

**LAPORAN PRAKTIKUM ALGORITMA DAN  
PEMROGRAMAN**

**PEKAN 6 PERULANGAN WHILE DAN DO-WHILE**



Disusun Oleh :

**DIKA GIOWANDA**

**NIM 2311533025**

Dosen Pengampu :

**DR. WAHYUDI, S.T, M.T**

Asisten Praktikum :

**JOVANTRI IMMANUEL GULO**

**FAKULTAS TEKNOLOGI INFORMASI**

**DEPARTEMEN INFORMATIKA**

**UNIVERSITAS ANDALAS**

**PADANG, NOVEMBER 2025**

## KATA PENGANTAR

Puji syukur penulis panjatkan atas ke hadirat Tuhan Yang Maha Esa atas rahmat, hidayah, dan karunia-Nya sehingga penulis dapat menyelesaikan penyusunan laporan praktikum ini mengenai perulangan *while* dan *do while* dalam bahasa pemrograman Java tepat pada waktunya. Shalawat serta salam senantiasa tercurah kepada Nabi Muhammad SAW, keluarga, sahabat, dan seluruh pengikutnya hingga akhir zaman.

Laporan ini disusun sebagai bentuk dokumentasi dan refleksi dari kegiatan praktikum yang telah dilakukan, dengan tujuan untuk memahami konsep dasar perulangan *while* dan *do while*. Dalam dunia pemrograman komputer, konsep perulangan merupakan salah satu fondasi utama yang memungkinkan suatu program menjalankan tugas secara berulang tanpa harus menulis kode yang sama berulang kali. Bahasa Java menyediakan berbagai struktur perulangan, dan di antara yang paling dasar namun sangat penting adalah perulangan *while* dan *do while*. Kedua struktur ini memiliki karakteristik unik *while* memeriksa kondisi sebelum mengeksekusi blok kode, sedangkan *do while* menjamin eksekusi minimal satu kali sebelum mengevaluasi kondisi. Pemahaman yang mendalam tentang perbedaan dan penerapan keduanya sangat membantu dalam menulis program yang efisien dan logis.

Penyusunan laporan ini dimaksudkan sebagai panduan pembelajaran bagi para pemula yang ingin memahami logika perulangan dalam Java, sekaligus sebagai bahan refleksi bagi penulis sendiri dalam memperdalam ilmu pemrograman. Semoga materi ini dapat memberikan manfaat dan menjadi langkah awal yang baik dalam perjalanan belajar pemrograman.

Penulis menyadari bahwa penyusunan laporan ini masih jauh dari sempurna. Berbagai kekurangan, baik dari segi penyajian, kedalaman materi, maupun teknis penulisan, mungkin masih terdapat di dalamnya. Oleh karena itu, kritik dan saran yang membangun sangat penulis harapkan demi perbaikan di masa mendatang.

Akhir kata, semoga ilmu yang disampaikan dalam materi ini dapat bermanfaat, membuka wawasan, dan menginspirasi pembaca untuk terus belajar dan berkarya dalam dunia pemrograman yang berbasis Java. Aamiin.

Padang, 2025

Penyusun

## .DAFTAR ISI

<b>KATA PENGANTAR.....</b>	ii
<b>DAFTAR ISI.....</b>	iii
<b>BAB 1 PENDAHULUAN.....</b>	1
1.1 Pengertian Praktikum.....	1
1.2 Tujuan Praktikum.....	1
1.3 Persyaratan Praktikum.....	1
1.4 Tempat dan Waktu Praktikum.....	2
1.5 Manfaat Praktikum.....	2
<b>BAB 2 PEMBAHASAN PRAKTIKUM.....</b>	3
2.1 Pengertian Perulangan While.....	3
2.2 Pengertian Perulangan Do While.....	5
2.3 Langkah Langkah Pengerjaan Praktikum Pekan 6.....	7
<b>BAB 3 PENUTUP.....</b>	26
3.1 Kesimpulan.....	26
3.2 Saran.....	26
<b>DAFTAR PUSTAKA.....</b>	27

# **BAB 1**

## **PENDAHULUAN**

### **1.1 Pengertian Praktikum**

Praktikum Java adalah kegiatan pembelajaran yang dilakukan di laboratorium komputer untuk mengasah keterampilan mahasiswa dalam memahami serta menerapkan konsep pemrograman Java. Kegiatan ini tidak hanya menekankan pada penguasaan teori,tetapi juga pada latihan penyusunan kode program, pengujian, hingga analisis hasil eksekusi. Praktikum dipandang sebagai wahana latihan yang menjembatani pemahaman konseptual dengan kemampuan teknis pemrograman.

### **1.2 Tujuan Praktikum**

Tujuan dari pelaksanaan praktikum ini antara lain sebagai berikut :

1. Membantu mahasiswa memahami konsep dasar pemrograman Java melalui penerapan langsung.
2. Melatih kemampuan menulis, mengompilasi, dan mengeksekusi program dengan mengikuti aturan sintaksis Java.
3. Meningkatkan keterampilan dalam memecahkan masalah (problem solving) dengan pendekatan algoritmik.
4. Membiasakan mahasiswa bekerja sistematis dalam menyusun laporan yang memuat analisis hasil praktikum.
5. Menanamkan sikap teliti, disiplin, serta tanggung jawab dalam melaksanakan kegiatan laboratorium.
6. Mengetahui dan mengaplikasikan if, else, if else, dan switch.

### **1.3 Persyaratan Praktikum**

Agar praktikum berjalan lancar, mahasiswa perlu memenuhi beberapa persyaratan berikut:

1. Telah mengikuti perkuliahan teori Pemrograman Java sebagai dasar pemahaman.

2. Membawa perlengkapan yang diperlukan, antara lain laptop atau komputer yang sudah terpasang *Java Development Kit* (JDK) dan *Integrated Development Environment* (IDE) yang direkomendasikan.
3. Mengikuti setiap sesi praktikum sesuai jadwal yang ditetapkan dan hadir minimal sesuai ketentuan program studi.
4. Mematuhi tata tertib laboratorium, termasuk menjaga keamanan data, perangkat, serta lingkungan kerja.
5. Menyusun laporan praktikum dengan format dan aturan yang telah ditetapkan dalam pedoman ini.

#### **1.4 Waktu dan Tempat Praktikum**

Pelaksanaan praktikum Java mengikuti kalender akademik yang berlaku pada program studi. Setiap sesi praktikum dilaksanakan sesuai jadwal yang ditentukan oleh dosen pengampu. Tempat kegiatan umumnya berlangsung di laboratorium komputer, namun pada kondisi tertentu dapat dilaksanakan secara mandiri dengan perangkat masing-masing, selama memenuhi syarat teknis yang ditetapkan.

#### **1.5 Manfaat Praktikum**

Manfaat praktikum bahasa Java meliputi kemampuan memahami dan menerapkan konsep dasar pemrograman seperti tipe data dan kontrol alur, melatih penggunaan kode yang modular dan dapat digunakan kembali, serta membangun pemahaman tentang sintaksis dan struktur *object oriented programming* (OOP) yang lebih mendalam. Selain itu, praktikum ini juga membantu membangun fondasi untuk mengembangkan aplikasi lintas platform yang lebih kompleks di berbagai bidang teknologi.

## BAB 2

### PEMBAHASAN PRAKTIKUM

#### 2.1 Pengertian Perulangan While

Perulangan *while* adalah struktur kontrol yang mengeksekusi blok kode berulang kali selama kondisi yang diberikan bernilai true (benar). Cara kerjanya adalah dengan memeriksa kondisi terlebih dahulu, jika true maka blok kode dieksekusi, dan jika false maka perulangan berhenti. Penggunaannya cocok untuk mengulang sesuatu yang jumlah iterasinya tidak diketahui sebelumnya. Contohnya di Java adalah mencetak angka 1 hingga 5 selama nilai variabel penghitung kurang dari atau sama dengan 5.

Pengertian bagian-bagian:

- 1) Kondisi *boolean*: *while* mengulang blok kode selama kondisi yang ditentukan bernilai benar (*true*).
- 2) Kontrol entri: Kondisi diuji terlebih dahulu sebelum blok kode dijalankan. Jika kondisi awalnya false, blok kode tidak akan pernah dieksekusi.
- 3) Jumlah iterasi tidak diketahui: Struktur ini sangat berguna ketika jumlah pengulangan tidak diketahui sebelumnya dan bergantung pada kondisi tertentu.

