

Template Lembar Kerja Individu

Nama & NPM	Topik:	Tanggal:
Syifa Ariqah Pajriyanti (G1F024009)	FOR dan WHILE JAVA	5 Oktober 2024

[Nomor 1] Identifikasi Masalah:

1) Uraikan permasalahan dan variable

```
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

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!

Jawab: tujuan penulisan pada kata kunci `continue` digunakan untuk melewati iterasi saat nilai `y` adalah bilangan ganjil, sehingga bilangan genap yang akan dicetak. Sedangkan tujuan penulisan kata kunci `break` untuk menghentikan loop ketika `y` mencapai 8, mencegah nilai 8 dan angka yang lebih besar dicetak. Tujuannya adalah untuk menampilkan hanya bilangan genap dari 0 hingga 6.

Hasil perubahan pada //Ubah 1 menjadi `for (int y = 0; y <= 15; y++)` { periksa hasilnya :

0
2
4
6

Hasil perubahan pada //Ubah 2 menjadi `if (y % 2 == 0)` :

1
3
5
7
9

11

13

15

Hasil perubahan pada //Ubah 3 menjadi `else if (y == 9)` :

1

3

5

7

Contoh 2:

```
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
```

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

- `continue` pertama; lalu running, periksa hasilnya
- `break` pertama; lalu running, periksa hasilnya
- `continue` kedua; lalu running, periksa hasilnya

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

Jawab :

Hasil running dari poin a :

```
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
```

hasil running dari poin b :

```
i = 1; j = 1
```

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

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

Hasil running dari poin c :

```
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
```

analisa perbedaan perubahan pada setiap kode adalah

- pada kode a: Melanjutkan ke iterasi berikutnya pada loop pertama, mengabaikan loop kedua untuk i = 2.
- Pada kode b : Menghentikan seluruh loop pertama, sehingga tidak ada output lebih lanjut untuk nilai i setelah 1.
- Pada kode c: Mengulang kembali loop kedua tanpa melanjutkan ke iterasi berikutnya pada loop pertama, menghasilkan output untuk j saat i = 2.

Contoh 3:

```
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

```

*
**
***
****
*****
*****
*****

```

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

Luaran:

Masukan Input: 7

```

*****
*****
*****
****
***
**
*

```

Susunlah analisa kode untuk menghasilkan luaran tersebut!

Jawab:

```

import java.util.Scanner;

public class ForBersarang {

    public static void main(String[] args) {

        Scanner input = new Scanner(System.in);

        System.out.print("Masukan Input: ");

        int tinggi = input.nextInt();

        for (int t = tinggi; t >= 1; t--) {

            for (int s = t; s < tinggi; s++) {

                System.out.print(" ");

            }

            for (int b = 1; b <= t; b++) {

                System.out.print("*");

            }

            System.out.println();

        }

    }

}

```

```

    }

    input.close();

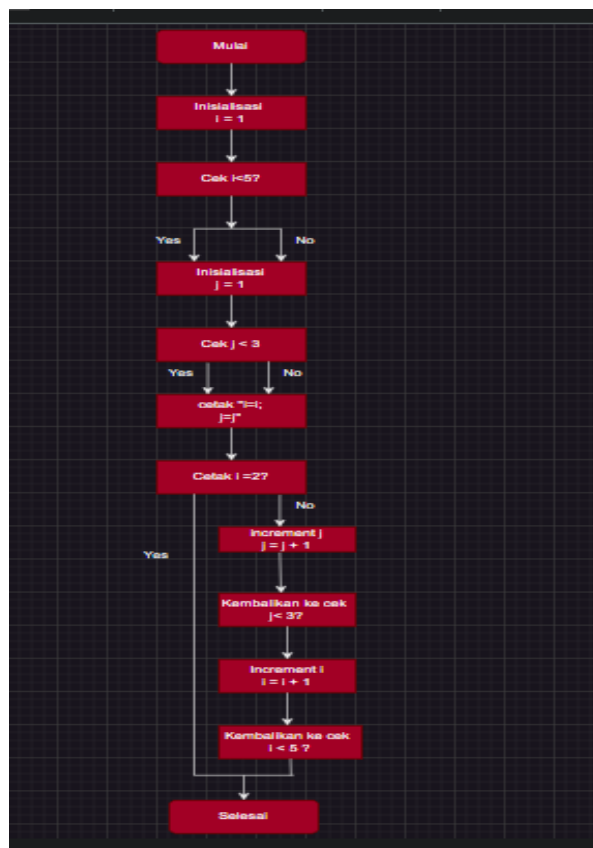
}

}

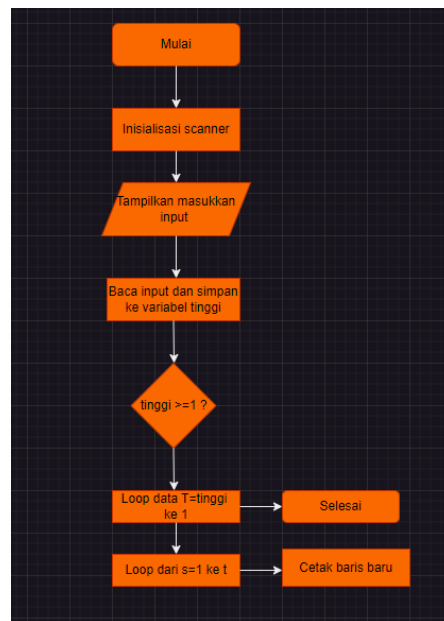
```

