

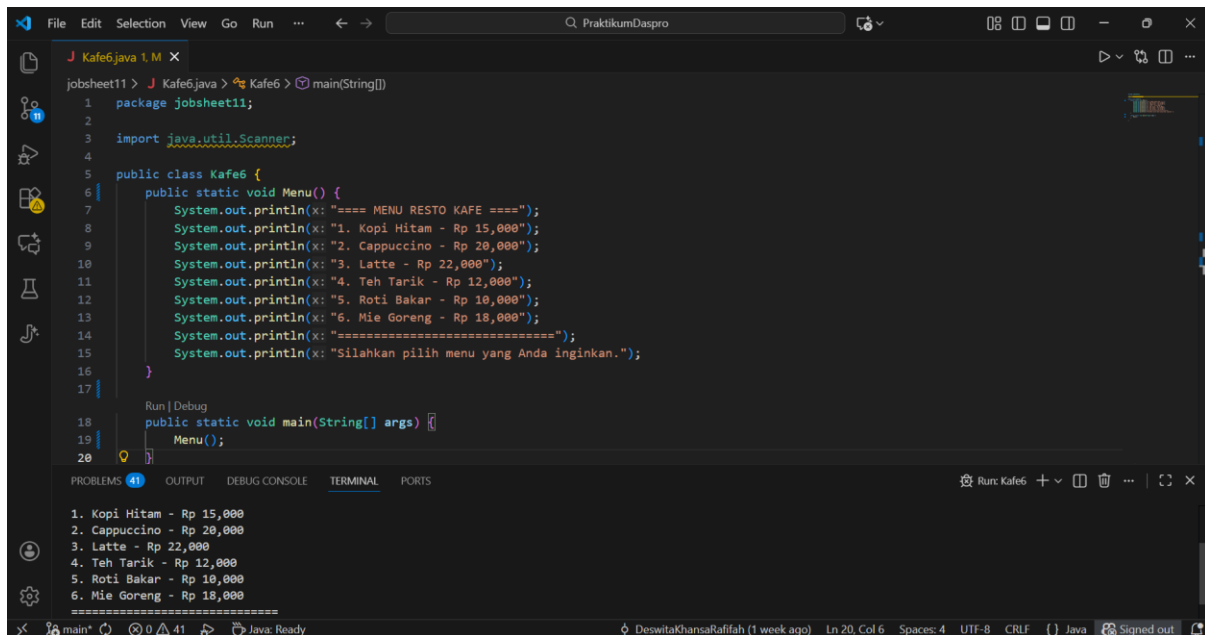
Nama : Deswita Khansa Rafifah

NIM : 254107020151

Kelas : TI_1G

JOBSHEET 11

Percobaan 1



```
1 package jobsheet11;
2
3 import java.util.Scanner;
4
5 public class Kafe {
6     public static void Menu() {
7         System.out.println(x: "==== MENU RESTO KAFE ====");
8         System.out.println(x: "1. Kopi Hitam - Rp 15,000");
9         System.out.println(x: "2. Cappuccino - Rp 20,000");
10        System.out.println(x: "3. Latte - Rp 22,000");
11        System.out.println(x: "4. Teh Tarik - Rp 12,000");
12        System.out.println(x: "5. Roti Bakar - Rp 10,000");
13        System.out.println(x: "6. Mie Goreng - Rp 18,000");
14        System.out.println(x: "=====");
15        System.out.println(x: "Silahkan pilih menu yang Anda inginkan.");
16    }
17 }
18
19 public static void main(String[] args) {
20     Menu();
21 }
```

1. Kopi Hitam - Rp 15,000
2. Cappuccino - Rp 20,000
3. Latte - Rp 22,000
4. Teh Tarik - Rp 12,000
5. Roti Bakar - Rp 10,000
6. Mie Goreng - Rp 18,000
=====

Pertanyaan

1. Apakah fungsi tanpa parameter selalu harus bertipe void?

Jawab: Tidak, fungsi tanpa parameter tidak harus selalu bertipe void. Fungsi tanpa parameter dapat memiliki tipe data kembalian apapun seperti int, double, String, boolean, dll. Hal yang membedakan adalah apakah fungsi tersebut mengembalikan nilai atau tidak. Fungsi void tidak mengembalikan nilai, sedangkan fungsi dengan tipe data lain (int, double, String, dll) harus mengembalikan nilai menggunakan keyword return.

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!

Jawab: Ya, daftar menu dapat ditampilkan tanpa menggunakan fungsi **Menu()**. Kita bisa langsung menuliskan semua perintah **System.out.println()** untuk menampilkan menu di dalam fungsi **main** tanpa membuat fungsi terpisah. Cukup pindahkan semua statement yang ada di dalam fungsi **Menu()** ke dalam fungsi **main()**.

3. Jelaskan keuntungan menggunakan fungsi **Menu()** dibandingkan menulis semua perintah penampilan menu langsung di dalam fungsi **main**.

Jawab:

Keuntungan menggunakan fungsi Menu():

- **Modular:** Kode lebih terstruktur dan terorganisir dengan baik karena setiap tugas dipisahkan dalam fungsi tersendiri
- **Reusable:** Fungsi Menu() dapat dipanggil berkali-kali di berbagai tempat dalam program tanpa perlu menulis ulang kode yang sama
- **Mudah dipelihara:** Jika ada perubahan pada menu, kita hanya perlu mengubah di satu tempat yaitu di fungsi Menu() saja
- **Readable:** Fungsi main menjadi lebih mudah dibaca karena tidak terlalu panjang dan kompleks
- **Efisien:** Mengurangi duplikasi kode dalam program