Cara Kerja:

- Cek Kondisi: Kontrol program pertama tama memeriksa kondisi yang ada di dalam kurung () .
- Eksekusi Kode: Jika kondisi bernilai *true* (benar), blok kode di dalam kurung kurawal {} akan dieksekusi.
- Ulangi: Setelah blok kode selesai dieksekusi, program kembali ke langkah 1 untuk memeriksa kondisi lagi.
- Keluar dari *Loop*: Jika kondisi bernilai *false* (salah), perulangan akan berhenti, dan program akan melanjutkan ke baris kode setelah blok *while*.

Contoh Penggunaan:

- Meminta input pengguna: Terus meminta pengguna memasukkan angka selama angka tersebut belum memenuhi kriteria tertentu (misalnya, memasukkan angka antara 1 dan 10).
- Proses yang bergantung pada kondisi: Mengulang suatu tindakan sampai ambang batas tertentu tercapai, seperti hingga pengguna menekan tombol keluar.
- Iterasi dengan jumlah tidak diketahui: Melanjutkan memproses data sampai tidak ada data lagi, meskipun jumlahnya tidak diketahui di awal.

Contoh pada Java:

Berikut adalah contoh sederhana yang mencetak angka dari 1 hingga 5:

```
public class WhileLoopExample {
    public static void main(String[] args) {
        int i = 1; // 1. Inisialisasi variabel penghitung

        while (i <= 5) { // 2. Kondisi: Loop berjalan selama i kurang dari
            atau sama dengan 5
            System.out.println(i); // 3. Aksi: Mencetak nilai i
            i++; // 4. Pembaruan: Menambah nilai i sebesar 1 di setiap
            iterasi
        }
    }
}
```

Penjelasan contoh:

1. *Inisialisasi*: Variabel i diinisialisasi dengan nilai 1.
2. Kondisi: while (*i* <= 5) mengecek apakah nilai i kurang dari atau sama dengan 5.
3. Aksi: Jika kondisi benar, program akan mencetak nilai i saat ini.
4. Pembaruan: i++ menambah nilai i sebesar 1. Proses ini berulang hingga kondisi *i* <= 5 menjadi false (ketika i mencapai 6).
5. Selesai.

## 2.2 Pengertian Perulangan Do While

Perulangan *do while* di Java adalah jenis perulangan yang mengeksekusi blok kode setidaknya sekali sebelum memeriksa kondisinya. Ini berbeda dengan perulangan *while* yang akan memeriksa kondisi terlebih dahulu sebelum mengeksekusi kode. Setelah kondisi dievaluasi, perulangan akan berlanjut selama kondisi tersebut bernilai benar (*true*).

Pengertian Bagian bagian *do while*:

Sintaks dasar perulangan do while terdiri dari tiga bagian utama:

- 1) *do*: Kata kunci ini menandai awal dari blok kode yang akan dieksekusi.
- 2) Blok Kode (*Statements*): Serangkaian instruksi di dalam tanda kurung kurawal {} yang akan dijalankan di setiap iterasi.
- 3) *while* (kondisi): Kata kunci ini muncul setelah blok kode. kondisi adalah ekspresi *boolean* (yang bernilai *true* atau *false*). Kondisi ini dievaluasi setelah blok kode dijalankan. Jika kondisi bernilai *true*, perulangan berlanjut, dan blok kode dijalankan lagi. Jika kondisi bernilai *false*, perulangan berakhir, dan eksekusi program berlanjut ke pernyataan berikutnya setelah perulangan.

Cara kerjanya:

1. Eksekusi pertama: Blok kode di dalam *do* akan dieksekusi secara langsung, tanpa memeriksa kondisi terlebih dahulu.
2. Pengecekan kondisi: Setelah blok kode selesai, kondisi yang ada di dalam *while* akan diperiksa.
3. Pengulangan (jika *true*): Jika kondisi bernilai benar (*true*), perulangan akan kembali ke awal blok *do* dan mengulanginya lagi.
4. Keluar dari perulangan (jika *false*): Jika kondisi bernilai salah (*false*), perulangan akan berhenti dan program akan melanjutkan ke baris kode setelah perulangan.

Contoh:

Berikut adalah contoh program Java yang menggunakan perulangan do-while untuk mencetak angka dari 1 hingga 5:

```
public class DoWhileDemo {  
    public static void main(String[] args) {  
        int hitung = 1; // Inisialisasi variabel hitung  
  
        do {  
            System.out.println("Hitungan: " + hitung); // Mencetak nilai  
hitung  
            hitung++; // Menambah nilai hitung (increment)  
        } while (hitung <= 5); // Kondisi perulangan  
    }  
}
```

Output dari contoh di atas:

Hitungan: 1

Hitungan: 2

Hitungan: 3

Hitungan: 4

Hitungan: 5

Penjelasan Kerja Contoh:

Berikut adalah cara kerja contoh program di atas langkah demi langkah:

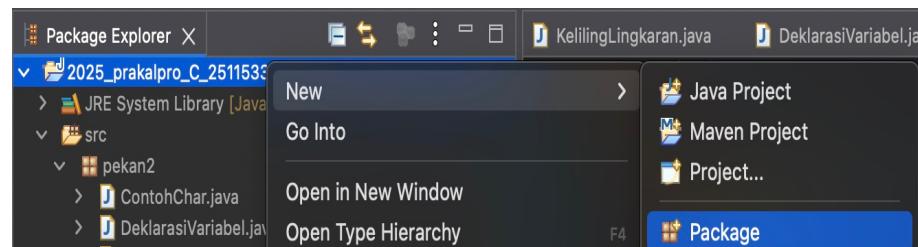
1. *Inisialisasi*: Variabel hitung *diinisialisasi* dengan nilai 1.
2. Eksekusi Blok *do*: Program masuk ke dalam blok *do* tanpa memeriksa kondisi terlebih dahulu.
3. *System.out.println("Hitungan: 1");* dicetak ke konsol.
4. hitung bertambah menjadi 2.
5. Pemeriksaan *while*: Kondisi (*hitung <= 5*) dievaluasi. Karena *2 <= 5* bernilai *true*, perulangan berlanjut.
6. Iterasi Kedua: Program kembali ke awal blok *do*.
7. *System.out.println("Hitungan: 2");* dicetak.
8. hitung bertambah menjadi 3.

9. Pemeriksaan *while*: Kondisi (*hitung <= 5*) dievaluasi. Karena  $3 \leq 5$  bernilai true, perulangan berlanjut.
10. Iterasi Ketiga, Keempat, dan Kelima: Proses ini berulang terus hingga *hitung* menjadi 6.
11. Iterasi Keenam (Akhir):
12. *hitung* bernilai 6.
13. Kondisi (*hitung <= 5*) dievaluasi. Karena  $6 \leq 5$  bernilai *false*, perulangan berakhir.
14. Selesai: Program melanjutkan eksekusi setelah perulangan *do while*.

### 2.3 Langkah Langkah Pengerjaan Praktikum Pekan 6

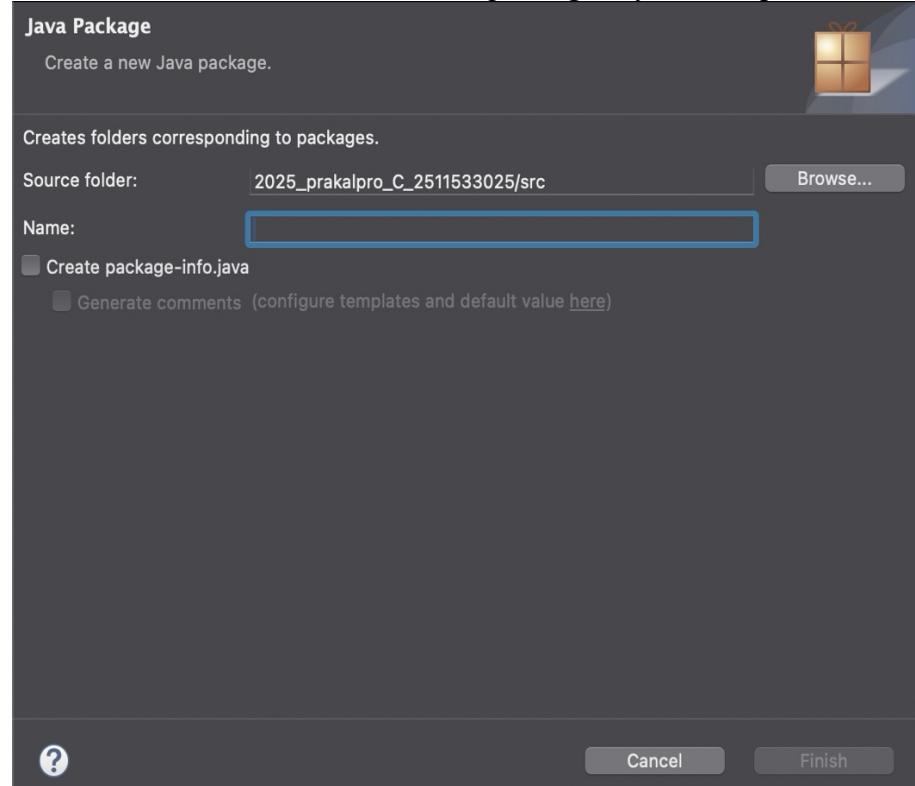
Sebelum ke langkah Langkah pengerjaan contoh projek, saya akan menjelaskan terlebih dahulu gimana caranya menambahkan projek nya terlebih dahulu :

1. Setelah kalian masuk ke eclipse java dan membuat folder file projek java, click kanan pada mouse kalian , kemudian pilih *new* terus pilih *package*.