1.4. Analisa diagram flowchart dari Latihan 1.2 dan 1.3!

Flowchart 1.2



Flowchart 1.3



- 2) Rincikan sumber informasi yang relevan (buku / webpage)

[Video Materi 1 tentang FOR – https://www.youtube.com/watch?v=Ij9qLLblxEU](https://www.youtube.com/watch?v=Ij9qLLblxEU)

[Nomor 1] Analisis dan Argumentasi

- 1.1 A) Rancangan solusi yang saya usulkan adalah mencakup beberapa aspek penting. Pertama, inisialisasi loop saat ini menggunakan `for (int y = 0; y <= 10; ++y)`, yang bisa diubah menjadi `y < 10` untuk membatasi nilai maksimum. Kedua, pengecekan bilangan ganjil dilakukan dengan `if (y % 2 == 1)`, yang sudah tepat jika tujuannya adalah mencetak bilangan genap. Ketiga, kondisi `else if (y == 8)` menyebabkan loop berhenti ketika `y` mencapai 8, dan Anda mungkin ingin mempertimbangkan untuk mengubah nilai ini atau menghapusnya agar semua bilangan genap dicetak. Saat ini, output dari program mencetak bilangan genap dari 0 hingga 6: 0, 2, 4, dan 6. Dengan penyesuaian ini, Anda dapat mengubah output sesuai kebutuhan.

b) Analisis solusi, kaitkan dengan permasalahan.

Jawab : Dengan pemahaman yang lebih baik tentang batasan dan logika dalam loop, kita dapat merumuskan solusi yang lebih efektif. Jika tujuannya adalah untuk hanya mencetak bilangan genap dari 0 hingga 10, maka perbaikan pada bagian inisialisasi dan pengecekan kondisi akan menghasilkan output yang lebih sesuai.

- 1.2 A) rancangan solusi yang saya usulkan adalah menunjukkan cara menggunakan loop bersarang dan label untuk mengontrol alur program secara lebih tepat. Dengan menggunakan label, kita dapat menghentikan loop tertentu tanpa mempengaruhi struktur loop lainnya, memungkinkan fleksibilitas dalam pengendalian alur program.

b) analisis solusi kaitkan dengan permasalahan Dengan demikian, rancangan solusi ini secara efektif mengatasi permasalahan dalam mencetak kombinasi nilai dengan memberikan kontrol yang tepat atas alur eksekusi. Ini menunjukkan pentingnya memahami cara menggunakan loop bersarang dan label dalam pemrograman untuk menyelesaikan masalah yang lebih kompleks.

