

Nama & NPM	Topik:	Tanggal:
Meida Dinafani G1F024058	FOR dan WHILE Java	07 Oktober 2024

[No. 1] Identifikasi Masalah:

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

```
public class ContohFor{
public static void main(String[] args) {
    for (int y = 0; y <= 10; ++y) { //ubah 1
        if (y % 2 == 1) //ubah 2
            continue; //baris 1
        else if (y == 8) //ubah 3
            break; //baris 2
        else
            System.out.println(y + " ");
    }
}
```

Luaran contoh 1:

```
0
2
4
6
```

Contoh 2: Salin dan tempel kode program berikut ke Eclipse.

```
public class ForBersarang {
    public static void main(String[] args) {
        pertama:
            for( int i = 1; i < 5; i++) {
                kedua:
                    for(int j = 1; j < 3; j ++ ) {
                        System.out.println("i = " + i + "; j = " +j);
                        if ( i == 2)
                            break kedua; //ubah1
                    }
            }
    }
}
```

Luaran Contoh 2:

```
i = 1; j = 1
i = 1; j = 2
i = 2; j = 1
i = 3; j = 1
i = 3; j = 2
i = 4; j = 1
i = 4; j = 2
```

Contoh 3: Salin dan tempel kode program berikut ke Eclipse.

```
import java.util.Scanner;

public class ForBersarang {
    public static void main(String[] args){
        //Instance Input Scanner
        Scanner input = new Scanner(System.in);
        System.out.print("Masukan Input: ");
        int tinggi = input.nextInt(); //Mendapatkan Input Dari User
        for(int t=tinggi; t>=1; t--){
            //Menghitung Jumlah Tinggi Piramida
            for(int s=tinggi; s>=t; s--){
                //Menghitung Jumlah Spasi per Baris
                System.out.print(" ");
            }
            System.out.println(); //Membuat Baris Baru
        }
    }
}
```

Luaran contoh 3:

```
Masukan Input: 7
*
**
***
****
*****
*****
*****
*****
```

Latihan 1

1.1 Analisa tujuan penulisan kata kunci `continue` dan `break` pada Contoh 1!

Buat perubahan nilai angka pada variabel di

//Ubah 1 menjadi `for (int y = 0; y <= 15; y++)` { lalu running, periksa hasilnya

//Ubah 2 menjadi `if (y % 2 == 0)` lalu running, periksa hasilnya

//Ubah 3 menjadi `else if (y == 9)` lalu running, periksa hasilnya

Analisa dampaknya perubahan ini terhadap luaran setelah running!

Jawaban :

Analisa Kata kunci

1. Kata kunci `continue` :

`Continue` digunakan untuk melewati iterasi saat ini dalam loop. Dalam konteks kode asli, jika `y` adalah bilangan ganjil, loop akan langsung melanjutkan ke iterasi berikutnya tanpa mencetak nilai `y`.

2. Kata kunci `break` :

`Break` digunakan untuk menghentikan loop sepenuhnya. Dalam kode asli, jika `y` sama dengan 8, loop akan berhenti dan tidak akan melanjutkan ke nilai berikutnya.

Analisa Dampak perubahan :

1. Penggunaan `continue` :

Semua bilangan genap (0,2,4,6,8,10,12,14) akan dilewatkan, sehingga tidak dicetak sama sekali.

2. Loop berhenti pada saat `y` mencapai 9. Ini mengakhiri semua iterasi lebih lanjut, jadi bilangan ganjil yang lebih tinggi dari 9 tidak akan dicetak misalnya (9,11,13,15).

1.2 Buat perubahan kode pada Contoh 2 di baris //Ubah1 menjadi

- a. `continue` pertama; lalu running, periksa hasilnya
- b. `break` pertama; lalu running, periksa hasilnya
- c. `continue` kedua; lalu running, periksa hasilnya

Analisa perbedaan perubahan kode pada Ubah 1 untuk setiap poin (a), (b), dan (c)!

Jawaban :

Analisa Perbedaan perubahan :

1. Continue pertama

- Menghentikan iterasi loop pertama untuk $i = 2$.
- Nilai $j = 2$ tidak dicetak untuk $i = 2$, tetapi iterasi dilanjutkan untuk nilai i berikutnya.

2. Break pertama

- Menghentikan seluruh loop pertama jika $i = 2$.
- Hanya output untuk $i = 1$ yang dicetak, tidak ada output untuk $i = 2$, $i = 3$, dan $i = 4$.

3. Continue kedua

- Menghentikan iterasi loop kedua ketika $i = 2$.
- Nilai $j = 2$ tidak dicetak untuk $i = 2$, tetapi program melanjutkan untuk nilai i yang lebih tinggi.

1.3 Cermati kode contoh 3. Apabila ingin menghasilkan luaran berikut:

```
Luaran:  
Masukan Input: 7  
*****  
*****  
*****  
****  
***  
**  
*
```

Susunlah analisa kode untuk menghasilkan luaran tersebut!

Jawaban :

Analisis Kode :

1. Input dari pengguna : menggunakan scanner untuk meminta pengguna memasukkan tinggi piramida.
2. Loop luar : menggunakan loop yang dimulai dari tinggi yang dimasukkan oleh pengguna dan berkurang hingga mencapai 1.
3. Loop dalam : untuk setiap iterasi dari loopluar, mencetak asterisk sebanyak nilai dari itersi loop luar (tinggi yang tersisa)
4. Mencetak asterisk : pada setiap baris, mencetak asterisk sesuai dengan nilaitinggi yang tersisa.
5. Baris baru : setelah mencetak asterisk untuk setiap baris, maka perlu membuat baris baru.