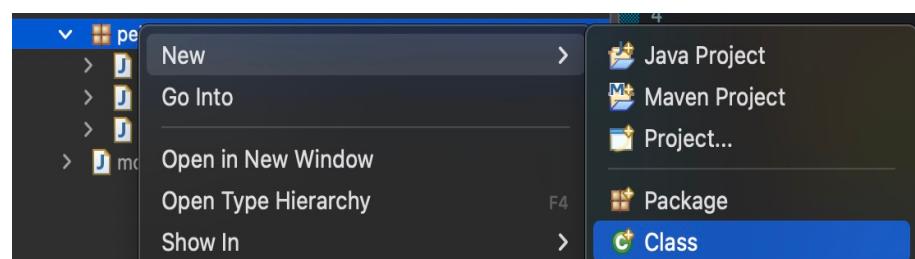


Setelah di click akan muncul halaman java *package*, dimana kita

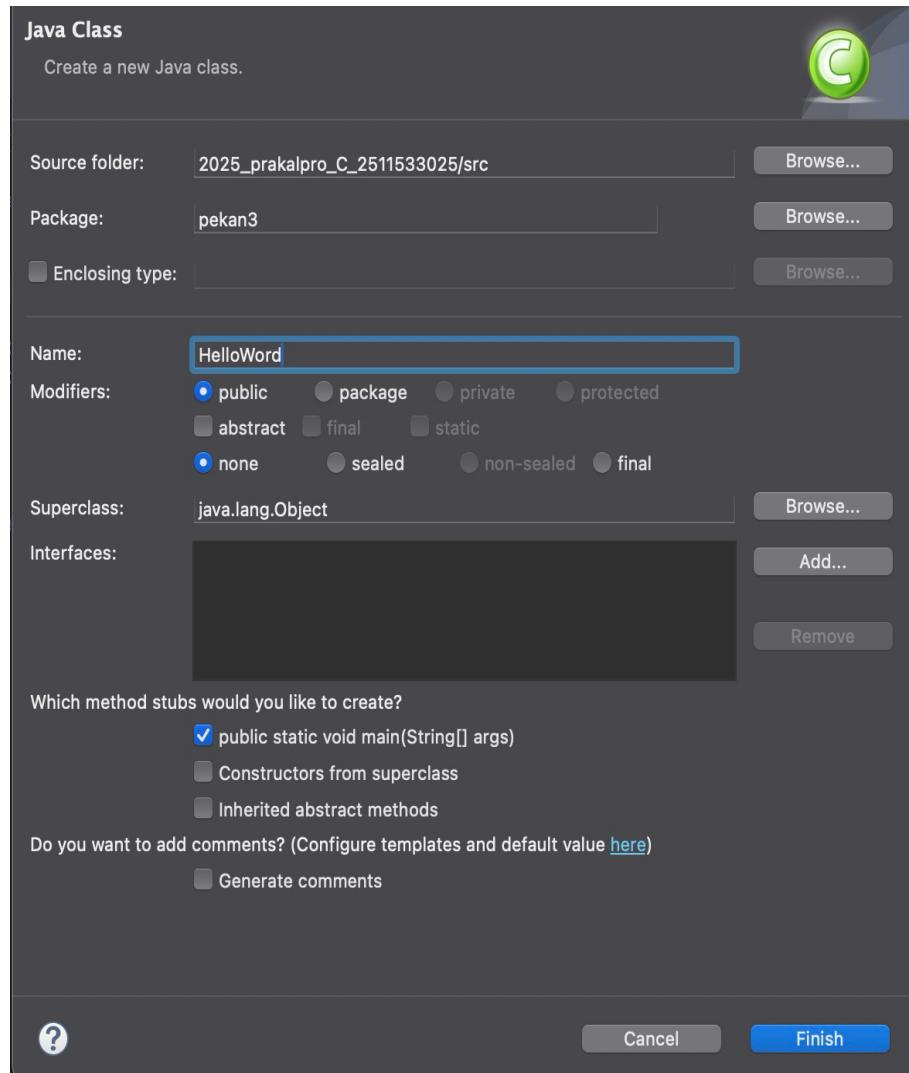
disuruh bikin nama untuk folder *package* nya, lalu pilih finish.



2. Jika folder *packagenya* sudah jadi, click kanan mouse pada folder *package* nya, lalu akan muncul pilihan, kalian pilih *new* kemudian pilih *class*.



Setelah di click akan muncul halaman java *class*, dimana kita disuruh untuk membuat nama *classnya* dibagian yang bertuliskan nama dan jangan lupa untuk centang pada bagian *public static void main*, lalu click finis.



Setelah selesai pembuatan folder *class* nya maka baru kita bisa membuat projek yang kita inginkan.

Catatan :

Jika kita ingin membuat projek baru lagi, maka cukup buat folder *class* yang baru.

- A. Baiklah karena proses menambahkan projek sudah dijelaskan, selanjutnya langkah langkah penggerjaan projek :

1. Membuat Perulangan While1

1. Tahap awal

Buat *class*, namakan sesuai dengan yang diperintahkan atau yang diinginkan, untuk format settingnya sesuai dengan format yang sudah disediakan atau sesuai intruksi yang diberikan. Jangan

sampai public classnya berbeda dengan nama folder *class* yang kita buat, karena merupakan kunci untuk mendeklarasikan sebuah kelas.