- 1.3 a) rancangan solusi yang saya usulkan adalah bagaimana menggunakan loop bersarang untuk mencetak piramida bintang berdasarkan input dari pengguna. Meskipun terdapat kesalahan dalam logika loop untuk spasi, ide dasar untuk menggunakan dua loop untuk mengatur tinggi dan spasi sudah ada. Dengan melakukan penyesuaian pada logika loop, kita dapat mencapai output yang diinginkan dengan benar.

b) Analisis ini menunjukkan bahwa meskipun struktur dasar untuk mencetak piramida sudah ada, terdapat kesalahan dalam logika yang harus diatasi. Dengan memperbaiki urutan pencetakan karakter (spasi terlebih dahulu, diikuti oleh bintang), solusi ini akan sepenuhnya menyelesaikan permasalahan yang dihadapi. Penggunaan loop bersarang merupakan pendekatan yang tepat untuk mencapai tujuan tersebut, tetapi perlu penyesuaian dalam implementasinya untuk mendapatkan hasil yang diinginkan.

[Nomor 1] Penyusunan Algoritma dan Kode Program

- 1) Rancang desain solusi atau algoritma
 - 1.1 algoritma
 - Mulai
 - Inisialisasi
 - Loop untuk mengontrol bilangan
 - Selesai

1.2 algoritma

point a :

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

hasil running dari poin b :

i = 1; j = 1

i = 1; j = 2

i = 2; j = 1

Hasil running dari poin c :

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

1.3 algoritma

- Mulai
- Deklarasi Inisialisasi
- Loop untuk mencetak baris
- Selesai

2) Tuliskan kode program dan luaran

a) Beri komentar pada kode

1.1 beri komentar pada kode

- Loop Utama: for (int y = 0; y <= 15; y++):
Loop ini akan mengulangi proses dari 0 hingga 15. Variabel y digunakan untuk melacak bilangan yang sedang diproses.
- Pemeriksaan Bilangan Genap: if (y % 2 == 0):
Kondisi ini memeriksa apakah bilangan y adalah genap. Jika kondisi ini benar, program akan melanjutkan ke iterasi berikutnya.
- Pernyataan continue:
Jika y adalah bilangan genap, pernyataan continue akan menghentikan eksekusi dari bagian selanjutnya dalam loop dan memulai iterasi berikutnya.
- Pemeriksaan Bilangan 9: else if (y == 9):
Kondisi ini memeriksa apakah y sama dengan 9. Jika benar, program akan menghentikan loop.
- Pernyataan break:
Jika y adalah 9, pernyataan break akan menghentikan eksekusi dari loop sepenuhnya.
- Output Bilangan Ganjil: System.out.println(y + " ");
Jika y tidak genap dan juga tidak sama dengan 9, program mencetak nilai y, yang merupakan bilangan ganjil.

1.2 berikan komentar pada kode:

- Kelas ForBersarang: Ini adalah kelas publik yang berisi metode main.
- Metode main: Titik masuk program Java.
- Label pertama: Label ini digunakan untuk loop luar (for dengan variabel i).
- Loop luar: Loop ini berjalan dari i = 1 hingga i < 5.
- Label kedua: Label untuk loop dalam (for dengan variabel j).
- Loop dalam: Loop ini berjalan dari j = 1 hingga j < 3.
- System.out.println: Mencetak nilai saat ini dari i dan j ke konsol.
- Pernyataan if: Jika i sama dengan 2,
 - perintah break digunakan untuk keluar dari loop luar (pertama), menghentikan kedua loop.
 - perintah continue digunakan untuk kembali ke awal loop dalam (kedua), melewati sisa iterasi saat ini dari loop dalam.
 - perintah continue digunakan untuk kembali ke awal loop luar (pertama), melewati sisa iterasi dari loop dalam dan melanjutkan ke iterasi berikutnya dari loop luar.

1.3 beri komentar pada kode :

- Impor Kelas: Kode dimulai dengan mengimpor kelas Scanner, yang diperlukan untuk mendapatkan input dari pengguna.
- Membuat Instance Scanner: Menginisialisasi objek Scanner untuk membaca input dari sistem (keyboard).
- Pesan Input: Menampilkan pesan untuk meminta pengguna memasukkan tinggi piramida.
- Mendapatkan Input: Membaca input dari pengguna dan menyimpannya dalam

variabel tinggi.