1.4 Analisa diagram flowchart dari Latihan 1.2 dan 1.3!

Jawaban :

Analisa diagram flowchart latihan 1.2 :

1. Mulai : merupakan titik awal dari program. Semua alur eksekusi dimulai dari sini.
2. Inisialisasi i : Variabel i diinisialisasi dengan nilai 1. Ini menandakan bahwa program akan mulai dengan iterasi pertama dari loop luar.
3. Loop luar ($i < 5$): Memeriksa apakah nilai i kurang dari 5.
 - Ya : Melanjutkan ke langkah berikutnya (inisialisasi j)
 - Tidak : Program selesai (End).
4. Inisialisasi j : Variabel j diinisialisasi dengan nilai 1 untuk memulai iterasi dalam loop dalam.
5. Loop dalam ($j < 3$): Memeriksa apakah nilai j kurang dari 3.
 - Ya : Melanjutkan ke langkah berikutnya (mencetak i dan j).
 - Tidak : Kembali ke loop luar dan meningkatkan i.
6. Cetak i dan j : Program mencetak nilai saat ini dari i dan j.
7. Kondisi $i == 2$: Memeriksa apakah nilai i sama dengan 2.
 - Jika Ya : Melakukan tindakan tergantung pada perubahan yang telah dibuat.
8. Tingkatkan j : Jika kondisi $i == 2$ tidak terpenuhi, program meningkatkan nilai j dan kembali ke langkah 5 untuk memeriksa kembali.
9. Selesai : Titik akhir program. Menandakan bahwa semua iterasi telah selesai dan tidak ada lagi output yang akan dihasilkan.

Analisa diagram flowchart latihan 1.3 :

1. Mulai : Menandakan bahwa program dimulai. Ini adalah titik awal dari alur eksekusi.
2. Input dari pengguna : Program meminta pengguna untuk memasukkan nilai tinggi piramida, yaitu untuk menentukan berapa banyak bintang yang akan dicetak setiap baris.
3. Inisialisasi t : Variabel t diatur dengan nilai yang dimasukkan oleh pengguna untuk menentukan jumlah bintang yang akan dicetak pada baris pertama dan seterusnya berkurang pada setiap iterasi.
4. Loop luar ($t \geq 1$) : Memeriksa apakah nilai t masih lebih besar atau sama dengan 1.
 - Jika Ya : Melanjutkan ke langkah berikutnya, yaitu mencetak bintang.
 - Jika Tidak : Arahkan ke langkah Selesai, menandakan bahwa semua baris telah dicetak.
5. Inisialisasi Loop dalam : Mengatur loop untuk mencetak bintang.

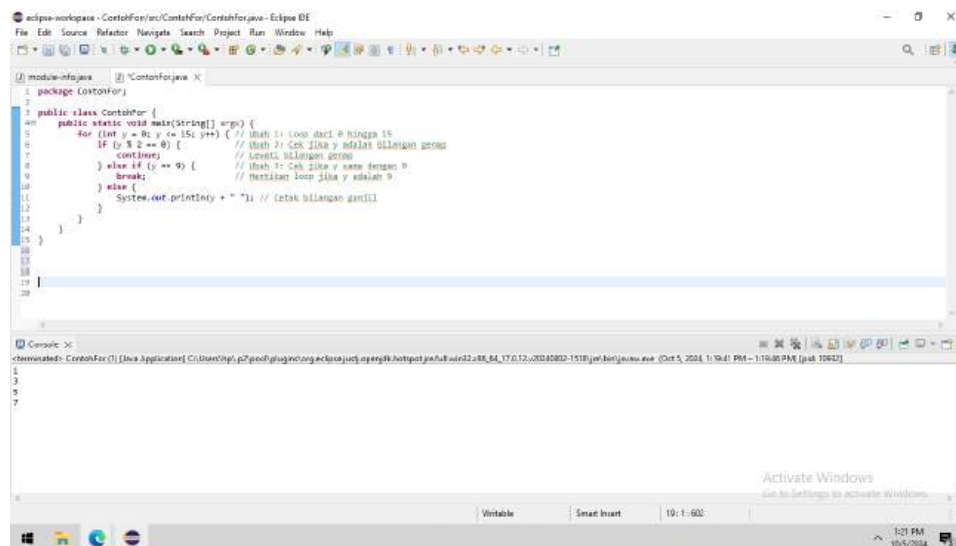
6. Loop dalam ($b \leq t$) : Memeriksa apakah nilai b kurang dari atau sama dengan t .
 - Jika Ya : Mencetak bintang ke layar.
 - Jika Tidak : Kembali ke loop luar untuk melanjutkan iterasi berikutnya.
7. Mencetak baris baru : Setelah mencetak semua bintang untuk satu baris, program mencetak baris baru menggunakan `System.out.println()`.
8. Tingkatkan t : Mengurangi nilai t setelah semua bintang pada baris tersebut dicetak.
9. Selesai : Menandakan akhir dari program. Di sini, program akan berhenti setelah semua baris dicetak.

[No 1.1] Analisis dan Argumentasi

- 1) Saya mengusulkan permasalahan ini dapat diatasi dengan cara menyusun Kode program dan menambahkan logika loop, kondisi continue, dan break serta menyesuaikan program.
- 2) Alasan solusi ini karena untuk mencapai tujuan yang diinginkan, baik itu menampilkan semua angka, hanya angka ganjil, atau menghentikan loop pada angka tertentu.
- 3) Perbaiki kode program dengan cara mengubah logika loop, kondisi continue, dan break di dalam beberapa program.

[No 1.1] Penyusunan Algoritma dan Kode Program

- 1) Algoritma
 - (a) Inisialisasi
 - (b) Mulai loop
 - (c) Periksa Ganjil
 - (d) Periksa untuk hentikan
 - (e) Cetak angka
 - (f) Increment
 - (g) Selesai
- 2) Kode program dan luaran
 - a) Screenshot Kode program, hasil luaran, dan Komentar



The screenshot shows the Eclipse IDE with a Java file named 'ContohFor.java'. The code is as follows:

```
1 package contohfor;  
2  
3 public class ContohFor {  
4     public static void main(String[] args) {  
5         for (int y = 0; y <= 15; y++) { // Loop dari 0 hingga 15  
6             if (y % 2 == 0) { // Jika y adalah bilangan genap  
7                 continue; // Lewati bilangan genap  
8             } else if (y == 9) { // Jika y sama dengan 9  
9                 break; // Hentikan loop jika y adalah 9  
10            } else {  
11                System.out.println(" " + y); // Cetak bilangan ganjil  
12            }  
13        }  
14    }  
15 }  
16  
17  
18  
19  
20
```

The console output shows the numbers 1, 3, 5, and 7, indicating that the loop was terminated before reaching 9.