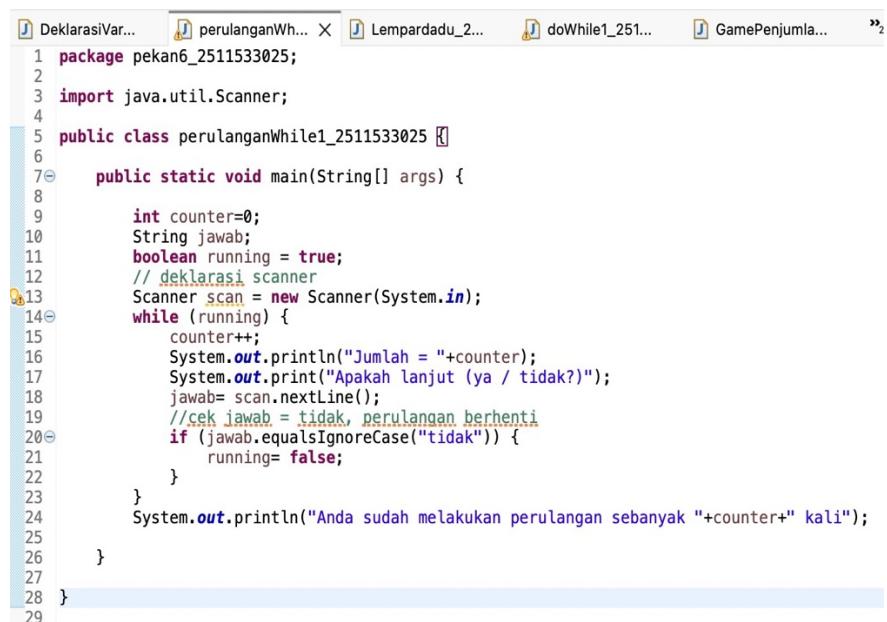
## 2. Tahap inti

- Biasanya kode *public static void main(String[] args)* akan muncul otomatis ketika membuat *class* baru, dimana berfungsi sebagai titik masuk utama untuk program Java yang memungkinkan JVM (*Java Virtual Machine*) untuk menjalankan aplikasi mandiri.
- Masukan kode *int counter = 0* untuk mendeklarasikan variabel *counter* dengan tipe data *integer* dan menginisialisainya dengan nilai 0.
- Masukan kode *String jawab* untuk mendeklarasikan variabel *jawab* dengan tipe data *string* untuk menyimpan input dari pengguna.
- Masukan kode *Boolean running = true* untuk mendeklarasikan variabel *running* dengan tipe data *Boolean* dan menginisialisasinya dengan nilai *true*, variabel ini digunakan sebagai kondisi untuk menjalankan atau menghentikan perulangan.
- Masukan kode *// deklarasi scanner* untuk menunjukkan bahwa program sedang menyiapkan objek *scanner* untuk digunakan.
- Masukan kode *Scanner scan = new Scanner(System.in)* untuk membuat objek *Scanner* baru, objek ini digunakan untuk membaca input dari pengguna, seperti teks atau angka, dari konsol atau *System.in..*
- Masukan kode *while (running)* karena merupakan awal dari sebuah *loop while*, *loop* ini akan terus berjalan selama kondisi di dalam tanda kurung, yaitu variabel *running*, bernilai *true*.
- Masukan kode *counter++* karena merupakan operator increment yang menambahkan nilai variabel *counter* sebanyak satu.
- Masukan kode *System.out.println("Jumlah = "+counter)* karena merupakan perintah untuk mencetak output ke konsol, teks "Jumlah = " digabungkan dengan nilai terbaru dari variabel *counter* sebelum dicetak.
- Masukan kode *System.out.println("Jumlah = "+counter)* untuk mencetak teks "Jumlah = " dan nilai dari variabel *counter* ke konsol dan akan menambahkan baris baru setelah teks dicetak, sehingga kursor akan pindah ke baris berikutnya.
- Masukan kode *jawab = scan.nextLine()* untuk membaca input berupa satu baris teks dari pengguna dan menyimpannya ke dalam variabel bernama *jawab*.

- Masukan kode `//cek jawab = tidak`, perulangan berhenti karena merupakan komentar yang menjelaskan bahwa jika nilai dari variabel jawab adalah "tidak", maka perulangan (*loop*) akan berhenti.
- Masukan kode `if (jawab.equalsIgnoreCase("tidak"))` karena merupakan pernyataan kondisional yang memeriksa apakah nilai dari variabel jawab sama dengan string "tidak", tanpa membedakan huruf besar atau kecil.
- Masukan kode `running = false` Jika kondisi di atas terpenuhi (yaitu, jika jawab berisi "tidak"), maka variabel running akan diatur menjadi false. Secara keseluruhan, kode ini menghentikan proses atau *loop* yang dikendalikan oleh variabel running ketika pengguna memberikan jawaban "tidak".
- Masukan kode `System.out.println("Anda sudah melakukan perulangan sebanyak "+counter+" kali")` untuk menampilkan teks ke konsol atau output standar.

### 3. Tahap akhir

Masukan kode `System.out.println` untuk melanjutkan proses atau mencetak programnya dimana akan memindahkan kursor ke garis baru. Usahakan semua sesuai dengan arahan yang diberikan, jika ada tanda silang maka ada kode yang tidak sesuai, untuk menghilangkan tanda silangnya kalian harus mencari letak kesalahannya , jika sudah ketemu segera diperbaiki, supaya hasilnya tidak eror dan jangan lupa setiap kode program per baris selalu diakhiri dengan tanda ( ; dan { ) tergantung apa isi kode programnya di baris tersebut jika sudah selesai memasukan kode program diakhiri dengan kurung kurawa tertutup. Berikut adalah gambar perulangan *while1*:



```

1 package pekan6_2511533025;
2
3 import java.util.Scanner;
4
5 public class perulanganWhile1_2511533025 {
6
7     public static void main(String[] args) {
8
9         int counter=0;
10        String jawab;
11        boolean running = true;
12        // deklarasi scanner
13        Scanner scan = new Scanner(System.in);
14        while (running) {
15            counter++;
16            System.out.println("Jumlah = "+counter);
17            System.out.print("Apakah lanjut (ya / tidak?)");
18            jawab= scan.nextLine();
19            //cek jawab = tidak, perulangan berhenti
20            if (jawab.equalsIgnoreCase("tidak")) {
21                running= false;
22            }
23        }
24        System.out.println("Anda sudah melakukan perulangan sebanyak "+counter+" kali");
25    }
26}
27
28}
29

```

#### 4. Hasil projek Perulangan While1.

Tapi ada perbedaan pada projek sekarang, karena *output* pada projek ini kita yang atur. Pada *output* ini apabila kita mengatakan ya, maka *outputnya* tidak akan berhenti dan jika kita langsung mengatakan tidak, maka *outputnya* akan langsung berhenti.



```
Jumlah = 1
Apakah lanjut (ya / tidak?)ya
Jumlah = 2
Apakah lanjut (ya / tidak?)ya
Jumlah = 3
Apakah lanjut (ya / tidak?)ya
Jumlah = 4
Apakah lanjut (ya / tidak?)ya
Jumlah = 5
Apakah lanjut (ya / tidak?)ya
Jumlah = 6
Apakah lanjut (ya / tidak?)ya
Jumlah = 7
Apakah lanjut (ya / tidak?)tidak
| Anda sudah melakukan perulangan sebanyak 7 kali
```

## 2. Membuat Projek Perulangan Do While1

### 1. Tahap awal

Buat *class*, namakan sesuai dengan yang diperintahkan atau yang diinginkan, untuk format setingginya sesuai dengan format yang sudah disediakan atau sesuai intruksi yang diberikan. Jangan sampai public *classnya* berbeda dengan nama folder *class* yang kita buat, karena merupakan kunci untuk mendeklarasikan sebuah kelas dan jangan lupa kasih kurung kurawa sebagai pembuka perintah.

### 2. Tahap inti

- Biasanya kode *public static void main(String[] args)* akan muncul otomatis ketika membuat *class* baru, dimana berfungsi sebagai titik masuk utama untuk program Java yang memungkinkan JVM (*Java Virtual Machine*) untuk menjalankan aplikasi mandiri.
- Masukan kode *Scanner console = new Scanner(System.in)* dimana *Scanner* merupakan kelas (*class*) di Java yang memungkinkan program membaca input dari berbagai sumber, seperti *keyboard* atau file sedangkan *console*

merupakan nama objek (variable) yang dibuat dari kelas *Scanner*. Lalu kode *new Scanner* merupakan perintah untuk membuat objek baru, lalu kode *System.in* untuk menentukan sumber input yang akan dibaca oleh objek *Scanner*, biasanya merujuk pada objek standar yaitu keyboard.

- Masukan kode *String phrase* berfungsi untuk mendeklarasikan sebuah variabel *String* bernama *phrase*.
- Masukan kode *do {* berfungsi menandai dimulainya blok kode untuk perulangan *do while*.
- Masukan kode *System.out.print("Input Password: ")* berfungsi untuk menampilkan teks "Input Password: " ke konsol atau layar tanpa membuat baris baru.
- Masukan kode *phrase = console.next()* kode ini berarti program akan menunggu pengguna memasukkan teks. Setelah pengguna mengetik sesuatu dan menekan *Enter*, metode *next()* akan membaca kata pertama yang diketik (hingga spasi pertama) dan menyimpannya ke dalam variabel bernama *phrase*.
- Masukan kode *while (!phrase.equals("abcd"))* dimana ! berfungsi membalikan nilai *boolean* dari kondisi berikutnya, lalu *phrase.equals("abcd")*: Ini adalah metode yang membandingkan string yang disimpan dalam variabel *phrase* dengan string "abcd". Metode ini akan mengembalikan *true* jika kedua *string* sama, dan *false* jika berbeda. Jadi kode *!phrase.equals("abcd")*: merupakan kombinasi berarti perulangan akan terus berjalan selama *string phrase* tidak sama dengan "abcd".