- Loop untuk Baris: Loop luar yang mengatur berapa banyak baris yang akan dicetak. Ini berjalan dari tinggi hingga 1.
- Loop untuk Spasi: Loop pertama dalam untuk mencetak spasi. Ini menentukan jumlah spasi yang perlu dicetak di depan bintang agar bintang terlihat terpusat.
- Loop untuk Bintang: Loop kedua dalam untuk mencetak bintang pada baris saat ini, jumlahnya ditentukan oleh nilai t.
- Baris Baru: Menggunakan `System.out.println()`; untuk berpindah ke baris baru setelah mencetak semua bintang di baris tersebut.

b) Uraikan luaran yang dihasilkan

1.1 output :
1
3
5
7

1.3 output :

point a :

`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`

hasil running dari poin b :

`i = 1; j = 1`

`i = 1; j = 2`

`i = 2; j = 1`

Hasil running dari poin c :

`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

1.3 Masukan Input: 7

**

*

c) Screenshot/ Capture potongan kode dan hasil luaran

1.1 gambar contoh soal 1

```
1 public class ContohFor{
2     public static void main(String[] args) {
3         for (int y = 0; y <= 15; y++) { //ubah 1
4             if (y % 2 == 0) //ubah 2
5                 continue; //baris 1
6             else if (y == 9) //ubah 3
7                 break; //baris 2
8             else
9                 System.out.println(y + " ");
10        } }
11 }
```

Input/Output

Input Arguments

Output

```
1
3
5
7
```

1.2 gambar contoh soal 2

gambar continue pertama point a

```
1 public class ForBersarang {
2     public static void main(String[] args) {
3         pertama:
4         for( int i = 1; i < 5; i++) {
5
6             kedua:
7             for(int j = 1; j < 3; j++) {
8                 System.out.println("i = " + i + "; j = " + j);
9                 if ( i == 2)
10                     continue pertama; //ubah1
11             } } } }
```

Input/Output

Input Arguments

Output

```
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
```

Gambar break pertama point b

```

1- public class ForBersarang {
2-     public static void main(String[] args) {
3-         pertama:
4-         for( int i = 1; i < 5; i++) {
5-
6-             kedua:
7-             for(int j = 1; j < 3; j ++ ) {
8-                 System.out.println("i = " + i + "; j = " +j);
9-                 if ( i == 2)
10-                     break pertama;        //ubah1
11-             } } } }

```

Input/Output

Output

```

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

```

Gambar continue kedua point c

```

1- public class ForBersarang {
2-     public static void main(String[] args) {
3-         pertama:
4-         for( int i = 1; i < 5; i++) {
5-
6-             kedua:
7-             for(int j = 1; j < 3; j ++ ) {
8-                 System.out.println("i = " + i + "; j = " +j);
9-                 if ( i == 2)
10-                     continue kedua;        //ubah1
11-             } } } }

```

Input/Output

Output

```

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

```

1.3 gambar contoh soal 3

```

import java.util.Scanner;
public class ForBersarang {
    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);
        System.out.print("Masukan Input: ");
        int tinggi = input.nextInt();

        for (int t = tinggi; t >= 1; t--) {
            for (int s = t; s <= tinggi; s++) {
                System.out.print(" ");
            }
            for (int b = 1; b <= t; b++) {
                System.out.print("*");
            }
            System.out.println();
        }
        input.close();
    }
}

```

Input/Output

Output

```

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

```

[Nomor 1] Kesimpulan

1) Evaluasi

a) Apa konsekuensi dari skenario pemrograman ini?

1.1 Skenario program ini menunjukkan cara sederhana untuk mencetak bilangan ganjil, tetapi juga mengungkapkan beberapa keterbatasan dan konsekuensi yang mungkin muncul dalam penggunaannya. Mengatasi kekurangan ini dengan penanganan kesalahan, fleksibilitas logika, dan pengalaman pengguna yang lebih baik dapat membuat program ini lebih kuat dan efektif.

1.2 Setiap skenario menggunakan continue atau break memiliki konsekuensi yang signifikan pada logika program, output, dan keterbacaan. Sangat penting untuk merencanakan dan mendokumentasikan alur kontrol ini agar pengembang lain dan diri sendiri dapat memahami niat di balik penggunaan struktur kontrol tersebut.