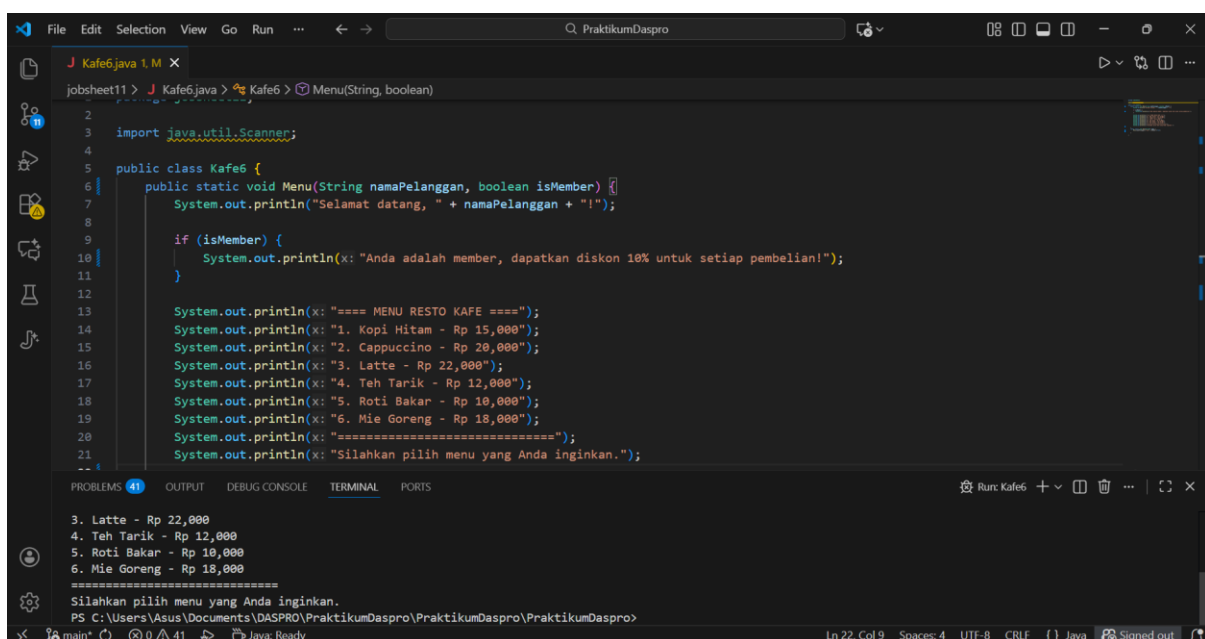
4. Uraikan secara singkat alur eksekusi program ketika fungsi Menu() dipanggil dari main (mulai dari program dijalankan sampai daftar menu tampil di layar).

Jawab:

Alur eksekusi program:

1. Program dijalankan, eksekusi dimulai dari fungsi main()
2. Di dalam fungsi main(), terdapat pemanggilan fungsi Menu()
3. Eksekusi berpindah ke fungsi Menu(), semua statement di dalam fungsi Menu() dijalankan secara berurutan dari atas ke bawah
4. Setiap System.out.println() di dalam fungsi Menu() menampilkan teks menu ke layar
5. Setelah semua statement dalam fungsi Menu() selesai dieksekusi, kontrol program kembali ke fungsi main()
6. Eksekusi dilanjutkan ke baris berikutnya setelah pemanggilan fungsi Menu() di dalam main()

Percobaan 2



The screenshot shows an IDE window titled 'PraktikumDaspro'. The main editor displays a Java file 'Kafe6.java' with the following code:

```
1  jobsheet11 > J Kafe6.java > Kafe6 > Menu(String, boolean)
2
3  import java.util.Scanner;
4
5  public class Kafe6 {
6      public static void Menu(String namaPelanggan, boolean isMember) {
7          System.out.println("Selamat datang, " + namaPelanggan + "!");
8
9          if (isMember) {
10             System.out.println("Anda adalah member, dapatkan diskon 10% untuk setiap pembelian!");
11         }
12
13         System.out.println("==== MENU RESTO KAFE ====");
14         System.out.println("1. Kopi Hitam - Rp 15,000");
15         System.out.println("2. Cappuccino - Rp 20,000");
16         System.out.println("3. Latte - Rp 22,000");
17         System.out.println("4. Teh Tarik - Rp 12,000");
18         System.out.println("5. Roti Bakar - Rp 10,000");
19         System.out.println("6. Mie Goreng - Rp 18,000");
20         System.out.println("=====");
21         System.out.println("Silahkan pilih menu yang Anda inginkan.");
22     }
23 }
```

The bottom panel shows the 'TERMINAL' output:

```
3. Latte - Rp 22,000
4. Teh Tarik - Rp 12,000
5. Roti Bakar - Rp 10,000
6. Mie Goreng - Rp 18,000
=====
Silahkan pilih menu yang Anda inginkan.
```

The status bar at the bottom indicates 'Ln 22, Col 9', 'Spaces: 4', 'UTF-8', 'CRLF', and 'Java'.

Pertanyaan

1. Apakah kegunaan parameter di dalam fungsi?

Jawab: Parameter berfungsi sebagai input atau data masukan yang akan diolah di dalam fungsi. Parameter memungkinkan fungsi menerima nilai dari luar untuk diproses, sehingga fungsi dapat bekerja dengan data yang berbeda-beda setiap kali dipanggil. Parameter membuat fungsi lebih fleksibel dan dinamis karena bisa menghasilkan output yang berbeda tergantung input yang diberikan.

2. Jelaskan mengapa pada percobaan ini fungsi **Menu()** menggunakan parameter **namaPelanggan** dan **isMember**?

Jawab:

Fungsi **Menu()** menggunakan parameter **namaPelanggan** dan **isMember** karena:

- **namaPelanggan:** Digunakan untuk menampilkan sapaan personal kepada pelanggan yang sedang memesan, membuat interaksi lebih ramah dan personal
- **isMember:** Digunakan untuk menentukan apakah pelanggan mendapat diskon member atau tidak, sehingga fungsi dapat menampilkan informasi yang berbeda untuk member dan non-member

3. Apakah parameter sama dengan variabel? Jelaskan.

Jawab: Parameter adalah jenis khusus dari variabel. Parameter dideklarasikan di dalam tanda kurung fungsi dan nilainya diberikan dari luar saat fungsi dipanggil. Sedangkan variabel biasa dideklarasikan di dalam fungsi dan nilainya diinisialisasi langsung di dalam fungsi tersebut. Keduanya termasuk variabel lokal yang hanya bisa diakses di dalam fungsi, namun sumber nilai mereka berbeda.

4. Jelaskan bagaimana cara kerja parameter **isMember** pada fungsi **Menu()**. Apa perbedaan output ketika **isMember** bernilai true dan ketika false?