### 3. Tahap akhir

Masukan kode *system.out.println* untuk melanjutkan proses atau mencetak programnya dimana akan memindahkan kursor ke garis baru. Jika kita ingin *outputnya* menyamping maka kita gunakan *print*. Usahakan semua sesuai dengan arahan yang diberikan, jika ada tanda silang maka ada kode yang tidak sesuai, untuk menghilangkan tanda silangnya kalian harus mencari letak kesalahannya , jika sudah ketemu segera diperbaiki, supaya hasilnya tidak eror dan jangan lupa setiap kode program per baris selalu diakhiri dengan tanda ( ; dan { ) tergantung apa isi kode programnya di baris tersebut, jika sudah selesai memasukan kode program diakhiri dengan kurung kurawa tertutup. Berikut adalah gambar perulangan *do while* :

A screenshot of a Java code editor showing a project named 'doWhile1\_2511533025'. The code is a simple do-while loop that prompts the user for a password until they enter 'abcd'. The code is as follows:

```
1 package pekan6_2511533025;
2 import java.util.Scanner;
3 public class doWhile1_2511533025 {
4
5     public static void main(String[] args) {
6         Scanner console = new Scanner(System.in);
7         String phrase;
8         do {
9             System.out.print("Input Password: ");
10            phrase = console.next();
11        } while (!phrase.equals("abcd"));
12    }
13 }
14 }
15 }
```

#### 4. Hasil projek Perulangan Do While1.

Tapi ada perbedaan pada projek sekarang, karena *output* pada projek ini kita yang atur. Pada *output* ini apabila kita memasukan *input* berupa angka, maka *outputnya* tidak akan ada batasnya dan jika kita langsung memasukan *input* sesuai projek, maka *outputnya* akan langsung berhenti.

A screenshot of the Eclipse IDE's Console tab. The output shows the program prompting for input and accepting four different inputs: '1', '2', '3', and 'abcd'. The console tab has tabs for Problems, Javadoc, Declaration, and Console, with Console being the active tab.

```
Input Password: 1
Input Password: 2
Input Password: 3
Input Password: abcd
```

#### 3. Membuat Projek Game Penjumlahan

##### 1. Tahap awal

Buat *class*, namakan sesuai dengan yang diperintahkan atau yang diinginkan, untuk format setinggnya sesuai dengan format yang sudah disediakan atau sesuai intruksi yang diberikan. Jangan sampai *public class*nya berbeda dengan nama folder *class* yang kita buat, karena merupakan kunci untuk mendeklarasikan sebuah kelas dan jangan lupa kasih kurung kurawa sebagai pembuka perintah. Sedikit tambahan pada projek ini kita harus buat *importnya* sendiri.

##### 2. Tahap inti

- Biasanya kode *public static void main(String[] args)* akan muncul otomatis ketika membuat class baru, dimana

berfungsi sebagai titik masuk utama untuk program Java yang memungkinkan JVM (*Java Virtual Machine*) untuk menjalankan aplikasi mandiri.

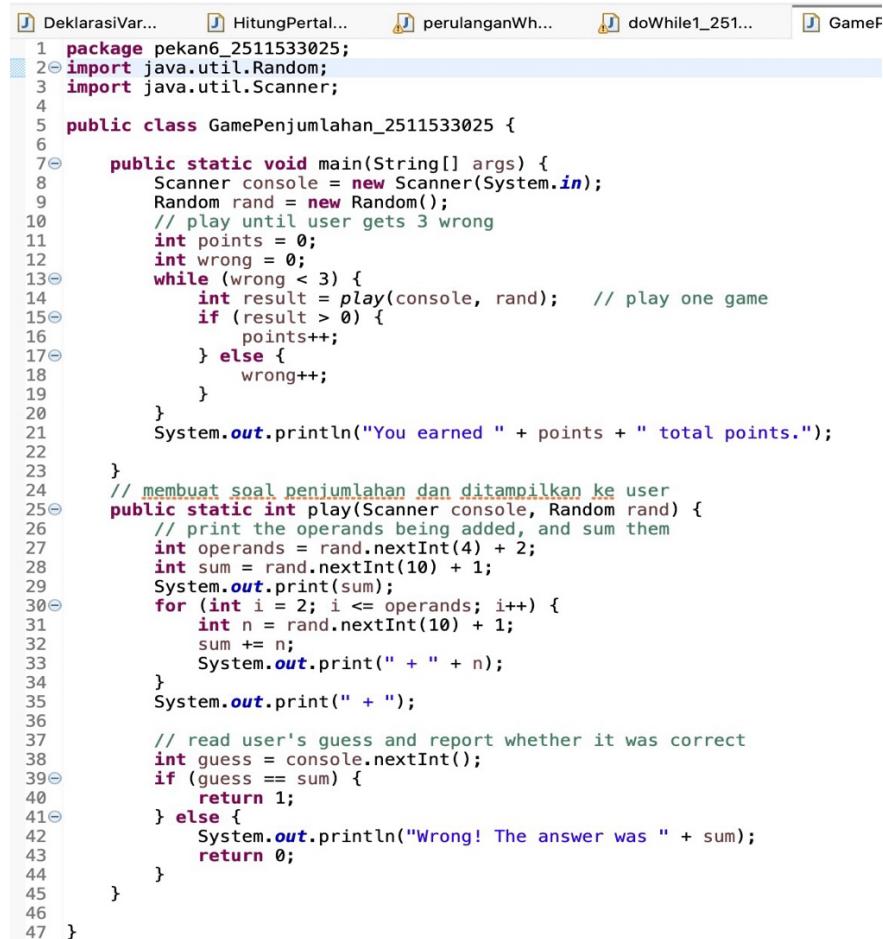
- Masukan kode `Scanner console = new Scanner(System.in)` dimana `Scanner` merupakan kelas (*class*) di Java yang memungkinkan program membaca input dari berbagai sumber, seperti `keyboard` atau file sedangkan `console` merupakan nama objek (*variable*) yang dibuat dari kelas `Scanner`. Lalu kode `new Scanner` merupakan perintah untuk membuat objek baru, lalu kode `System.in` untuk menentukan sumber input yang akan dibaca oleh objek `Scanner`, biasanya merujuk pada objek standar yaitu `keyboard`.
- Masukan kode `Random rand = new Random()` untuk membuat sebuah objek baru dari kelas `random` dimana kelas angka yang dihasilkan adalah acak.
- Masukan kode `// play until user gets 3 wrong` untuk memberikan penjelasan singkat tentang bagian kode tersebut.
- Masukan kode `int points = 0` untuk mendeklarasikan sebuah variabel dengan nama `points` dan memberikan nilai awal 0.
- Masukan kode `int wrong = 0` untuk mendeklarasikan sebuah variabel dengan nama `wrong` dan memberikan nilai awal 0.
- Masukan kode `while (wrong < 3)` untuk menjalankan blok kode yang ada di dalam kurung kurawal {} secara berulang-ulang selama kondisi (`wrong < 3`) bernilai benar (`true`).
- Masukan kode `int result = play(console, rand); // play one game` untuk memanggil sebuah fungsi bernama `play()`, dimana `int` menyimpan nilai yang dikembalikan oleh fungsi `play()`, dan bagian `//` menjelaskan bahwa kode ini digunakan untuk menjalankan satu permainan.
- Masukan kode `if (result > 0)` untuk memeriksa apakah nilai dari variabel `result` lebih besar dari 0.
- Masukan kode `points++` untuk menambahkan nilai 1 pada variabel `points`.
- Masukan kode `else` untuk menentukan blok kode yang akan dieksekusi jika kondisi dalam pernyataan `if` sebelumnya bernilai salah (`false`).
- Masukan kode `wrong++` untuk menambah nilai variabel `wrong` 1.
- Masukan kode `System.out.println("You earned " + points + " total points.")` untuk mencetak teks ke konsol atau layar. Dimana "You earned " dan " total points " sebagai *string* dan `points` sebagai nilai dari variabel.