- b) Analisa luaran yang dihasilkan.

Luaran sudah sesuai dengan program yang disusun. Perubahan beberapa program ini berpengaruh pada hasil keluaran yang lebih terfokus pada angka ganjil, dan loop dihentikan lebih awal.

[No 1.1] Kesimpulan

(PILIH SALAH SATU ANDA INGIN MEMBAHAS DENGAN CARA ANALISA/ EVALUASI / KREASI)

1) Analisa

Pada Program ini, dapat disimpulkan terkait penggunaan continue yang digunakan untuk melewati iterasi dari loop sehingga angka yang memenuhi kondisi tertentu yang akan diproses, dan break digunakan untuk menghentikan loop sepenuhnya, yang memungkinkan untuk keluar dari loop sebelum mencapai batas yang ditentukan. Perubahan pada kondisi loop dan pengkondisian dapat memengaruhi hasil yang dicetak. Dengan memodifikasi batas loop, kondisi untuk mencetak, dan kriteria untuk menghentikan loop, kita dapat mengontrol output program secara efektif.

[No 1.2] Analisis dan Argumentasi

- 1) Saya mengusulkan permasalahan ini dapat diatasi dengan cara menyusun Kode program yang terdiri atas continue pertama, break pertama, dan continue kedua .
- 2) Alasan solusi ini karena untuk menganalisa perbedaan setiap perubahan kode untuk setiap poin (a), (b), dan (c).
- 3) Perbaiki kode program dengan cara mengubah //Ubah1 menjadi continue pertama, lalu mengubah //Ubah1 menjadi break pertama, dan mengubah //Ubah1 menjadi continue kedua.

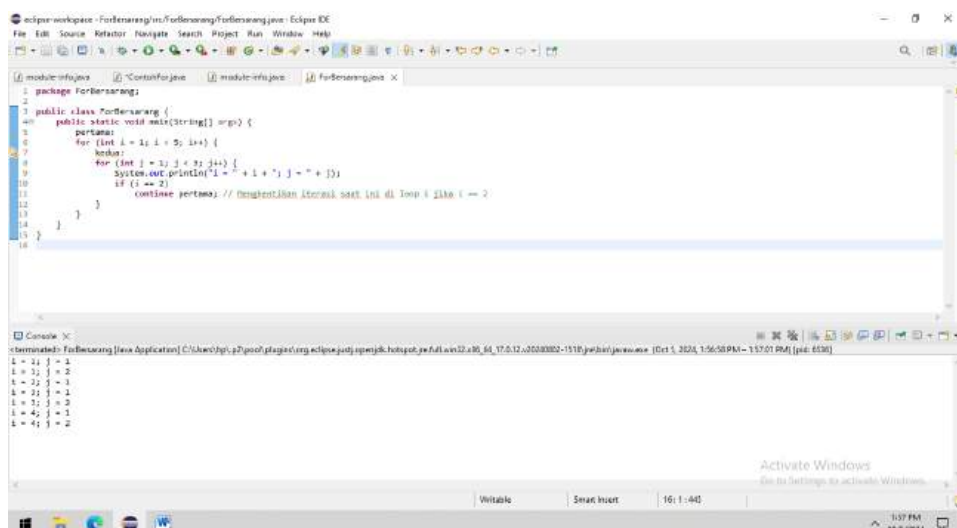
[No 1.2] Penyusunan Algoritma dan Kode Program

1) Algoritma

- (a) Inisialisasi loop
- (b) Mulai loop dalam
- (c) Cetak nilai
- (d) Kondisi untuk control alur
- (e) Selesaikan Iterasi
- (f) Akhiri loop
- (g) Akhiri program

2) Kode program dan luaran

- a) Screenshot Kode program, hasil luaran, dan Komentar



The screenshot shows the Eclipse IDE interface. The top part displays the source code for a Java class named `FortBersarang`. The code includes a package declaration, a class declaration, and a static method `main` that takes an array of strings as input. Inside the `main` method, there are two nested loops. The outer loop is a `for` loop that iterates from `i = 1` to `i = 5`. The inner loop is a `for` loop that iterates from `j = 1` to `j = i`. Inside the inner loop, there is a `System.out.println` statement that prints the value of `i` followed by a space and the value of `j`. There is a comment `// Ubah1` above the inner loop. Below the inner loop, there is a `continue pertama;` statement. The bottom part of the screenshot shows the console output, which displays the results of the program execution. The output shows the values of `i` and `j` for each iteration of the loops.

```
package FortBersarang;

public class FortBersarang {
    public static void main(String[] args) {
        pertama:
        for (int i = 1; i <= 5; i++) {
            kedua:
            for (int j = 1; j <= i; j++) {
                System.out.println("i = " + i + ", j = " + j);
                // Ubah1
            }
            continue pertama; // Menghentikan iterasi saat ini di loop i jika i == 2
        }
    }
}
```

Console Output:

```
i = 1, j = 1
i = 1, j = 2
i = 2, j = 1
i = 2, j = 2
i = 3, j = 1
i = 3, j = 2
i = 4, j = 1
i = 4, j = 2
```

```

package ForBersarang;

public class ForBersarang {
    public static void main(String[] args) {
        pertama:
        for (int i = 1; i < 3; i++) {
            kedua:
            for (int j = 1; j < 3; j++) {
                System.out.println("i = " + i + "; j = " + j);
                if (i == 2)
                    break pertama; // Menghentikan seluruh loop pertama jika i == 2
            }
        }
    }
}