Jawab:

Parameter **isMember** bertipe boolean yang digunakan untuk pengecekan kondisi menggunakan if-else. Cara kerjanya:

- Ketika **isMember** bernilai **true**: Fungsi akan menampilkan informasi bahwa pelanggan mendapat diskon member (misalnya "Anda mendapatkan diskon 10% karena member!")
- Ketika **isMember** bernilai **false**: Fungsi akan menampilkan menu tanpa informasi diskon atau menampilkan pesan bahwa pelanggan bukan member

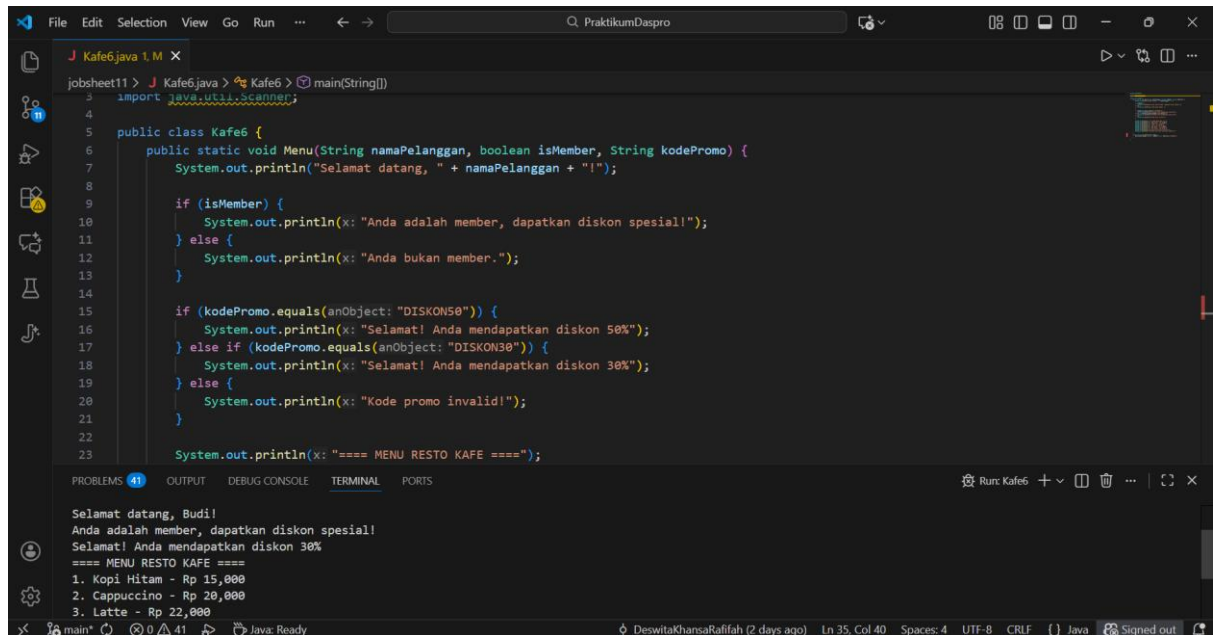
Perbedaan output terletak pada ada atau tidaknya informasi diskon member yang ditampilkan.

5. Apa yang akan terjadi jika memanggil fungsi **Menu()** tanpa menyertakan parameter **namaPelanggan** dan **isMember**?

Jawab: Program akan error pada saat kompilasi karena fungsi mengharuskan dua parameter (`namaPelanggan` dan `isMember`), tetapi pemanggilan tidak memberikan nilai apa pun.

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.

Jawab:



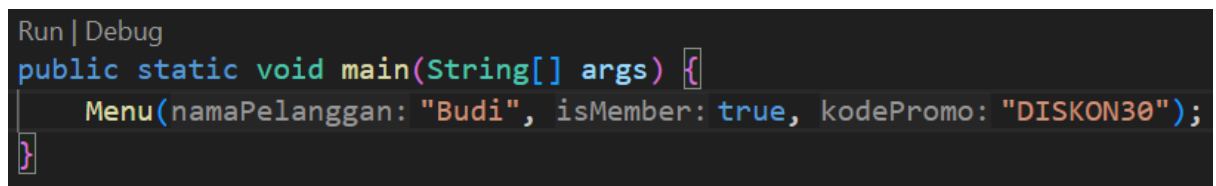
```
1  import java.util.Scanner;
2
3  public class Kafe6 {
4
5      public static void Menu(String namaPelanggan, boolean isMember, String kodePromo) {
6          System.out.println("Selamat datang, " + namaPelanggan + "!");
7
8          if (isMember) {
9              System.out.println(x: "Anda adalah member, dapatkan diskon spesial!");
10         } else {
11             System.out.println(x: "Anda bukan member.");
12         }
13
14         if (kodePromo.equals(anObject: "DISKON50")) {
15             System.out.println(x: "Selamat! Anda mendapatkan diskon 50%");
16         } else if (kodePromo.equals(anObject: "DISKON30")) {
17             System.out.println(x: "Selamat! Anda mendapatkan diskon 30%");
18         } else {
19             System.out.println(x: "Kode promo invalid!");
20         }
21
22         System.out.println(x: "==== MENU RESTO KAFE ====");
23     }
24 }
```

PROBLEMS 41 OUTPUT DEBUG CONSOLE TERMINAL PORTS

Selamat datang, Budi!
Anda adalah member, dapatkan diskon spesial!
Selamat! Anda mendapatkan diskon 30%
==== MENU RESTO KAFE ====
1. Kopi Hitam - Rp 15,000
2. Cappuccino - Rp 20,000
3. Latte - Rp 22,000

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.

Jawab:

