. Fungsi **main** yang memanggilnya tidak perlu diubah sama sekali. Ini adalah penerapan prinsip desain modular yang sangat baik, di mana logika bisnis terpisah dari cara pemanggilannya.

9. Commit dan push hasil modifikasi Anda ke Github dengan pesan “Modifikasi

Percobaan 2”

Pertanyaan

1. Jelaskan secara singkat kapan suatu fungsi membutuhkan nilai kembalian (return

value) dan kapan fungsi tidak perlu mengembalikan nilai. Berikan minimal satu contoh

- dari program kafe pada Percobaan 3 untuk masing-masing kasus.
  Fungsi membutuhkan nilai kembalian (return value) ketika tujuannya adalah untuk melakukan perhitungan atau menghasilkan sebuah nilai yang akan digunakan oleh bagian lain dari program. Nilai ini menjadi "hasil" dari fungsi. Contoh dari program kafe: Fungsi hitungTotalHarga(pilihanMenu, banyakItem). Tujuan utama fungsi ini adalah menghitung total harga berdasarkan input yang diberikan. Nilai total ini kemudian "dikembalikan" agar bisa ditampilkan ke pelanggan atau digunakan untuk proses lebih lanjut (seperti pembayaran).
- Fungsi tidak perlu mengembalikan nilai (bertipe void) ketika tujuannya adalah untuk melakukan sebuah aksi atau prosedur tanpa perlu memberikan hasil perhitungan kembali ke pemanggilnya. Contoh dari program kafe: Fungsi tampilkanMenu(). Tugas fungsi ini adalah mencetak daftar menu ke layar. Setelah selesai mencetak, tidak ada nilai "hasil" yang perlu diberikan kembali ke fungsi main. Aksinya sudah selesai.

2. Fungsi hitungTotalHargaNoPresensi saat ini mengembalikan total harga berdasarkan pilihanMenu dan jumlahPesanan. Sebutkan tipe data nilai kembalian dan dua buah parameter yang digunakan fungsi tersebut. Jelaskan arti masing-masing

- parameter dalam konteks program kafe.
  Tipe Data Nilai Kembalian: double (atau int). Tipe double lebih direkomendasikan karena harga, terutama setelah diskon, dapat mengandung nilai desimal. Tipe data ini merepresentasikan total harga dalam bentuk numerik yang harus dibayar oleh pelanggan.
- Parameter 1: pilihanMenu (bertipe int): Parameter ini merepresentasikan kode atau identitas dari menu yang dipilih oleh pelanggan (misalnya, 1 untuk Kopi, 2 untuk Teh). Ini adalah input untuk menentukan harga dasar item.
- Parameter 2: banyakItem (bertipe int): Parameter ini merepresentasikan jumlah porsi atau unit dari menu yang dipilih yang dipesan oleh pelanggan. Ini adalah input untuk mengalikan harga dasar agar mendapatkan total harga untuk item tersebut.

3. Modifikasi kode di atas sehingga fungsi hitungTotalHargaNoPresensi dapat menerima kodePromo. Jika kodePromo adalah "DISKON50", maka mendapat diskon 50% dari totalHarga dan tampilkan diskon. Jika kodePromo adalah "DISKON30", maka mendapat diskon 30% dari totalHarga dan tampilkan diskon. Jika tidak ada kode promo yang berlaku, tampilkan kode invalid dan tidak ada pengurangan total harga totalHarga.

```
    public static int hitungTotalHarga26(int pilihanMenu, int banyakItem, String kodePromo){
    int [] hargaItems = {15000, 20000, 22000, 12000, 10000, 18000};

    int hargaTotal = hargaItems[pilihanMenu - 1]* banyakItem;
    int diskon = 0;

    if(kodePromo.equalsIgnoreCase(anotherString: "DISKON50")){
        diskon = hargaTotal*50/100;
        System.out.println("Diskon 50% : Rp " + diskon);
    }
    else if (kodePromo.equalsIgnoreCase(anotherString: "DISKON30")) {
        diskon = hargaTotal * 30 / 100;
        System.out.println("Diskon 30%: Rp " + diskon);
    } else {
        System.out.println(x: "Kode promo invalid.");
    }

    int totalBayar = hargaTotal - diskon;
    System.out.println("Subtotal untuk menu ini: Rp " + totalBayar);
    return totalBayar;

    }
}
```

4. Modifikasi kode di atas sehingga bisa memilih beberapa jenis menu berbeda serta menampilkan total keseluruhan pesanan. Bagaimana memodifikasi program sehingga pengguna dapat: memesan lebih dari satu jenis menu (misalnya menu 1 dan 3 sekaligus), dan menampilkan total keseluruhan pesanan (gabungan dari semua jenis