- Masukan kode // membuat soal penjumlahan dan ditampilkan ke *user* untuk menjelaskan tujuan dari kode yang dibuat.
- Masukan kode *public static int play(Scanner console, Random rand)* untuk mendefinisikan sebuah metode (atau fungsi) bernama *play* yang dapat digunakan untuk menjalankan sebuah permainan atau proses yang memerlukan input dari pengguna dan elemen acak.
- Masukan kode // *print the operands being added, and sum them* untuk menjelaskan tujuan dari kode yang dibuat.
- Masukan kode *int operands = rand.nextInt(4) + 2* untuk untuk mendeklarasikan variabel integer bernama *operands* dan memberinya nilai acak antara 2 hingga 5.
- Masukan kode *int sum = ran.nextInt(10) + 1* untuk menghasilkan angka acak antara 1 hingga 10 dan menyimpannya dalam variabel bernama *sum*.
- Masukan kode *System.out.print(sum)* untuk menampilkan nilai dari variabel *sum* ke *konsol*.
- Masukan kode *for (int i = 2; i <= operands; i++)* untuk melakukan perulangan dengan variabel *i* yang dimulai dari 2 dan akan terus berlanjut hingga nilai *i* mencapai *operands*.
- Masukan kode *int n = rand.nextInt(10) + 1* untuk menghasilkan sebuah bilangan bulat acak dan menyimpannya ke dalam variabel.
- Masukan kode *sum += n* untuk menjumlahkan nilai *n* ke variabel *sum*.
- Masukan kode *System.out.print(" " + n)* untuk mencetak nilai variabel *n* ke *konsol* atau *output standar*.
- Masukan kode *System.out.print(" " + ")* untuk menampilkan karakter *+* ke *konsol*.
- Masukan kode // *read user's guess and report whether it was correct* untuk menjelaskan tujuan dari kode yang dibuat.
- Masukan kode *int guess = console.nextInt()* untuk membaca input berupa bilangan bulat (*integer*) dari *konsol* dan menyimpannya ke dalam variabel bernama *guess*.
- Masukan kode *if (guess == sum)* untuk mengecek apakah nilai dari variabel *guess* sama dengan nilai dari variabel *sum*. Kode *==* adalah operator perbandingan yang memeriksa apakah nilai di sisi kiri (*guess*) sama dengan nilai di sisi kanan (*sum*).
- Masukan kode *return 1* untuk mengakhiri eksekusi sebuah fungsi dan mengembalikan nilai 1.

- Masukan kode `} else {` untuk menentukan blok kode yang akan dieksekusi jika kondisi dalam pernyataan `if` sebelumnya bernilai salah (`false`).
- Masukan kode `System.out.println("Wrong! The answer was + sum)` untuk mencetak teks ke *konsol*.
- Masukan kode `return 0` untuk mengakhiri eksekusi program dan mengembalikan nilai 0 ke sistem operasi.

### 3. Tahap akhir

Masukan kode `system.out.println` untuk melanjutkan proses atau mencetak programnya dimana akan memindahkan kursor ke garis baru. Jika kita ingin *outputnya* menyamping maka kita gunakan *print*. Usahakan semua sesuai dengan arahan yang diberikan, jika ada tanda silang maka ada kode yang tidak sesuai, untuk menghilangkan tanda silangnya kalian harus mencari letak kesalahannya , jika sudah ketemu segera diperbaiki, supaya hasilnya tidak eror dan jangan lupa setiap kode program per baris selalu diakhiri dengan tanda ( ; dan { ) tergantung apa isi kode programnya di baris tersebut, jika sudah selesai memasukan kode program diakhiri dengan kurung kurawa tertutup. Berikut adalah gambar perulangan for 3 :



```

1 package pekan6_251153025;
2 import java.util.Random;
3 import java.util.Scanner;
4
5 public class GamePenjumlahan_2511533025 {
6
7     public static void main(String[] args) {
8         Scanner console = new Scanner(System.in);
9         Random rand = new Random();
10        // play until user gets 3 wrong
11        int points = 0;
12        int wrong = 0;
13        while (wrong < 3) {
14            int result = play(console, rand);    // play one game
15            if (result > 0) {
16                points++;
17            } else {
18                wrong++;
19            }
20        }
21        System.out.println("You earned " + points + " total points.");
22    }
23
24    // membuat soal penjumlahan dan ditampilkan ke user
25    public static int play(Scanner console, Random rand) {
26        // print the operands being added, and sum them
27        int operands = rand.nextInt(4) + 2;
28        int sum = rand.nextInt(10) + 1;
29        System.out.print(sum);
30        for (int i = 2; i <= operands; i++) {
31            int n = rand.nextInt(10) + 1;
32            sum += n;
33            System.out.print(" + " + n);
34        }
35        System.out.print(" + ");
36
37        // read user's guess and report whether it was correct
38        int guess = console.nextInt();
39        if (guess == sum) {
40            return 1;
41        } else {
42            System.out.println("Wrong! The answer was " + sum);
43            return 0;
44        }
45    }
46
47 }

```

#### 4. Hasil projek game penjumlahan.

Tapi ada perbedaan pada projek sekarang, karena *output* pada projek ini kita yang atur. Pada *output* ini apabila kita memasukan *input* angka yang nggak sesuai pada hasil penjumlahan, maka akan muncul dibawahnya jawaban yang benar. Disini kita hanya boleh salah sebanyak 3 kali, apabila kita menjawab salah sebanyak 3 kali akan muncul total *points* berdasarkan jawaban yang benar dan *outputnya* berhenti, maksudnya game telah selesai. Tapi apabila jawaban selalu benar soal akan terus muncul sampai batas yang kita tentukan berdasarkan projek yang kita buat.

```
Problems @ Javadoc Declaration Console X
<terminated> GamePenjumlahan_2511533025 [Java Application]
1 + 8 + 9 + 2 + 10 + 2
Wrong! The answer was 30
3 + 1 + 4
8 + 9 + 17
10 + 3 + 1 + 14
10 + 5 + 15
10 + 4 + 9 + 1 +
1
Wrong! The answer was 24
2 + 3 + 1
Wrong! The answer was 5
You earned 4 total points.
```

#### d. Membuat projek Lempar Dadu

##### 1. Tahap awal

Buat *class*, namakan sesuai dengan yang diperintahkan atau yang diinginkan, untuk format setingginya sesuai dengan format yang sudah disediakan atau sesuai intruksi yang diberikan. Jangan sampai public classnya berbeda dengan nama folder *class* yang kita buat, karena merupakan kunci untuk mendeklarasikan sebuah kelas dan jangan lupa kasih kurung kurawa sebagai pembuka perintah. Sedikit tambahan pada projek ini kita harus buat *importnya* sendiri.

##### 2. Tahap inti

- Biasanya kode *public static void main(String[] args)* akan muncul otomatis ketika membuat *class* baru, dimana berfungsi sebagai titik masuk utama untuk program Java yang memungkinkan JVM (*Java Virtual Machine*) untuk menjalankan aplikasi mandiri.

- Masukan kode `Random rand = new Random()` untuk membuat objek bernama `rand` dari kelas `Random`. Kode ini menghasilkan angka acak.
- Masukan kode `int tries = 0` untuk mendeklarasikan dan menginisialisasi variabel `tries` yang dirancang khusus untuk menyimpan angka bulat, dan memberinya nilai awal nol.
- Masukan kode `int sum = 0` untuk mendeklarasikan variabel bernama `sum` dengan tipe data integer (bilangan bulat) dan menginisialisasi nilainya menjadi 0.
- Masukan kode `while (sum != 7) {` berfungsi agar perulangan akan terus berjalan selama nilai dari variabel `sum` tidak sama dengan 7.
- Masukan kode `// roll the dice once` untuk menjelaskan tujuan dari kode yang dibuat.
- Masukan kode `int dadu1 = rand.nextInt(6) + 1` untuk menghasilkan angka acak yang mensimulasikan hasil lemparan dadu bersisi enam dimana hasil akhir disimpan dalam variabel integer bernama `dadu1`.
- Masukan kode `int dadu2 = rand.nextInt(6) + 1` untuk menghasilkan angka acak yang mensimulasikan hasil lemparan dadu bersisi enam dimana hasil akhir disimpan dalam variabel integer bernama `dadu2`.
- Masukan kode `sum = dadu1 + dadu2` untuk menghitung jumlah dari dua variabel.
- Masukan kode `System.out.println(dadu1 + " + dadu2 + " = " + sum)` untuk mencetak hasil penjumlahan dua buah dadu ke *konsol*.
- Masukan kode `tries++` untuk menambah nilai variabel bernama `tries` sebanyak satu.
- Masukan kode `System.out.println("You won after 11 11 + tries! ")` untuk mencetak teks ke *konsol*.