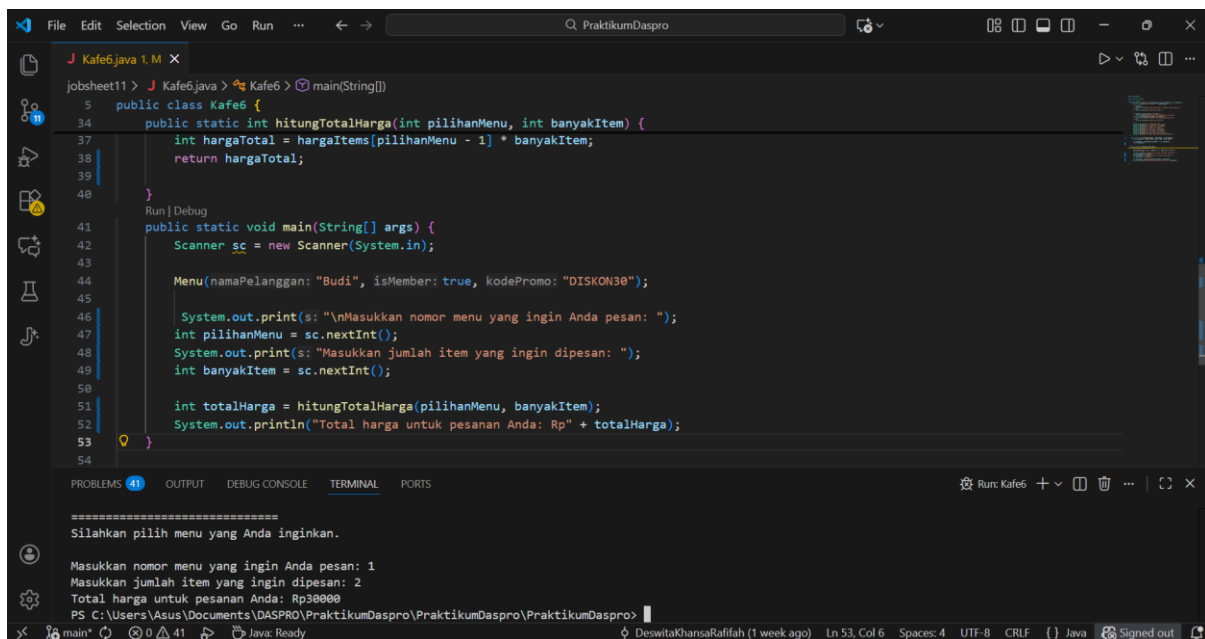


```
public static void main(String[] args) {
    Menu(namaPelanggan: "Budi", isMember: true, kodePromo: "DISKON30");
}
```

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.

Jawab: Ya. Penggunaan parameter membuat fungsi fleksibel, mudah diperbarui, dan mudah dipahami. Tanpa parameter, nilai-nilai seperti nama pelanggan atau status member harus ditulis manual di dalam fungsi, sehingga membuat program sulit dikembangkan.

Percobaan 3



```
File Edit Selection View Go Run ... < > PraktikumDaspro
J Kafe6.java 1.M x
jobsheet11 > J Kafe6.java > Kafe6 > main(String[])
5 public class Kafe6 {
34 public static int hitungTotalHarga(int pilihanMenu, int banyakItem) {
37     int hargaTotal = hargaItems[pilihanMenu - 1] * banyakItem;
38     return hargaTotal;
39 }
40
41 Run [Debug]
42 public static void main(String[] args) {
43     Scanner sc = new Scanner(System.in);
44
45     Menu(namaPelanggan: "Budi", isMember: true, kodePromo: "DISKON30");
46
47     System.out.print(s: "\nMasukkan nomor menu yang ingin Anda pesan: ");
48     int pilihanMenu = sc.nextInt();
49     System.out.print(s: "Masukkan jumlah item yang ingin dipesan: ");
50     int banyakItem = sc.nextInt();
51
52     int totalHarga = hitungTotalHarga(pilihanMenu, banyakItem);
53     System.out.println("Total harga untuk pesanan Anda: Rp" + totalHarga);
54 }
55
PROBLEMS 41 OUTPUT DEBUG CONSOLE TERMINAL PORTS
=====
Silahkan pilih menu yang Anda inginkan.
Masukkan nomor menu yang ingin Anda pesan: 1
Masukkan jumlah item yang ingin dipesan: 2
Total harga untuk pesanan Anda: Rp30000
PS C:\Users\Asus\Documents\Daspro\PraktikumDaspro>
```

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.

Jawab:

- **Fungsi membutuhkan return value** ketika hasil perhitungan atau proses di dalam fungsi perlu digunakan untuk proses selanjutnya di luar fungsi. Contoh: fungsi `hitungTotalHarga()` mengembalikan nilai total harga yang akan digunakan untuk ditampilkan atau proses pembayaran di fungsi `main`.
- **Fungsi tidak perlu return value (void)** ketika fungsi hanya melakukan aksi seperti menampilkan output ke layar tanpa perlu memberikan nilai untuk proses selanjutnya. Contoh: fungsi `Menu()` hanya menampilkan daftar menu ke layar tanpa mengembalikan nilai apapun.

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.

Jawab:

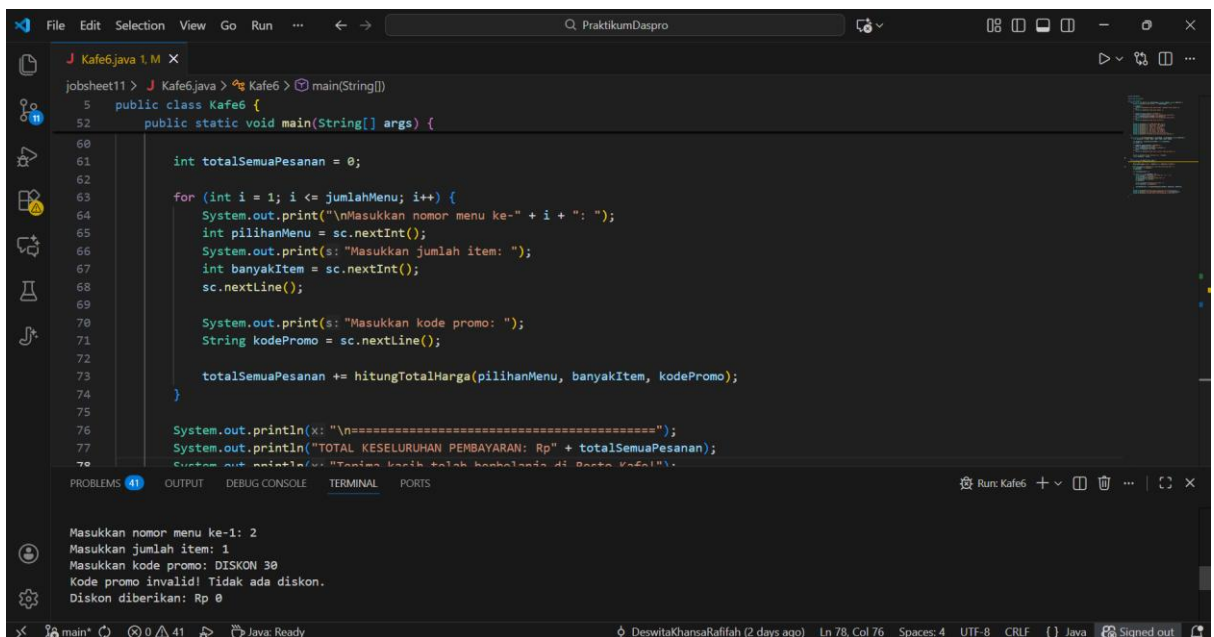
- **Tipe data nilai kembalian:** `int` atau `double`, yaitu total harga

Dua parameter:

1. **`pilihanMenu (int)`:** Menyimpan nomor menu yang dipilih pelanggan (misalnya 1 untuk Kopi, 2 untuk Teh, dst). Parameter ini digunakan untuk mengakses harga menu yang sesuai dari array harga.