menu)?

```
public class kafe26n03 {
    Run | Debug
    public static void main(String[] args) {
       Menu(namaPelanggan: "Budi", isMember: true, kodePromo: "DISKON30");

        Scanner sc = new Scanner(System.in);    Resource leak: 'sc' is never closed
        int totalKeseluruhan = 0;
        char pesanlagi;

        do {
        System.out.println(x: "\nMasukkan nomor menu yang ingin anda pesan : ");
        int pilihanMenu = sc.nextInt();
        System.out.println(x: "Masukkan jumlah item yang ingin dipesan : ");
        int banyakItem = sc.nextInt();

        int totalHarga = hitungTotalHarga26(pilihanMenu, banyakItem, kodePromo: "DISKON30");
        totalKeseluruhan += totalHarga;

        System.out.println(x: "Mau tambah pesanan ? (y/n)");
        pesanlagi = sc.next().charAt(index: 0);

        }while (pesanlagi == 'y' || pesanlagi == 'Y');


       System.out.println("Total yang harus dibayar: Rp " + totalKeseluruhan);
       System.out.println(x: "Terima kasih atas pesanan Anda!");
    }
    public static void Menu(String namaPelanggan, boolean isMember, String kodePromo) {
        System.out.println("Selamat datang, " + namaPelanggan + "!");
```

```
        if (isMember) {
            System.out.println(x: "Anda adalah member, dapatkan diskon 10% untuk pembelian!");
        }

        if (kodePromo.equals(anObject: "DISKON50")) {
            System.out.println(x: "Selamat! Anda mendapatkan diskon 50%!");
        } else if (kodePromo.equals(anObject: "DISKON30")) {
            System.out.println(x: "Selamat! Anda mendapatkan diskon 30%!");
        } else {
            System.out.println(x: "Kode promo invalid.");
        }

        System.out.println(x: "==== MENU RESTO KAFE ====");
        System.out.println(x: "1. KOPI HITAM - Rp 15.000");
        System.out.println(x: "2. CAPPUCCINO - Rp 20.000");
        System.out.println(x: "4. TEH TARIK - Rp 12.000");
        System.out.println(x: "3. LATTE - Rp 22.000");
        System.out.println(x: "5. ROTI BAKAR - Rp 10.000");
        System.out.println(x: "6. MIE GORENG - Rp 18.000");
        System.out.println(x: "========================");
        System.out.println(x: "Silahkan pilih menu yang anda inginkan");
    }

    public static int hitungTotalHarga26(int pilihanMenu, int banyakItem, String kodePromo){
    int [] hargaItems = {15000, 20000, 22000, 12000, 10000, 18000};

    int hargaTotal = hargaItems[pilihanMenu - 1]* banyakItem;
    int diskon = 0;
```

```
     public static int hitungTotalHarga26(int pilihanMenu, int banyakItem, String kodePromo){
     int [] hargaItems = {15000, 20000, 22000, 12000, 10000, 18000};

     int hargaTotal = hargaItems[pilihanMenu - 1]* banyakItem;
     int diskon = 0;

     if(kodePromo.equalsIgnoreCase(anotherString: "DISKON50")){
         diskon = hargaTotal*50/100;
         System.out.println("Diskon 50% : Rp " + diskon);
     }
     else if (kodePromo.equalsIgnoreCase(anotherString: "DISKON30")) {
         diskon = hargaTotal * 30 / 100;
         System.out.println("Diskon 30%: Rp " + diskon);
     } else {
         System.out.println(x: "Kode promo invalid.");
     }

     int totalBayar = hargaTotal - diskon;
     System.out.println("Subtotal untuk menu ini: Rp " + totalBayar);
     return totalBayar;

     }

}
```

5. Commit dan push hasil modifikasi Anda ke Github dengan pesan “Modifikasi Percobaan 3”

Pertanyaan

1. Jelaskan mengapa penulisan parameter di praktikum 4 ditulis dengan String...

namaPengunjung!
Jawaban: Penulisan String... namaPengunjung adalah sintaks khusus dalam Java yang disebut Varargs (Variable-length Arguments). Alasan penggunaannya adalah untuk memberikan fleksibilitas pada fungsi. Dengan varargs, fungsi daftarPengunjung dapat menerima jumlah argumen yang berbeda-beda (nol, satu, dua, hingga tak terbatas) dengan tipe data yang sama (String). Di dalam fungsi, parameter namaPengunjung akan otomatis diperlakukan sebagai sebuah array (String[]). Ini sangat mempermudah pemanggilan fungsi tanpa harus membuat array secara manual.

2. Modifikasi method daftarPengunjung menggunakan for-each loop.

```
for(String nama : namePengunjung) {
   System.out.println("- " + nama);
}
```

3. Bisakah menggunakan dua tipe data varaargs dalam satu fungsi? Jelaskan jawaban

Anda berdasarkan aturan varargs di Java, dan berikan contohnya!
Jawaban: Tidak, kita tidak dapat menggunakan dua parameter varargs dalam satu fungsi yang sama. Java memiliki aturan tegas mengenai hal ini:

1. Sebuah fungsi hanya boleh memiliki satu parameter varargs.
2. Parameter varargs tersebut harus menjadi parameter terakhir dalam daftar parameter fungsi.

Alasan utamanya adalah untuk menghindari ambiguitas bagi kompiler. Jika sebuah fungsi memiliki dua varargs, kompiler tidak akan bisa memahami cara membagi argumen yang diberikan saat pemanggilan.

4. Jelaskan apa yang terjadi jika fungsi daftarPengunjung dipanggil tanpa argumen.

Apakah program akan error saat kompilasi, error saat dijalankan, atau tetap berjalan?

Jika tetap berjalan, bagaimana output yang dihasilkan?
Jawaban: Jika fungsi daftarPengunjung dipanggil tanpa argumen, program akan tetap berjalan tanpa mengalami error, baik saat kompilasi maupun saat dijalankan (runtime).

Penjelasannya: Fitur varargs secara inheren mendukung pemanggilan tanpa argumen. Dalam kasus ini, parameter namaPengunjung akan menjadi sebuah array String yang kosong (new String[0]).

Perulangan for-each akan mendeteksi bahwa panjang array tersebut adalah nol. Oleh karena itu, badan perulangan tidak akan pernah dieksekusi. Program akan melanjutkan ke baris berikutnya setelah loop selesai.

5. Commit dan push hasil modifikasi Anda ke Github dengan pesan “Modifikasi

Percobaan 4”