```

Console Output:

```

i = 1; j = 1
i = 1; j = 2
i = 2; j = 1

```

```

package ForBersarang;

public class ForBersarang {
    public static void main(String[] args) {
        pertama:
        for (int i = 1; i < 3; i++) {
            kedua:
            for (int j = 1; j < 3; j++) {
                System.out.println("i = " + i + "; j = " + j);
                if (i == 2)
                    continue kedua; // Menghentikan iterasi saat ini di loop j jika i == 2
            }
        }
    }
}

```

Console Output:

```

i = 1; j = 1
i = 1; j = 2
i = 1; j = 3
i = 2; j = 1
i = 2; j = 2
i = 2; j = 3
i = 3; j = 1
i = 3; j = 2
i = 3; j = 3

```

- c) Analisa luaran yang dihasilkan.
 Luaran sudah sesuai dengan program yang disusun. Serta luaran dari ketiga variasi program yang dihasilkan sudah sesuai dengan kode program yang dibuat dengan modifikasi pada bagian //Ubah1.

[No 1.2] Kesimpulan

(PILIH SALAH SATU ANDA INGIN MEMBAHAS DENGAN CARA ANALISA/ EVALUASI / KREASI)

1) Analisa

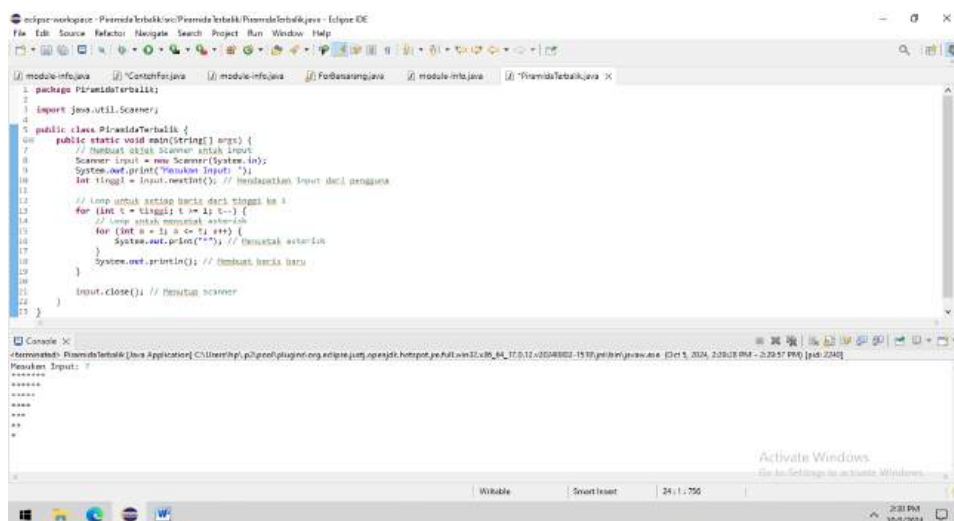
Pada Program ini, menggunakan break dan continue dalam loop bersarang. Dengan break, program dapat menghentikan loop sepenuhnya, sementara continue memungkinkan untuk melanjutkan ke iterasi berikutnya, baik dalam loop yang sama atau yang lebih luar. Setiap pernyataan continue dan break menghasilkan pola output yang berbeda berdasarkan bagaimana keduanya mengelola control alur di dalam dan di luar loop.

[No 1.3] Analisis dan Argumentasi

- 1) Saya mengusulkan permasalahan ini dapat diatasi dengan cara menyusun Kode program untuk menghasilkan piramida terbalik dari asterisk (*).
- 2) Alasan solusi ini karena untuk menghasilkan piramida terbalik dari asterisk (*), yang berkurang satu per baris hingga mencapai satu asterisk.
- 3) Perbaiki kode program dengan cara membuat kode program untuk menghasilkan keluaran dalam bentuk piramida terbalik yang dibuat dari asterisk (*).

[No 1.3] Penyusunan Algoritma dan Kode Program

- 1) Algoritma
 - (a) Mulai
 - (b) Inisialisasi Scanner
 - (c) Tanya pengguna
 - (d) Baca input
 - (e) Tutup Scanner
 - (f) Selesai
- 2) Kode program dan keluaran
 - a) Screenshot Kode program, hasil keluaran, dan Komentar



The screenshot shows a code editor with the following Java code:

```
1 package PiramidaTerbalik;
2
3 import java.util.Scanner;
4
5 public class PiramidaTerbalik {
6     public static void main(String[] args) {
7         // membuat objek scanner untuk input
8         Scanner input = new Scanner(System.in);
9         System.out.print("Masukkan Input: ");
10        int tinggi = input.nextInt(); // mendapatkan input dari pengguna
11
12        // Loop untuk setiap baris dari tinggi ke 1
13        for (int t = tinggi; t >= 1; t--) {
14            // Loop untuk mencetak asterisk
15            for (int s = 1; s <= t; s++) {
16                System.out.print("*"); // mencetak asterisk
17            }
18            System.out.println(); // membuat baris baru
19        }
20        input.close(); // menutup scanner
21    }
22 }
```

The console output shows the user entering '7' and the resulting reversed pyramid:

```
Masukkan Input: 7
*****
****
***
**
*

```

- b) Analisa keluaran yang dihasilkan.
Keluaran sudah sesuai dengan program yang disusun, dan jika pengguna memasukkan angka maka akan menghasilkan keluaran sesuai yang diinginkan.

[No 1.3] Kesimpulan

(PILIH SALAH SATU ANDA INGIN MEMBAHAS DENGAN CARA ANALISA/ EVALUASI / KREASI)

- 1) Analisa
Pada Program ini, meliputi pada cara mencetak piramida terbalik menggunakan asterisk pada java. Program dimulai dengan meminta input dari pengguna mengenai tinggi piramida, sebelum melanjutkan program melakukan validasi untuk memastikan bahwa input yang diberikan adalah bilangan positif. Setelah validasi, program menggunakan dua loop. Setiap kali loop luar berjalan, jumlah asterisk yang dicetak berkurang satu, sehingga membentuk pola piramida terbalik.