1.3 Skenario program ini mengilustrasikan cara sederhana untuk mencetak pola bintang, tetapi juga menunjukkan beberapa batasan dan konsekuensi yang mungkin muncul dalam penggunaannya. Menambahkan penanganan kesalahan, meningkatkan fleksibilitas, dan memberikan instruksi lebih lanjut akan membuat program ini lebih kuat dan ramah pengguna.

Template Lembar Kerja Individu

Nama & NPM	Topik:	Tanggal:
Syifa Ariqah Pajriyanti (G1F024009)	FOR dan WHILE JAVA	5 Oktober 2024

[Nomor 2] Identifikasi Masalah:

1)Uraikan permasalahan dan variable

Contoh 4 :

```
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

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!

Jawab: perubahan nilai luaran yang dihasilkan pada contoh 4

1
2
3
4
5
6

Kegunaan break dan continue :

- Kegunaan break adalah untuk menghentikan loop sepenuhnya. Ketika break dieksekusi, program akan keluar dari loop terdekat dan melanjutkan eksekusi di luar loop.
- Kegunaan dari continue adalah digunakan untuk melewati sisa iterasi saat ini dan melanjutkan dengan iterasi berikutnya. Ketika i mencapai 4, pernyataan continue menginstruksikan program untuk melewati sisa blok loop dan langsung melanjutkan ke iterasi berikutnya. Pada iterasi ke 4, 4 akan dicetak tetapi increment i (menjadi 5) akan dilakukan setelah pernyataan continue, sehingga loop akan melanjutkan dengan i=5.

Contoh 5:

```
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

```

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!

Jawab: perubahan luaran setelah running adalah