### 3. Tahap akhir

Masukan kode `System.out.println` untuk melanjutkan proses atau mencetak programnya dimana akan memindahkan kursor ke garis baru. Jika kita ingin *outputnya* menyamping maka kita gunakan `print`. Usahakan semua sesuai dengan arahan yang diberikan, jika ada tanda silang maka ada kode yang tidak sesuai, untuk menghilangkan tanda silangnya kalian harus mencari letak kesalahannya , jika sudah ketemu segera diperbaiki, supaya hasilnya tidak eror dan jangan lupa setiap kode program per baris selalu diakhiri dengan tanda ( ; dan { ) tergantung apa isi kode programnya di baris tersebut, jika sudah selesai memasukan kode program diakhiri dengan kurung kurawa tertutup. Berikut adalah gambar lempar dadu :

```

1 package pekan6_2511533025;
2 import java.util.Random;
3
4 public class Lempardadu_2511533025 {
5
6     public static void main(String[] args) {
7         Random rand = new Random();
8         int tries = 0;
9         int sum = 0;
10    while (sum != 7) {
11        // roll the dice once
12        int dadu1 = rand.nextInt(6) + 1;
13        int dadu2 = rand.nextInt(6) + 1;
14        sum = dadu1 + dadu2;
15        System.out.println(dadu1 + " + " + dadu2 + " = " + sum);
16        tries++;
17    }
18    System.out.println("You won after " + tries + " tries!");
19
20}
21
22}
23

```

#### 4. Hasil projek lempar dadu.

Tapi ada perbedaan pada projek sekarang, karena *output* yang dikeluarkan *random*, namanya juga game lempar dadu otomatis tergantung keberuntungan yang kita dapat.

```

Problems @ Javadoc Declaration Console X
<terminated> Lempardadu_2511533025 [Java Application] /App
2 + 5 = 7
You won after 1 tries!

```

```

Problems @ Javadoc Declaration Console X
<terminated> Lempardadu_2511533025 [Java Application] /App
1 + 2 = 3
3 + 6 = 9
5 + 6 = 11
6 + 2 = 8
4 + 3 = 7
You won after 5 tries!

```

#### e. Membuat projek Sentinel Loop

##### 1. Tahap awal

Buat *class*, namakan sesuai dengan yang diperintahkan atau yang diinginkan, untuk format setinggnya sesuai dengan format yang sudah disediakan atau sesuai intruksi yang diberikan. Jangan sampai *public class*nya berbeda dengan nama folder *class* yang kita buat, karena merupakan kunci untuk mendeklarasikan sebuah kelas

dan jangan lupa kasih kurung kurawa sebagai pembuka perintah. Sedikit tambahan pada projek ini kita harus buat *importnya* sendiri.

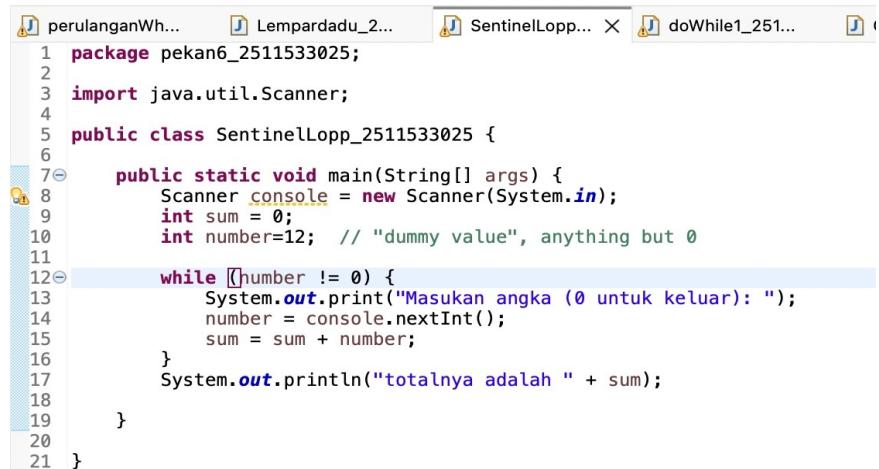
### 2. Tahap inti

- Biasanya kode `public static void main(String[] args)` akan muncul otomatis ketika membuat *class* baru, dimana berfungsi sebagai titik masuk utama untuk program Java yang memungkinkan JVM (*Java Virtual Machine*) untuk menjalankan aplikasi mandiri.
- Masukan kode `Scanner console = new Scanner(System.in)` untuk membuat objek Scanner bernama *console* yang digunakan untuk membaca input dari pengguna melalui konsol (keyboard).
- Masukan kode `int sum = 0` untuk mendeklarasikan variabel bernama *sum* dengan tipe data integer (bilangan bulat) dan menginisialisasi nilainya menjadi 0.
- Masukan kode `int number=12 // "dummy value", anything but 0` untuk mendeklarasikan sebuah variabel *integer* bernama *number* dan menginisialisasikannya dengan nilai 12, dan kode `//` untuk menjelaskan tujuan dari kode yang dibuat.
- Masukan kode `while (number != 0) {` untuk memulai blok kode yang akan terus dieksekusi secara berulang selama kondisi yang ditentukan di dalamnya bernilai benar (*true*).
- Masukan kode `System.out.print("Masukan angka (0 untuk keluar): ")` untuk menampilkan teks tertentu ke *konsol* atau layar *output* standar.
- Masukan kode `number = console.nextInt()` untuk membaca input bilangan bulat (*integer*) yang dimasukkan oleh pengguna melalui *konsol* dan menyimpannya ke dalam variabel bernama *number*.
- Masukan kode `sum = sum + number` untuk menambahkan nilai dari variabel 'number' ke variabel 'sum'.
- Masukan kode `System.out.println("totalnya adalah " + sum)` untuk menampilkan teks tertentu ke *konsol*.

### 3. Tahap akhir

Masukan kode `system.out.println` untuk melanjutkan proses atau mencetak programnya dimana akan memindahkan kursor ke garis baru. Jika kita ingin *outputnya* menyamping maka kita gunakan *print*. Usahakan semua sesuai dengan arahan yang diberikan, jika ada tanda silang maka ada kode yang tidak sesuai, untuk menghilangkan tanda silangnya kalian harus mencari letak kesalahannya , jika sudah ketemu segera diperbaiki, supaya hasilnya tidak eror dan jangan lupa setiap kode program per baris selalu diakhiri dengan tanda ( ; dan { ) tergantung apa isi kode programnya di baris tersebut, jika sudah selesai memasukan kode program

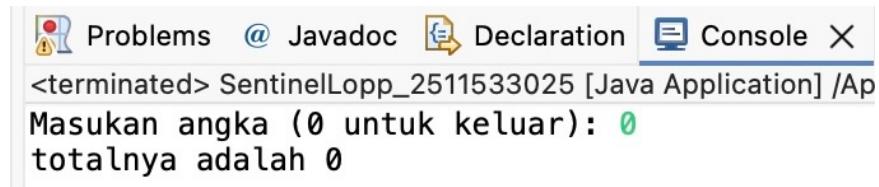
diakhiri dengan kurung kurawa tertutup. Berikut adalah gambar Sentinel Loop :



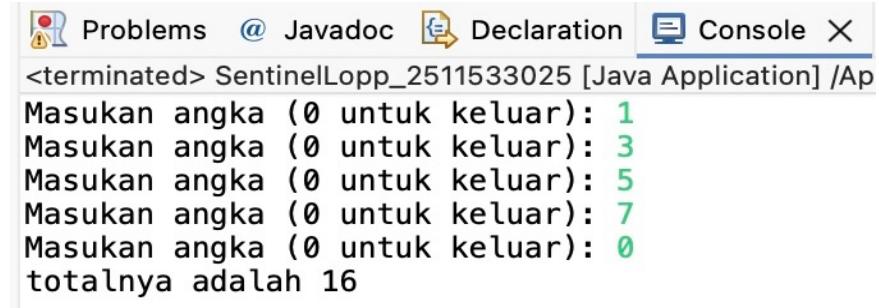
```
1 package pekan6_2511533025;
2
3 import java.util.Scanner;
4
5 public class SentinelLopp_2511533025 {
6
7     public static void main(String[] args) {
8         Scanner console = new Scanner(System.in);
9         int sum = 0;
10        int number=12; // "dummy value", anything but 0
11
12        while (!number != 0) {
13            System.out.print("Masukan angka (0 untuk keluar): ");
14            number = console.nextInt();
15            sum = sum + number;
16        }
17        System.out.println("totalnya adalah " + sum);
18    }
19 }
20 }
```

##### 5. Hasil projek Sentinel Loop.

Tapi ada perbedaan pada projek sekarang, karena *output* pada projek ini kita yang atur, apabila kita memasukan angka selain 0 *outputnya* tidak akan berhenti akan terus lanjut, dan setelah beberapa angka lalu kalian masukan 0, *output* yang keluar adalah total dari angka yang kita masukan. Dan apabila kita langsung memasukan angka 0, *output* dari program akan langsung selesai.