2. **jumlahPesanan (int)**: Menyimpan jumlah porsi atau item yang dipesan. Parameter ini digunakan untuk mengalikan harga satuan menu dengan jumlah yang dipesan untuk mendapatkan total harga.
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**.

Jawab:

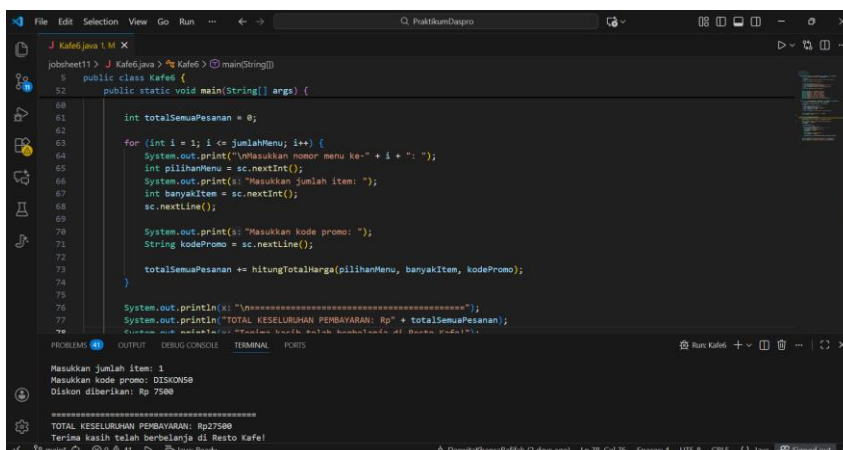


```
jobsheet11 > J Kafe6.java > Kafe6 > main(String[])
5 public class Kafe6 {
52 public static void main(String[] args) {
60
61     int totalSemuaPesanan = 0;
62
63     for (int i = 1; i <= jumlahMenu; i++) {
64         System.out.print("\nMasukkan nomor menu ke-" + i + ": ");
65         int pilihanMenu = sc.nextInt();
66         System.out.print(s: "Masukkan jumlah item: ");
67         int banyakItem = sc.nextInt();
68         sc.nextLine();
69
70         System.out.print(s: "Masukkan kode promo: ");
71         String kodePromo = sc.nextLine();
72
73         totalSemuaPesanan += hitungTotalHarga(pilihanMenu, banyakItem, kodePromo);
74     }
75
76     System.out.println(x: "\n=====");
77     System.out.println("TOTAL KESELURUHAN PEMBAYARAN: Rp" + totalSemuaPesanan);
78     System.out.println(s: "Terima kasih telah berbelanja di Resto Kafe6!");
79 }
```

Masukkan nomor menu ke-1: 2
Masukkan jumlah item: 1
Masukkan kode promo: DISKON 30
Kode promo invalid! Tidak ada diskon.
Diskon diberikan: Rp 0

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)?

Jawab:



```
jobsheet11 > J Kafe6.java > Kafe6 > main(String[])
5 public class Kafe6 {
52 public static void main(String[] args) {
60
61     int totalSemuaPesanan = 0;
62
63     for (int i = 1; i <= jumlahMenu; i++) {
64         System.out.print("\nMasukkan nomor menu ke-" + i + ": ");
65         int pilihanMenu = sc.nextInt();
66         System.out.print(s: "Masukkan jumlah item: ");
67         int banyakItem = sc.nextInt();
68         sc.nextLine();
69
70         System.out.print(s: "Masukkan kode promo: ");
71         String kodePromo = sc.nextLine();
72
73         totalSemuaPesanan += hitungTotalHarga(pilihanMenu, banyakItem, kodePromo);
74     }
75
76     System.out.println(x: "\n=====");
77     System.out.println("TOTAL KESELURUHAN PEMBAYARAN: Rp" + totalSemuaPesanan);
78     System.out.println(s: "Terima kasih telah berbelanja di Resto Kafe6!");
79 }
```

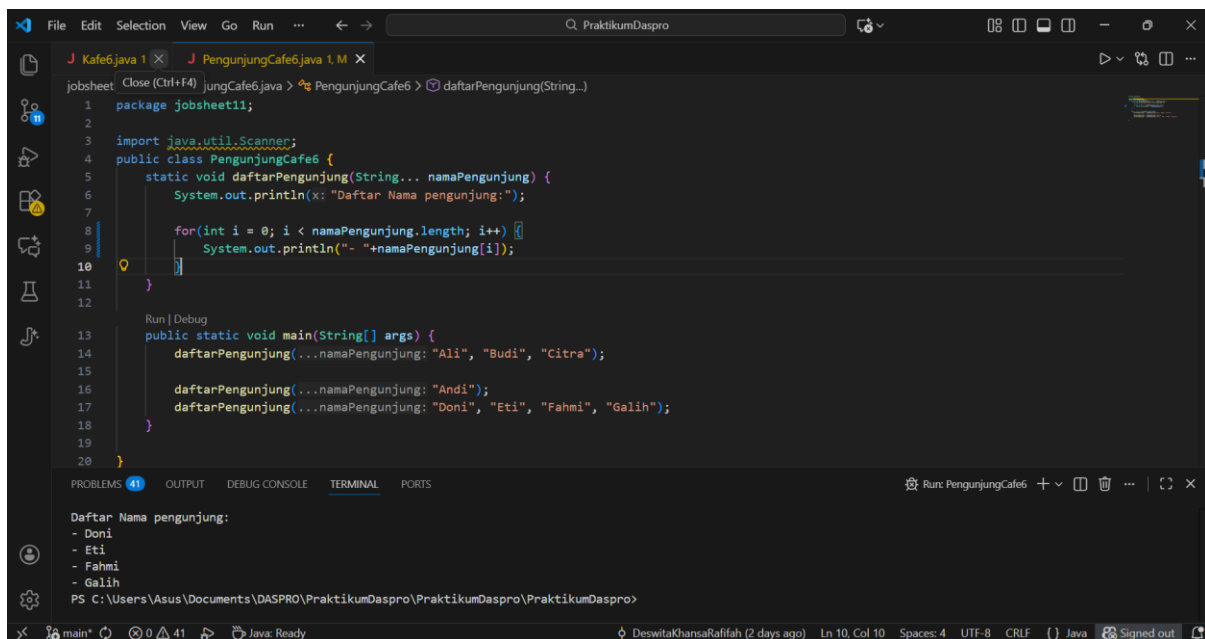
Masukkan jumlah item: 1
Masukkan kode promo: DISKON50
Diskon diberikan: Rp 7500