```

0
5
10
15

```

Kegunaan % untuk angka yang berbeda pada perintah tersebut

- Memeriksa keterbagian : operator ini berguna untuk menentukan apakah suatu bilangan adalah kelipatan dari bilangan lain. Ini bias digunakan dalam berbagai situasi, seperti menentukan genap atau ganjil.
- Fleksibilitas : mengganti angka pada operator modulus memungkinkan untuk mengontrol pola output. Bias juga menggunakan angka lain seperti 4, 6, atau 7, tergantung pada kebutuhan logika yang diinginkan.

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`!

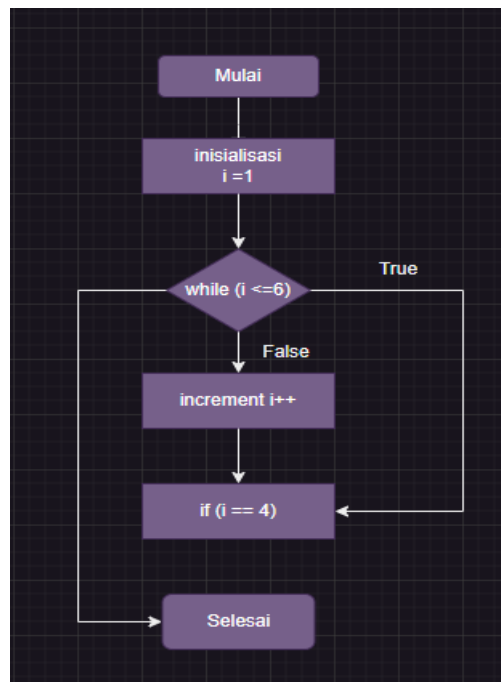
Jawab: bandingkan hasil luaran

`While (count < 0)` : tidak ada output yang dihasilkan, karena kondisi awal yang tidak terpenuhi.

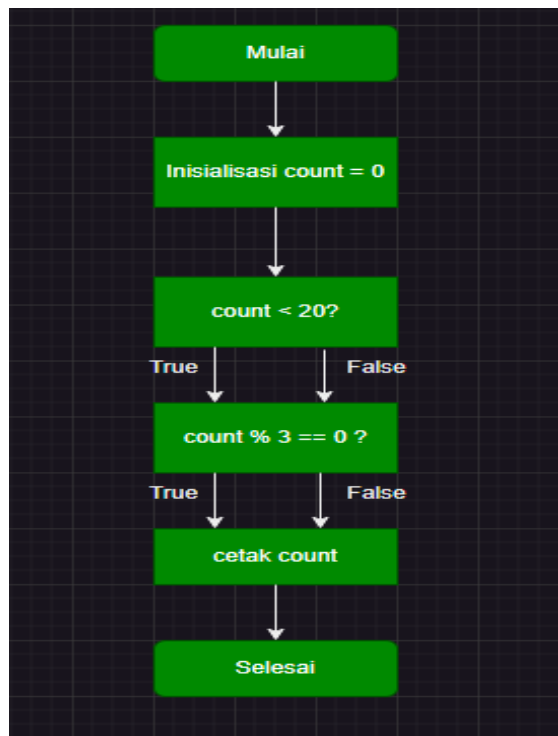
`Do...while (count < 0)`: mencetak 0 sekali, karena blok di dalam do dieksekusi sebelum kondisi diperiksa.

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

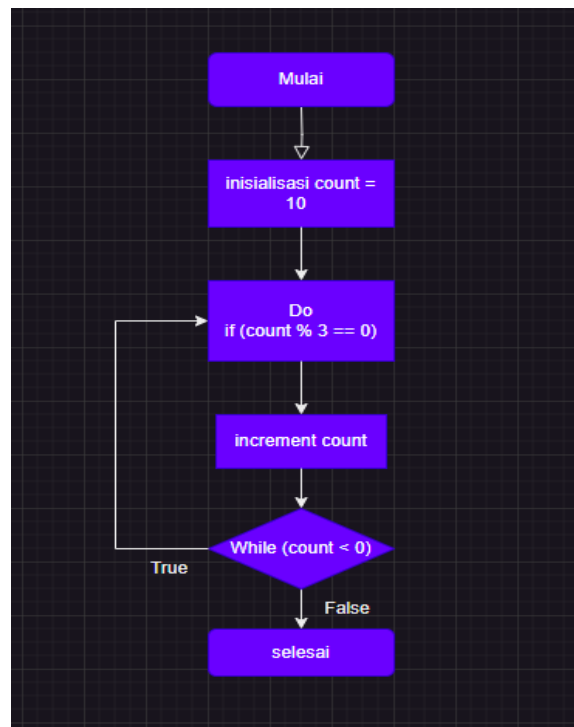
Flowchar latihan 2.1 :



Flowchart contoh 5 :



Flowchart 2.3 :



2) Rincikan sumber informasi yang relevan (buku / webpage)

[Video Materi 2 tentang WHILE – https://www.youtube.com/watch?v=ORA4JyJMFss](https://www.youtube.com/watch?v=ORA4JyJMFss)

[Nomor 2] Analisis dan Argumentasi

2.1

1) saya mengusulkan rancangan ini dapat diatasi dengan cara memberikan pendekatan alternatif untuk menangani kondisi ketika i mencapai 4. Menggunakan `continue` mengabaikan cetakan dan melanjutkan ke iterasi berikutnya, sedangkan menggunakan `break` menghentikan loop sepenuhnya.

2) Perbaiki kode program dengan cara

Perbaiki Logika: Solusi ini mengatasi permasalahan dengan menyertakan logika untuk menangani angka 4. Alih-alih mencetak angka 4, program akan melanjutkan ke angka berikutnya setelah menaikkan nilai i .

Keberlanjutan Loop: Penggunaan `continue` di sini tetap mematuhi struktur loop, tetapi dengan cara yang lebih jelas dan terarah. Tidak ada angka yang tercetak dua kali, dan iterasi dilanjutkan dengan benar.

Output yang Diharapkan: Luaran yang dihasilkan adalah 1, 2, 3, 5, dan 6, yang mengindikasikan bahwa angka 4 telah diabaikan, sesuai dengan tujuan yang lebih jelas.

2.2

1) saya mengusulkan rancangan ini dapat diatasi dengan cara memastikan output sesuai kriteria bahwa program hanya mencetak angka yang memenuhi syarat kelipatan 5. Hal ini sudah dilakukan dengan menggunakan `if (count % 5 == 0)`.

2) Analisis solusi, kaitkan dengan permasalahan.

Rancangan solusi ini bertujuan untuk memastikan kode berfungsi sesuai harapan dengan cara yang lebih jelas dan terstruktur. Dengan menambahkan komentar, memperbaiki format output, dan mempertimbangkan penanganan kasus tambahan, kita dapat meningkatkan kualitas dan keterbacaan kode. Solusi ini tidak hanya mencetak angka kelipatan 5 tetapi juga menyediakan konteks yang lebih baik bagi pengguna.

2.3

1) saya mengusulkan rancangan ini dapat diatasi dengan cara ubah kondisi loop: perlu mengubah kondisi pada while untuk memastikan loop akan terus berjalan hingga count mencapai batas tertentu. Misalnya, kita dapat menggunakan $\text{count} < 10$ untuk mencetak kelipatan 3 dari 0 hingga 9. Perbaiki struktur loop: seharusnya menggunakan loop while atau do-while yang sesuai. Dalam hal ini, jika kita ingin memastikan setidaknya satu iterasi dilakukan, do-while bisa digunakan.