[No. 2] Identifikasi Masalah:

Contoh 4: Salin dan tempel kode program berikut ke Eclipse.

```
public class ContohWhile{  
    public static void main(String[] args) {  
        int i=1;  
        while(i<=6){  
            System.out.println(i);  
            i++;  
            if(i==4){  
                break;          //ubah1  
            }  
        }  
    }  
}
```

Luaran:

```
1  
2  
3
```

Contoh 5: Salin dan tempel kode program berikut ke Eclipse.

```
public class WhileBersarang {  
    public static void main(String[] args) {  
        int count = 0; //ubah1  
        while (count < 20) {  
            if (count % 3 == 0) //ubah2  
                System.out.println(count);  
            count++;  
        }  
    }  
}
```

Luaran:

```
0  
3  
6  
9  
12  
15  
18
```

Latihan 2

2.1 Buat perubahan nilai angka pada variabel di Contoh 4

//Ubah 1 menjadi continue; lalu running, periksa hasilnya

Analisa dampaknya perubahan terhadap luaran setelah running dan uraikan kegunaan break dan continue!

Jawaban :

Dampak Perubahan :

- Dengan menggunakan continue, ketika i mencapai 4, program tidak akan mencetak angka tersebut. Namun, karena continue hanya mengalihkan eksekusi kembali ke awal loop, pernyataan System.out.println(i); setelah continue tidak akan dieksekusi untuk nilai i yang sama (yaitu 4)
- Namun, setelah continue, i tetap akan ditambah (incrementation) sehingga pada iterasi berikutnya, i menjadi 5, dan angka 5 akan dicetak.
- Program akan mencetak semua angka dari 1 hingga 6 tanpa menghentikan loop.

Kegunaan break dan continue :

1. Break :

- Break digunakan untuk menghentikan loop sepenuhnya. Ketika pernyataan break dijalankan, program akan keluar dari loop dan melanjutkan eksekusi kode setelah loop tersebut.
- Kegunaan : Berguna saat ingin menghentikan loop berdasarkan suatu kondisi tertentu (misalnya, menemukan nilai yang diinginkan).

2. Continue :

- Continue digunakan untuk melewati iterasi saat ini dan melanjutkan dengan iterasi berikutnya dalam loop. Ini hanya memengaruhi siklus saat ini dan tidak menghentikan keseluruhan loop.
- Kegunaan : Berguna ketika ingin melewati beberapa kondisi tertentu tetapi tetap ingin menjalankan loop untuk nilai lainnya.

2.2 Buat perubahan nilai angka pada variabel di Contoh 5

//Ubah2 menjadi `if (count % 5 == 0)` lalu running, periksa hasilnya

Analisa dampaknya perubahan terhadap luaran setelah running dan uraikan kegunaan % untuk angka yang berbeda pada perintah tersebut!

Jawaban :

Dampak perubahan :

- Pengaruh pada Output

Dengan menggunakan `count % 5 == 0`, program hanya mencetak angka-angka yang dapat dibagi oleh 5. Berbeda dengan sebelumnya yang mencetak angka yang dapat dibagi oleh 3.

Kegunaan Operator % (Modulus) :

Operator modulus (%) digunakan untuk menentukan sisa dari pembagian dua angka.

Dalam konteks ini, `count % 5` menghasilkan 0 ketika count adalah kelipatan dari 5 (misalnya : 0, 5, 10, 15).

Jika mengganti angka dalam kondisi lain, misalnya :

- `Count % 2 == 0` : akan mencetak kelipatan dari 2 (0, 2, 4, 6, 8, 10, 12, 14, 16).
- `Count % 3 == 0` : akan mencetak kelipatan dari 3 (0, 3, 6, 9, 12, 15, 18).
- `Count % 7 == 0` : akan mencetak kelipatan dari 7 (0, 7, 14, 21, 28).

2.3 Buat perubahan nilai angka pada variabel di

//Ubah1 menjadi `while (count < 0) {` lalu running, periksa hasilnya

ubahlah baris kode `while` pada Contoh 5 menjadi `do ... while` dengan persyaratan yang sama `while (count < 0)`. Bandingkan hasil luaran antara menggunakan `while` dan `do ... while`!

Jawaban :

Perbandingan Hasil luaran :

1. Menggunakan while :

- Tidak ada output, karena kondisi `while (count < 0)` dan tidak terpenuhi sejak awal.

2. Menggunakan do...while

- Menghasilkan output 0, karena block dalam `do` dieksekusi setidaknya sekali, meskipun kondisinya false setelah iterasi pertama.

2.4 Analisa diagram flowchart dari Latihan 2.1, Contoh 5, dan Latihan 2.3!

Jawaban :

Analisa diagram flowchart latihan 2.1 :

1. Mulai : Menandakan bahwa program dimulai. Ini adalah titik awal dari alur eksekusi.
2. Inisialisasi i : Variabel i diinisialisasi dengan nilai 1. Ini adalah nilai awal yang akan digunakan dalam loop
3. Loop while ($i \leq 6$) : Memeriksa kondisi apakah i kurang dari atau sama dengan 6.
 - Jika Ya : Program melanjutkan untuk mengeksekusi blok di dalam loop.
 - Jika Tidak : Program menuju langkah selesai dan berhenti.
4. Cetak i : Mencetak nilai i ke layar. Ini adalah output yang dihasilkan dalam setiap iterasi loop.
5. Tingkatkan i : Nilai i ditingkatkan dengan 1, untuk memastikan bahwa loop akhirnya berhenti setelah beberapa iterasi.
6. Cek kondisi if ($i == 4$) : Memeriksa apakah nilai i sama dengan 4.
 - Jika Ya : tergantung apakah menggunakan break atau continue.
Dengan break : Menghentikan eksekusi loop sepenuhnya dan menuju langkah Selesai. Dengan continue : Mengabaikan sisa kode dalam iterasi dan kembali ke awal loop.
7. Selesai : Menandakan akhir dari program. Program selesai setelah semua iterasi yang valid telah diproses.