=====

TOTAL KESELURUHAN PEMBAYARAN: Rp7500

Terima kasih telah berbelanja di Resto Kafe6!

Percobaan 4



```
1 package jobsheet11;
2
3 import java.util.Scanner;
4 public class PengunjungCafe6 {
5     static void daftarPengunjung(String... namaPengunjung) {
6         System.out.println(x: "Daftar Nama pengunjung:");
7
8         for(int i = 0; i < namaPengunjung.length; i++) {
9             System.out.println("- " + namaPengunjung[i]);
10        }
11    }
12
13    public static void main(String[] args) {
14        daftarPengunjung(...namaPengunjung: "Ali", "Budi", "Citra");
15
16        daftarPengunjung(...namaPengunjung: "Andi");
17        daftarPengunjung(...namaPengunjung: "Doni", "Eti", "Fahmi", "Galih");
18    }
19 }
20 }
```

Daftar Nama pengunjung:

- Doni
- Eti
- Fahmi
- Galih

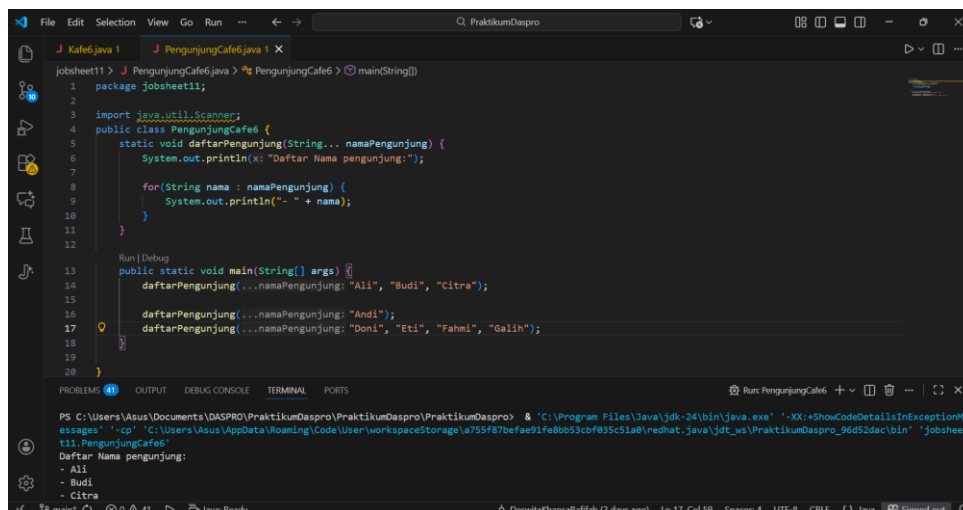
Pertanyaan

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

Jawab: Penulisan `String... namaPengunjung` menggunakan tiga titik (...) adalah sintaks untuk **Variable Arguments (Varargs)** di Java. Varargs digunakan ketika jumlah parameter yang akan diterima fungsi tidak diketahui secara pasti atau bisa berbeda-beda setiap kali fungsi dipanggil. Dengan varargs, kita dapat mengirimkan 0, 1, 2, atau lebih argumen bertipe String ke dalam fungsi. Di dalam fungsi, varargs diperlakukan sebagai array, sehingga kita bisa mengakses setiap elemen menggunakan loop. Ini membuat fungsi lebih fleksibel karena tidak perlu menentukan jumlah parameter secara tetap.

2. Modifikasi method **daftarPengunjung** menggunakan **for-each** loop.

Jawab:



```
1 package jobsheet11;
2
3 import java.util.Scanner;
4 public class PengunjungCafe6 {
5     static void daftarPengunjung(String... namaPengunjung) {
6         System.out.println(x: "Daftar Nama pengunjung:");
7
8         for(String nama : namaPengunjung) {
9             System.out.println("- " + nama);
10        }
11    }
12
13    public static void main(String[] args) {
14        daftarPengunjung(...namaPengunjung: "Ali", "Budi", "Citra");
15
16        daftarPengunjung(...namaPengunjung: "Andi");
17        daftarPengunjung(...namaPengunjung: "Doni", "Eti", "Fahmi", "Galih");
18    }
19 }
20 }
```

Daftar Nama pengunjung:

- Ali
- Budi
- Citra

3. Bisakah menggunakan dua tipe data varargs dalam satu fungsi? Jelaskan jawaban Anda berdasarkan aturan varargs di Java, dan berikan contohnya!

Jawab: Tidak bisa. Berdasarkan aturan varargs di Java, setiap fungsi hanya diperkenankan memiliki maksimal 1 (satu) varargs saja. Selain itu, varargs harus diletakkan di posisi paling akhir dalam daftar parameter jika fungsi memiliki lebih dari satu parameter.

- Contoh yang benar: `static void contoh(String nama, int... nilai)`
- Contoh yang salah: `static void contoh(int... nilai, String... nama)` **Salah** (dua varargs)

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?