2) solusi yang diusulkan secara efektif mengatasi masalah yang ada dalam kode asli dengan memperbaiki kondisi loop dan memastikan logika yang tepat untuk mencetak hasil yang diinginkan. Solusi ini tidak hanya memperbaiki kesalahan tetapi juga meningkatkan struktur dan keterbacaan kode, sehingga lebih mudah dipahami dan diadaptasi untuk tujuan yang lebih luas.

[Nomor 2] Penyusunan Algoritma dan Kode Program

1) Rancang desain solusi atau algoritma

2.1 Algoritma :

- Mulai
- Inisialisasi
- Mulai loop while
- Cek kondisi
- Akhiri loop
- Akhiri program
- Selesai

2.2 Algoritma :

- Inisialisasi
- Perulangan
- Selesai

2.3 Algoritma :

- Mulai
- Inisialisasi
- Mulai loop while
- Cek kondisi
- Akhiri loop
- Akhiri program selesai
- Selesai

2) Tuliskan kode program dan luaran

a) Beri komentar pada kode

2.1

- Inisialisasi Variabel: Menjelaskan bahwa variabel i diinisialisasi dengan nilai awal 1.
- Loop While: Menjelaskan kondisi loop dan tujuan dari loop tersebut.
- Mencetak Nilai: Menjelaskan apa yang dilakukan dalam setiap iterasi, yaitu mencetak nilai i .
- Menambahkan Nilai: Menjelaskan bahwa i ditambah 1 di setiap iterasi.
- Cek Kondisi: Menjelaskan kondisi di mana continue digunakan, meskipun di sini tidak mengubah perilaku program secara signifikan.

2.2 berikan komentar pada kode

- inisialisasi variabel: Menjelaskan bahwa count dimulai dari 0.
- Perulangan: Menunjukkan bahwa selama count kurang dari 20, blok kode di dalam while akan dieksekusi.
- Pemeriksaan kelipatan 5: Menggambarkan bahwa program memeriksa apakah count adalah

kelipatan 5 dengan menggunakan operator modulus (%).

- Menampilkan hasil: Jika kondisi kelipatan terpenuhi, program akan menampilkan nilai count.
- Increment: Menggambarkan langkah untuk menambah nilai count setiap iterasi agar perulangan bergerak maju.

2.3 Beri komentar pada kode

1. Inisialisasi: Komentar pada baris inisialisasi menjelaskan bahwa variabel count dimulai dari 0, yang merupakan titik awal untuk iterasi.
2. Mulai Loop: Komentar sebelum loop do-while menjelaskan bahwa kita akan memulai sebuah loop yang akan terus berjalan hingga kondisi tertentu dipenuhi.
3. Cek Kelipatan: Komentar di dalam blok if menjelaskan tujuan dari pemeriksaan, yaitu untuk mengecek apakah count adalah kelipatan dari 3. Ini memberikan kejelasan tentang logika di balik cetakan yang dihasilkan.
4. Cetak Nilai: Komentar pada pernyataan `System.out.println(count);` menjelaskan bahwa angka yang memenuhi syarat (kelipatan 3) akan dicetak.
5. Increment Count: Komentar sebelum `count++` menjelaskan bahwa kita perlu meningkatkan nilai count untuk iterasi berikutnya, memastikan bahwa loop tidak berjalan selamanya.
6. Kondisi Loop: Komentar pada `while (count < 10);` menjelaskan bahwa loop akan terus berjalan selama count kurang dari 10, memberikan batasan yang jelas pada iterasi.

b) Uraikan luaran yang dihasilkan

Luaran 2.1

1
2
3
4
5
6

Luaran 2.2

0
5
10
15

Luaran 2.3

0

c) Screenshot/ Capture potongan kode dan hasil luaran

Gambar 2.1