Analisa diagram flowchat Contoh 5 :

1. Mulai : Menandakan bahwa program dimulai. Ini adalah titik awal dari eksekusi program.
2. Inisialisasi count : Variabel count diinisialisasi dengan nilai 0. Ini adalah nilai awal yang akan digunakan untuk memulai perhitungan dalam loop.
3. Loop while ($\text{count} < 20$) : Memeriksa kondisi apakah count kurang dari 20
 - Jika Ya : Program melanjutkan ke langkah berikutnya di dalam loop.
 - Jika Tidak : Program akan menuju langkah selesai, menandakan bahwa tidak ada nilai yang akan diproses lebih lanjut.
4. Cek kondisi if ($\text{count} \% 3 == 0$) : Dalam setiap iterasi, program memeriksa apakah count dapat dibagi oleh 3 (yaitu, sisa bagi 3 adalah 0).
 - Jika Ya : Program mencetak nilai count ke layar.
 - Jika Tidak : Program akan melewati langkah ini dan langsung melanjutkan ke langkah berikutnya.
5. Tingkatkan count : Setelah memeriksa kondisi, nilai count ditingkatkan dengan 1. Ini penting untuk memastikan bahwa loop akan berhenti setelah mencapai 20.

6. Kembali ke Loop : Program kembali ke langkah 3 untuk memeriksa kembali kondisi loop. Jika count masih kurang dari 20, proses diulang.
7. Selesai : Menandakan akhir dari program. Ketika semua iterasi yang valid telah diproses, program berhenti.

Analisa diagram flowchart latihan 2.3 :

Menggunakan While

1. Mulai : Menandakan bahwa program dimulai. Ini adalah titik awal eksekusi
2. Inisialisasi count : Variabel count diinisialisasi dengan nilai 0, untuk menentukan nilai awal yang akan digunakan dalam loop.
3. Loop while (count < 0) : Program memeriksa kondisi apakah count kurang dari 0.
 - Jika Ya : Program akan melanjutkan ke langkah berikutnya.
 - Jika Tidak : Program akan langsung menuju langkah Selesai.
4. Cek kondisi if (count % 3 == 0) : Memeriksa apakah count dapat dibagi oleh 3.
 - Jika Ya : Mencetak nilai count.
 - Jika Tidak : Langsung ke langkah berikutnya tanpa melakukan apa-apa.
5. Tingkatkan count : Meningkatkan nilai count sebesar 1. Namun, karena tidak ada eksekusi di langkah sebelumnya, bagian ini tidak akan dijalankan.
6. Kembali ke loop : Kembali untuk memeriksa kondisi loop lagi. Tetapi karena count sudah 0, tidak ada iterasi yang terjadi.
7. Selesai : Menandakan akhir dari program. Program berhenti tanpa menghasilkan output.

Menggunakan Do...While

1. Mulai : Menandakan bahwa program dimulai. Ini adalah titik awal eksekusi.
2. Inisialisasi count : Variabel count diinisialisasi dengan nilai 0, untuk menentukan nilai awal yang digunakan dalam loop.
3. Eksekusi Blok : Program langsung menjalankan blok kode di dalam do setidaknya sekali.
4. Cek kondisi if (count % 3 == 0) : Memeriksa apakah count dapat dibagi oleh 3.
 - Jika Ya : Mencetak nilai count.
 - Jika Tidak : Langsung ke langkah berikutnya tanpa melakukan apa-apa.
5. Tingkatkan count : Meningkatkan nilai count menjadi 1.

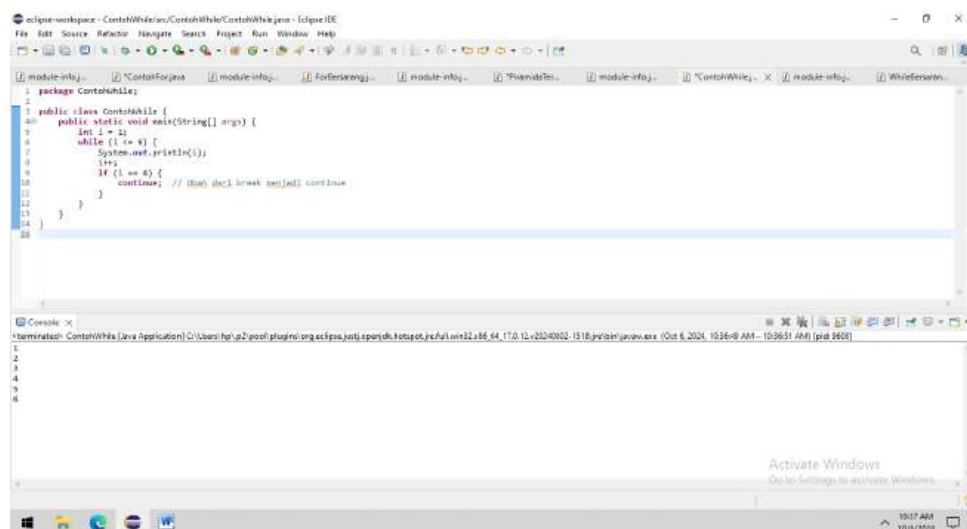
6. Loop while ($\text{count} < 0$) : Memeriksa apakah count kurang dari 0.
 - Jika Ya : Kembali ke langkah 3 untuk menjalankan blok lagi (tidak akan terjadi karena count adalah 1).
 - Jika Tidak : Program menuju langkah Selesai.
7. Selesai : Menandakan akhir dari program. Program selesai setelah mencetak 0.

[No 2.1] Analisis dan Argumentasi

- 1) Saya mengusulkan permasalahan ini dapat diatasi dengan cara menyusun Kode program dan mengganti pernyataan break dengan continue.
- 2) Alasan solusi ini karena untuk mengetahui perubahan terhadap luaran setelah di running.
- 3) Perbaiki kode program dengan cara membuat kode program dengan mengganti break ke continue. Karena ini memungkinkan program untuk berjalan dan mencetak angka setelah kondisi tertentu.