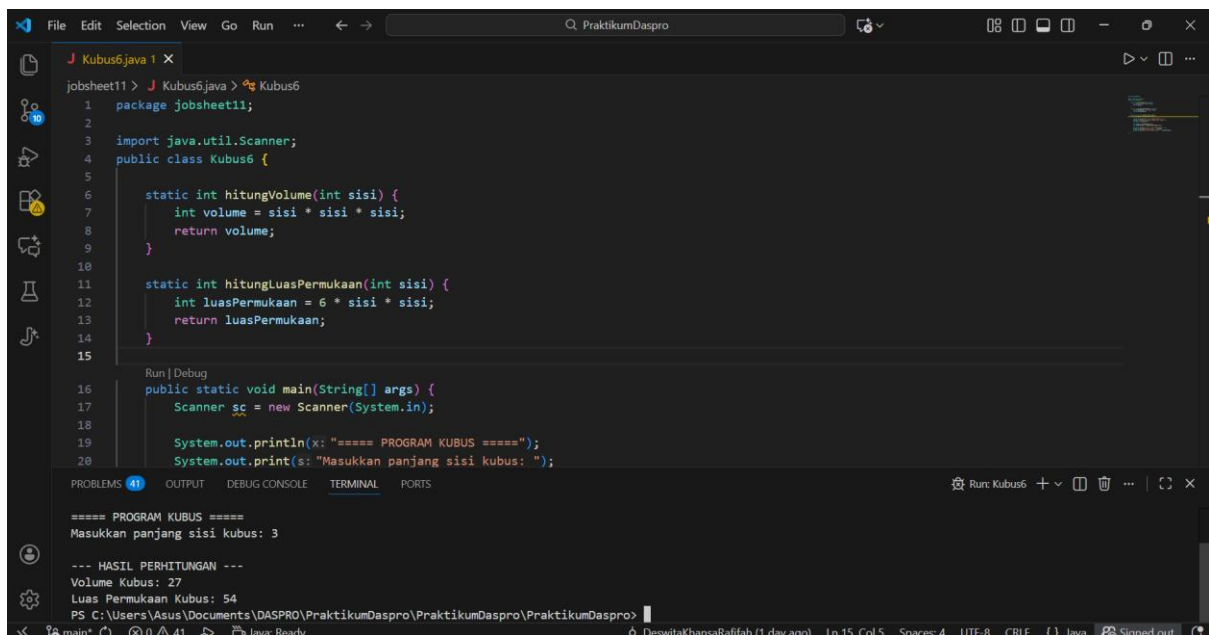
Jawab:

Program akan **tetap berjalan** tanpa error. Varargs bersifat opsional, sehingga fungsi dapat dipanggil tanpa argumen sama sekali. Jika fungsi `daftarPengunjung()` dipanggil tanpa argumen, parameter varargs akan menjadi array kosong dengan panjang 0. Output yang dihasilkan tergantung implementasi fungsi:

- Jika menggunakan loop biasa, loop tidak akan dieksekusi karena panjang array = 0
- Jika ada statement di luar loop (seperti header), tetap akan ditampilkan
- Tidak akan ada nama pengunjung yang ditampilkan karena array kosong

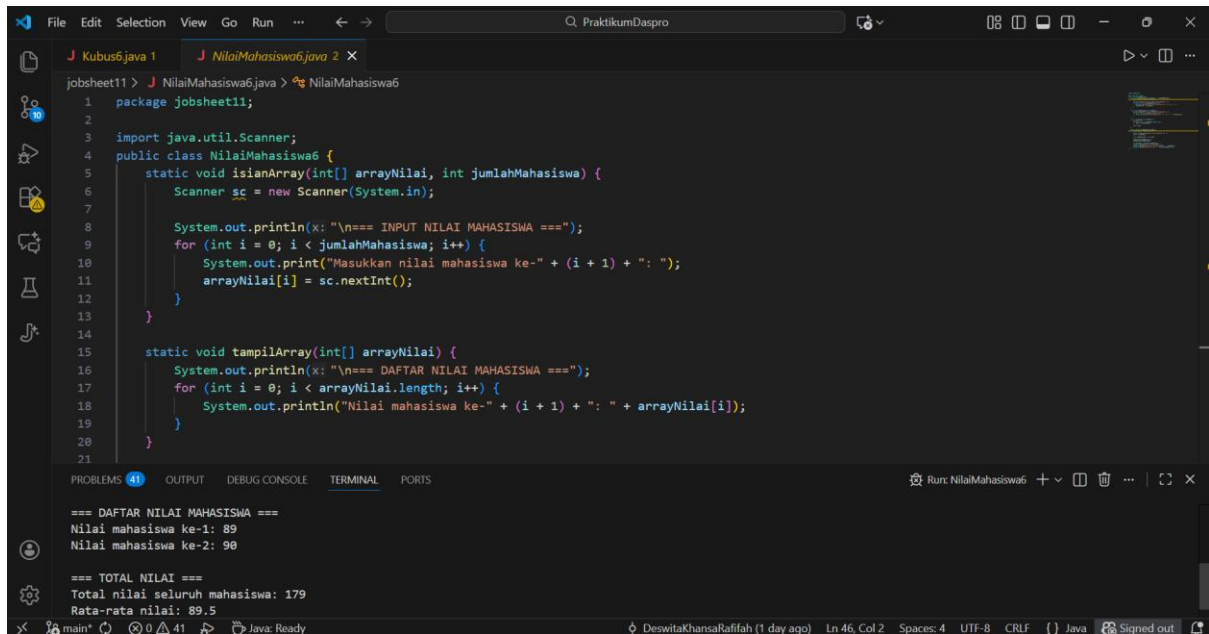
TUGAS

Tugas 1



```
File Edit Selection View Go Run ... < > PraktikumDaspro
J Kubus6.java 1 x
jobsheet11 > J Kubus6.java > Kubus6
1 package jobsheet11;
2
3 import java.util.Scanner;
4 public class Kubus6 {
5
6     static int hitungVolume(int sisi) {
7         int volume = sisi * sisi * sisi;
8         return volume;
9     }
10
11    static int hitungLuasPermukaan(int sisi) {
12        int luasPermukaan = 6 * sisi * sisi;
13        return luasPermukaan;
14    }
15
16    Run | Debug
17    public static void main(String[] args) {
18        Scanner sc = new Scanner(System.in);
19
20        System.out.println(x: "===== PROGRAM KUBUS =====");
21        System.out.print(s: "Masukkan panjang sisi kubus: ");
22
23    PROBLEMS 41 OUTPUT DEBUG CONSOLE TERMINAL PORTS
24
25    ===== PROGRAM KUBUS =====
26    Masukkan panjang sisi kubus: 3
27
28    --- HASIL PERHITUNGAN ---
29    Volume Kubus: 27
30    Luas Permukaan Kubus: 54
31    PS C:\Users\Asus\Documents\Daspro\PraktikumDaspro\PraktikumDaspro>
```


Tugas 2



The screenshot shows an IDE with a Java file named `NilaiMahasiswa6.java`. The code defines a class `NilaiMahasiswa6` with two static methods: `isianArray` and `tampilArray`. The `isianArray` method prompts the user to input grades for a list of students. The `tampilArray` method displays the entered grades and calculates the total and average grade.