```
1 public class ContohWhile{
2 public static void main(String[] args) {
3     int i=1;
4     while(i<=6){
5         System.out.println(i);
6         i++;
7         if(i==4){
8             continue; //ubah1
9         }
10    }
11 }
12 }
```

Input/Output

Input Argu

Output

1
2
3
4
5
6
|

Gambar 2.2



```
1 public class WhileBersarang {
2 public static void main(String[] args) {
3     int count = 0; //ubah1
4     while (count < 20) {
5         if (count % 5 == 0) //ubah2
6             System.out.println(count);
7         count++;
8     }
9 }
10 }
11 }
```

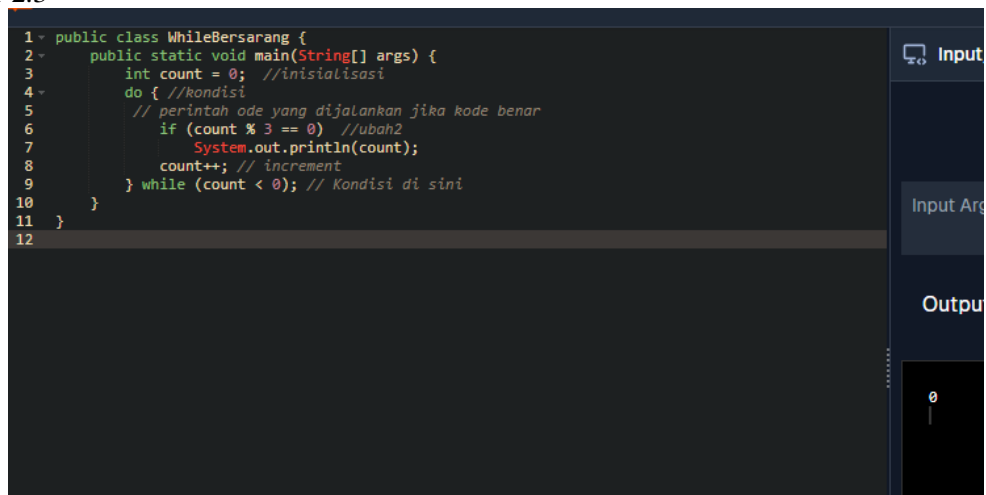
Input/Output

Input Argu

Output

0
5
10
15
|

Gambar 2.3



```
1 public class WhileBersarang {
2 public static void main(String[] args) {
3     int count = 0; //inisialisasi
4     do { //kondisi
5         // perintah ode yang dijalankan jika kode benar
6         if (count % 3 == 0) //ubah2
7             System.out.println(count);
8         count++; // increment
9     } while (count < 0); // Kondisi di sini
10 }
11 }
12 }
```

Input/Output

Input Argu

Output

0
|

[Nomor 2] Kesimpulan
<ol style="list-style-type: none">1. Analisa<ol style="list-style-type: none">a. Susunlah kesimpulan berdasarkan permasalahan, algoritma, dan kode program!<p>Jawab : program ini berhasil menyelesaikan permasalahan yang diajukan dengan cara yang terstruktur dan logis. Pemrograman berbasis kondisi dan perulangan yang digunakan dalam kode menunjukkan keterampilan dalam menyusun logika dan algoritma pemrograman. Program ini dapat menjadi dasar yang baik untuk eksplorasi lebih lanjut mengenai kontrol alur dalam pemrograman.</p>a) Apakah dasar alasan pengambilan keputusan Anda untuk kasus ini?<p>Jawab : Pengambilan keputusan dalam kasus ini didasarkan pada prinsip-prinsip dasar pemrograman, efisiensi, dan pentingnya keterbacaan kode. Dengan memfokuskan pada elemen-elemen ini, solusi yang dihasilkan tidak hanya memenuhi tujuan yang diinginkan, tetapi juga memberikan fondasi yang kuat untuk pemrograman lebih lanjut.</p>