[No 2.1] Penyusunan Algoritma dan Kode Program

- 1) Algoritma
 - (a) Inisialisasi variabel
 - (b) Looping
 - (c) Akhir Loop
 - (d) Selesai
- 2) Kode program dan luaran
 - a) Screenshot Kode program, hasil luaran, dan Komentar



The screenshot shows an IDE window with a Java file named 'ContohWhile.java'. The code is as follows:

```
1 package ContohWhile;
2
3 public class ContohWhile {
4     public static void main(String[] args) {
5         int i = 1;
6         while (i <= 5) {
7             System.out.println(i);
8             i++;
9             if (i == 4) {
10                 continue; // (atau break也行) continue
11             }
12         }
13     }
14 }
```

The console output shows the numbers 1, 2, 3, and 5, indicating that the loop skipped the number 4 due to the 'continue' statement.

- b) Analisa luaran yang dihasilkan.
Luaran sudah sesuai dengan program yang disusun. Serta kode program yang dibuat sudah sesuai dengan hasilnya.

[No 2.1] Kesimpulan

(PILIH SALAH SATU ANDA INGIN MEMBAHAS DENGAN CARA ANALISA/ EVALUASI / KREASI)

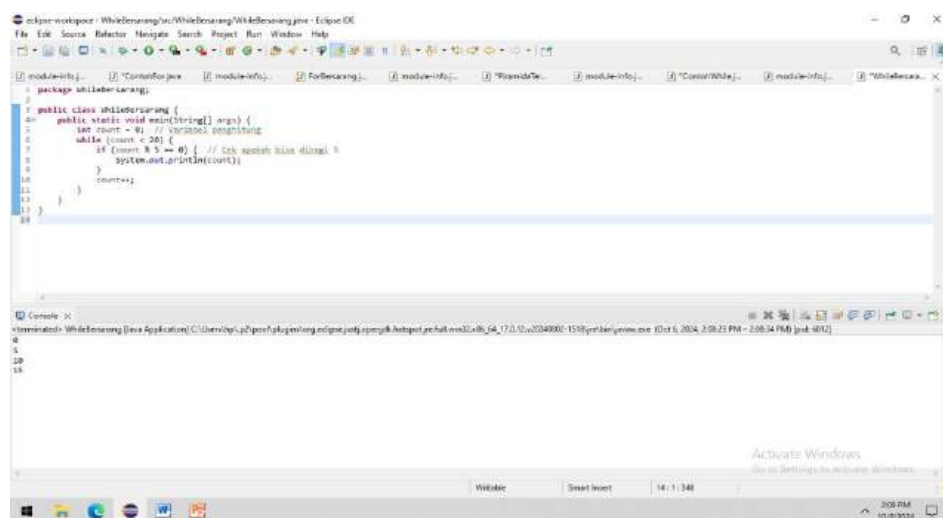
- 1) **Analisa**
Pada program ini, menggunakan pernyataan break dan continue. Dengan memahami dan menerapkan penggunaan break dan continue, dapat lebih fleksibel dalam mengontrol alur program. Pilihan antara keduanya tergantung pada kebutuhan spesifik dalam program yang sedang dikembangkan. Ini memperlihatkan fleksibilitas yang berbeda antara keduanya dalam pengontrolan alur program.

[No 2.2] Analisis dan Argumentasi

- 1) Saya mengusulkan permasalahan ini dapat diatasi dengan cara menyusun kode program dengan cara menggunakan operator modulus dengan tepat .
- 2) Alasan solusi ini karena untuk memeriksa apakah suatu angka habis dibagi dengan angka tertentu.
- 3) Perbaiki kode program dengan cara membuat kode program dengan menggunakan operator modulus, yaitu untuk memeriksa apakah suatu nilai habis dibagi oleh nilai lainnya, yang meningkatkan keterbacaan dan modularitas.

[No 2.2] Penyusunan Algoritma dan Kode Program

- 1) Algoritma
 - (a) Inisialisasi
 - (b) Loop
 - (c) Cek kondisi
 - (d) Cetak nilai
 - (e) Tingkatkan penghitung
 - (f) Selesai
- 2) Kode program dan luaran
 - a) Screenshot Kode program, hasil luaran, dan Komentar



The screenshot shows the Eclipse IDE with a Java project named 'Whiledenarang'. The source code in 'Whiledenarang.java' is as follows:

```
1 package whiledenarang;
2
3 public class Whiledenarang {
4     public static void main(String[] args) {
5         int count = 0; // variabel penghitung
6         while (count < 20) { // Cek apakah bisa dibagi 5
7             if (count % 5 == 0) {
8                 System.out.println(count);
9             }
10            count++;
11        }
12    }
13 }
```

The console output at the bottom shows the numbers 0, 5, 10, 15, and 20, which are the values of 'count' that are divisible by 5.

- b) Analisa luaran yang dihasilkan.

Luaran sudah sesuai dengan program yang disusun. Dengan perubahan ini, program akan mencetak angka- angka dari 0 hingga 19 yang habis dibagi oleh 5.