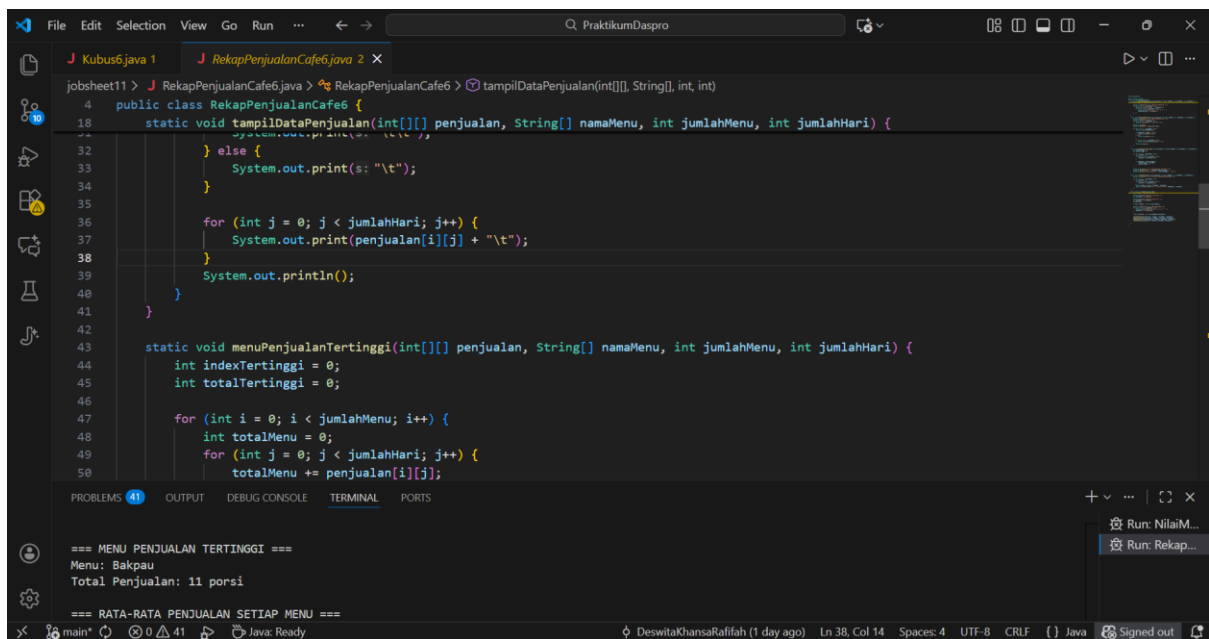
```
1 package jobsheet11;
2
3 import java.util.Scanner;
4 public class NilaiMahasiswa6 {
5     static void isianArray(int[] arrayNilai, int jumlahMahasiswa) {
6         Scanner sc = new Scanner(System.in);
7
8         System.out.println(x: "\n=== INPUT NILAI MAHASISWA ===");
9         for (int i = 0; i < jumlahMahasiswa; i++) {
10             System.out.print("Masukkan nilai mahasiswa ke-" + (i + 1) + ": ");
11             arrayNilai[i] = sc.nextInt();
12         }
13     }
14
15     static void tampilArray(int[] arrayNilai) {
16         System.out.println(x: "\n=== DAFTAR NILAI MAHASISWA ===");
17         for (int i = 0; i < arrayNilai.length; i++) {
18             System.out.println("Nilai mahasiswa ke-" + (i + 1) + ": " + arrayNilai[i]);
19         }
20     }
21 }
```

The terminal output shows the execution of the program:

```
=== DAFTAR NILAI MAHASISWA ===
Nilai mahasiswa ke-1: 89
Nilai mahasiswa ke-2: 90

=== TOTAL NILAI ===
Total nilai seluruh mahasiswa: 179
Rata-rata nilai: 89.5
```

Tugas 3



The screenshot shows an IDE with a Java file named `RekapPenjualanCafe6.java`. The code defines a class `RekapPenjualanCafe6` with two static methods: `tampilDataPenjualan` and `menuPenjualanTertinggi`. The `tampilDataPenjualan` method displays sales data for a menu over a period of days. The `menuPenjualanTertinggi` method identifies the menu with the highest total sales.

```
18 public class RekapPenjualanCafe6 {
19     static void tampilDataPenjualan(int[][] penjualan, String[] namaMenu, int jumlahMenu, int jumlahHari) {
20         System.out.println(x: "\n=== DATA PENJUALAN ===");
21         for (int i = 0; i < jumlahMenu; i++) {
22             System.out.print(namaMenu[i] + ": ");
23             for (int j = 0; j < jumlahHari; j++) {
24                 System.out.print(penjualan[i][j] + " ");
25             }
26             System.out.println();
27         }
28     }
29
30     static void menuPenjualanTertinggi(int[][] penjualan, String[] namaMenu, int jumlahMenu, int jumlahHari) {
31         int indexTertinggi = 0;
32         int totalTertinggi = 0;
33
34         for (int i = 0; i < jumlahMenu; i++) {
35             int totalMenu = 0;
36             for (int j = 0; j < jumlahHari; j++) {
37                 totalMenu += penjualan[i][j];
38             }
39             if (totalMenu > totalTertinggi) {
40                 indexTertinggi = i;
41                 totalTertinggi = totalMenu;
42             }
43         }
44         System.out.println(x: "\n=== MENU PENJUALAN TERTINGGI ===");
45         System.out.println(namaMenu[indexTertinggi]);
46         System.out.println("Total Penjualan: " + totalTertinggi + " porsi");
47     }
48 }
```

The terminal output shows the execution of the program:

```
=== MENU PENJUALAN TERTINGGI ===
Menu: Bakpau
Total Penjualan: 11 porsi

=== RATA-RATA PENJUALAN SETIAP MENU ===
```