[No 2.2] Kesimpulan

(PILIH SALAH SATU ANDA INGIN MEMBAHAS DENGAN CARA ANALISA/ EVALUASI / KREASI)

- 1) **Analisa**

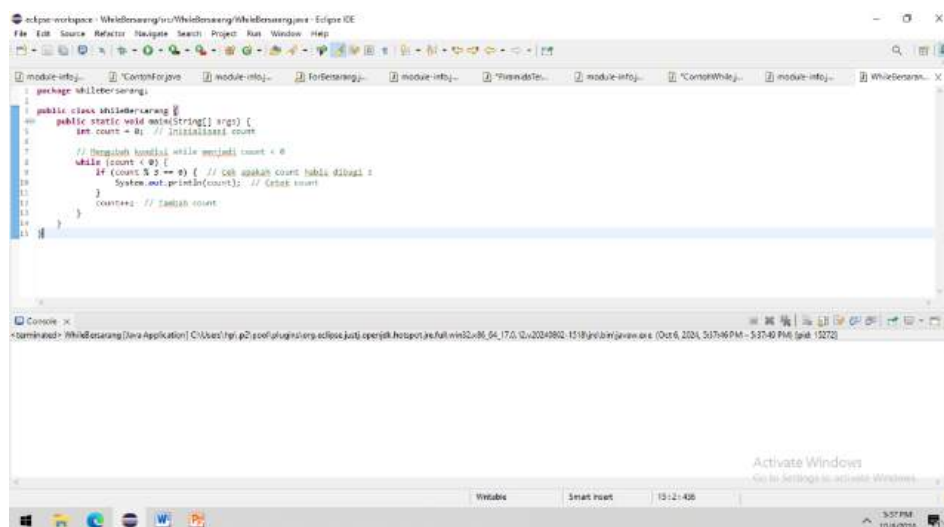
Pada program ini, menggunakan operator modulus. Operator ini dapat digunakan untuk memeriksa apakah suatu nilai memenuhi kriteria tertentu, seperti apakah habis dibagi oleh nilai lainnya. Salah satu penggunaan paling umum dari operator modulus adalah untuk menentukan apakah suatu angka genap atau ganjil. Jika hasil pembagian suatu angka dengan dua menghasilkan sisa nol, maka angka tersebut genap. Sebaliknya, jika sisa pembagiannya satu, maka angka tersebut ganjil.

[No 2.3] Analisis dan Argumentasi

- 1) Saya mengusulkan permasalahan ini dapat diatasi dengan cara menyusun kode program dengan mengubah kode while ($\text{count} < 0$) lalu mengubah kode dengan menggunakan do...while.
- 2) Alasan solusi ini karena untuk mengetahui perbandingan while dan do... while ketika di eksekusi untuk mengetahui outputnya.
- 3) Perbaiki kode program dengan cara membuat kode program dengan menggunakan while ($\text{count} < 0$) dan do...while.

[No 2.3] Penyusunan Algoritma dan Kode Program

- 1) Algoritma
 - (a) Inisialisasi
 - (b) Mengecek penggunaan while
 - (c) Mengecek penggunaan do...while
 - (d) Akhiri program
 - (e) Selesai
- 2) Kode program dan luaran
 - a) Screenshot Kode program, hasil luaran, dan Komentar



The screenshot shows the Eclipse IDE interface. The main editor displays a Java class named `WhiledoWhile` with the following code:

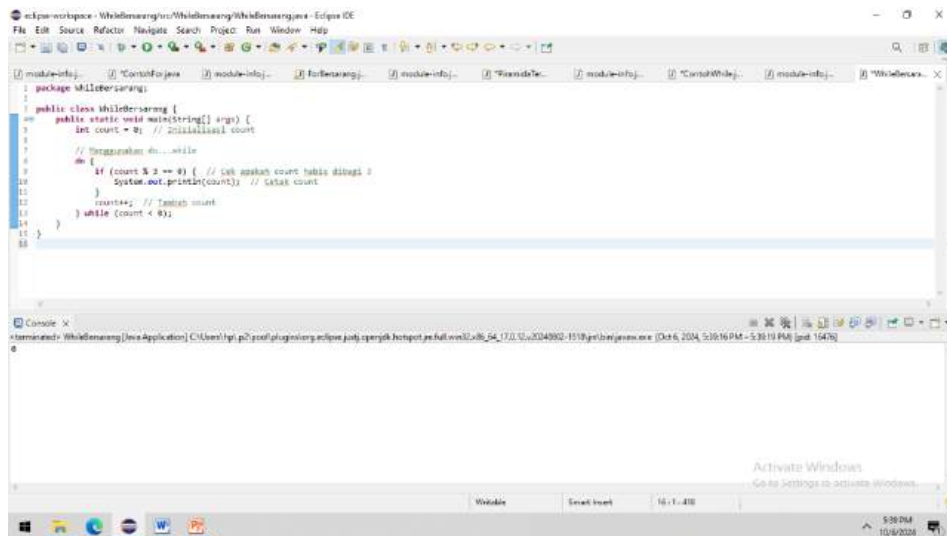
```
package whiledoWhile;

import java.util.Scanner;

public class WhiledoWhile {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        int count = 0; // Inisialisasi count

        // Mengubah kondisi while menjadi count < 0
        while (count < 0) {
            if (count % 2 == 0) { // cek apakah count habis dibagi 2
                System.out.println(count); // cetak count
            }
            count++; // tambahkan count
        }
    }
}
```

The console window at the bottom shows the output of the program, which is empty, indicating that the loop did not execute because the initial value of `count` (0) did not satisfy the condition `count < 0`.



- b) analisa luaran yang dihasilkan.
 Luaran sudah sesuai dengan program yang disusun. Dengan menampilkan output dari while serta output do...while.

[No 2.3] Kesimpulan

(PILIH SALAH SATU ANDA INGIN MEMBAHAS DENGAN CARA ANALISA/ EVALUASI / KREASI)

1) Analisa

Pada program ini, menggunakan Kode dengan while (count < 0) dan menggunakan kode yaitu do...while. Hasil dari while yaitu tidak ada output karena kondisi count < 0 tidak terpenuhi pada awal. Serta hasil dari do...while tidak ada output karena count tetap 0, dan meskipun blok dijalankan sekali, kondisi untuk melanjutkan tidak terpenuhi.

Refleksi :

Pengalaman belajar yang didapat yakni mendapatkan pelajaran dan pembahasan mengenai pemrograman yaitu FOR dan WHILE yang ada di java, meskipun masih ada yang belum dipahami. Pengetahuan yang di dapatkan yakni dapat mengetahui bagaimana cara kerja yang ada pada FOR dan WHILE . Serta mendapat pengetahuan juga mengenai Coding. Tantangan yang saya hadapi yakni dalam melakukan coding terkadang terjadi Error.