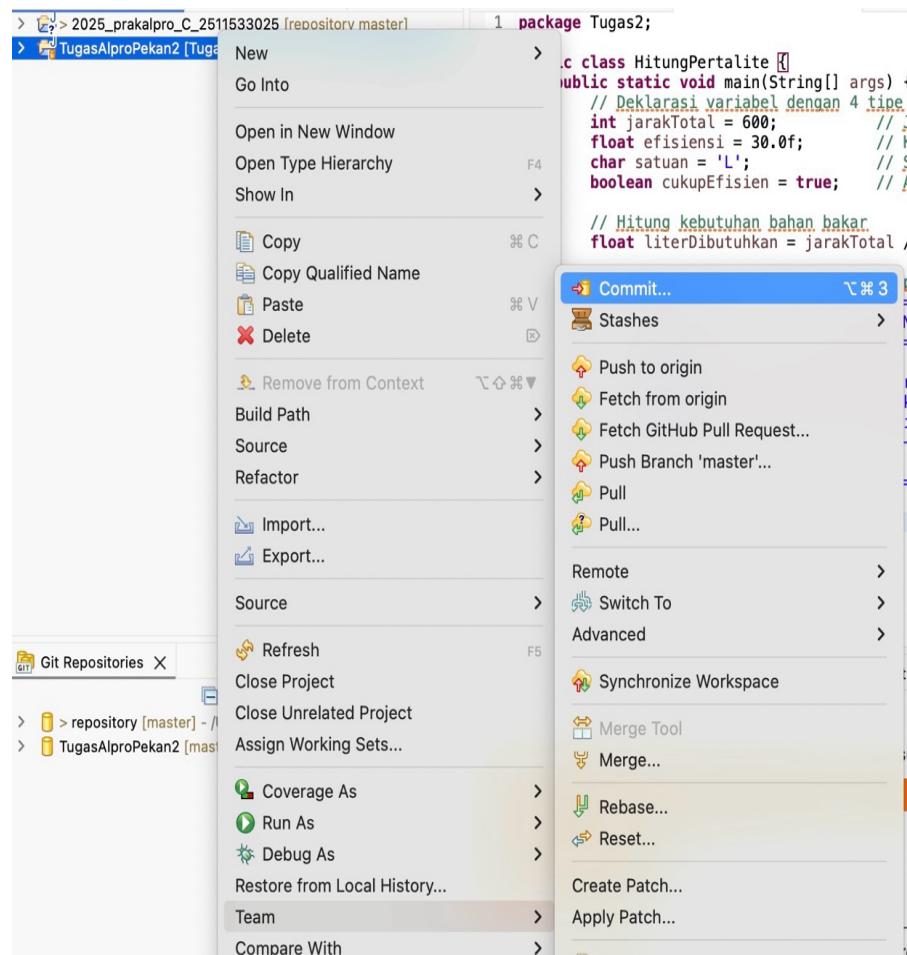
```
<terminated> SentinelLopp_2511533025 [Java Application] /Ap
Masukan angka (0 untuk keluar): 0
totalnya adalah 0
```



```
<terminated> SentinelLopp_2511533025 [Java Application] /Ap
Masukan angka (0 untuk keluar): 1
Masukan angka (0 untuk keluar): 3
Masukan angka (0 untuk keluar): 5
Masukan angka (0 untuk keluar): 7
Masukan angka (0 untuk keluar): 0
totalnya adalah 16
```

## B. Langkah Penyimpanan

1. Sebelum kita masuk ke langkah, kita harus buat akun github dulu. Jika sudah buat akun github, baru bisa kita simpan di githubnya. Selanjutnya ikutin langkah ini.Tekan kanan mouse pada folder projek, setelah itu pilih team, terus ke *commit*.



2. Lalu tekan tombol plus 2/double plus warna hijau.



3. Lalu masukan pesan pada kolom *commit message*.

### Commit Message

i Unborn branch: this commit will create the branch 'master'.

- Setelah itu tekan *commit* and pus, lalu ikutin arahan umtuk memasukan nama dan pw akun git hub kalian, maka projek kalian sudah tersimpan di akun github kalian.



- Projek yang sudah disimpan di akun projek.



DikaSiNpc/2025\_prakalpro\_C\_251153  
3025

- Gambar projek apabila sudah masuk di akun github.

A screenshot of a GitHub repository page. The repository name is 'DikaSiNpc/2025\_prakalpro\_C\_251153'. The left sidebar shows the file structure: '2025\_prakalpro\_C\_251153' contains 'src', '.settings', 'pekan1', 'pekan2', 'pekan3', 'pekan4', and 'pekan5'. 'pekan5' contains several Java files like 'PerulanganFor1\_251153025.java', 'nestedFor0\_251153025.java', etc. The right panel shows the commit history for 'src/pekan5\_251153025'. There are 10 commits from 'fe29f8a' 2 days ago, all titled 'Praktikum pekan5\_251153025' and dated '2 days ago'.

Name	Last commit message	Last commit date
..	Praktikum pekan5_251153025	2 days ago
PerulanganFor1_251153025.java	Praktikum pekan5_251153025	2 days ago
PerulanganFor4_251153025.java	Praktikum pekan5_251153025	2 days ago
nestedFor0_251153025.java	Praktikum pekan5_251153025	2 days ago
nestedFor1_251153025.java	Praktikum pekan5_251153025	2 days ago
nestedFor2_251153025.java	Praktikum pekan5_251153025	2 days ago
perulanganFor2_251153025.java	Praktikum pekan5_251153025	2 days ago
perulanganFor3_251153025.java	Praktikum pekan5_251153025	2 days ago

Sedikit tambahan jika ingin cek hasil program nya tekan tombol run warna hijau.



Tapi pada program yang sekarang berbeda dari sebelumnya karena pada program ini ada yang inputnya kita masukan angka atau huruf sesuai keinginan kita pada kolom *console*, tapi ingat harus sesusi perintah kode yang kita masukan.

## BAB 3

### PENUTUP

#### 3.1 Kesimpulan

Berdasarkan pelaksanaan praktikum mengenai perulangan *while* dan *do while* dalam bahasa pemrograman Java, dapat disimpulkan bahwa kedua struktur perulangan ini sangat berguna dalam menangani kasus-kasus yang memerlukan eksekusi berulang dengan jumlah iterasi yang tidak tetap atau tidak diketahui sebelumnya. Perulangan *while* mengevaluasi kondisi terlebih dahulu sebelum menjalankan blok perintah, sehingga jika kondisi tidak terpenuhi sejak awal, blok kode tidak akan dieksekusi sama sekali. Sebaliknya, perulangan *do while* menjamin bahwa blok kode di dalamnya akan dijalankan minimal satu kali karena pengecekan kondisi dilakukan setelah eksekusi. Melalui praktikum ini, kami memahami perbedaan mendasar antara kedua jenis perulangan tersebut, serta mampu menerapkannya dalam berbagai skenario pemrograman, seperti validasi input pengguna, menampilkan deret angka, atau mengulang proses hingga kondisi tertentu terpenuhi. Secara keseluruhan, penguasaan konsep perulangan *while* dan *do while* merupakan dasar penting dalam pemrograman Java yang mendukung pembuatan program yang efisien, interaktif, dan adaptif terhadap berbagai kondisi. Dan saya sangat senang bisa mengikuti praktikum ini, karena menambah wawasan ilmu pengetahuan.

#### 3.2 Saran

Sebaiknya dalam praktikum selanjutnya, diberikan lebih banyak contoh soal sederhana yang menunjukkan perbedaan jelas antara *while* dan *do while*, serta latihan untuk menghindari kesalahan umum seperti *infinite loop*, agar pemahaman konsep perulangan menjadi lebih kuat dan mudah diterapkan.

## **DAFTAR PUSTAKA**

- Sumber daring ( website ):

[1] Geekster, "Java While Loop," 2025. [Daring]. Tersedia pada: <https://www.geekster.in/articles/java-while-loop/>. [Diakses: 07-Nov-2025].

[2] ScholarHat, "While Loop in Java," 2025. [Daring]. Tersedia pada: <https://www.scholarhat.com/tutorial/java/while-loop-in-java#>. [Diakses: 07-Nov-2025].

[3] Programiz, "Java do...while Loop," 2025. [Daring]. Tersedia pada: <https://www.programiz.com/java-programming/do-while-loop#>. [Diakses: 07-Nov-2025].

[4] Oracle, "The while and do-while Statements," 2025. [Daring]. Tersedia pada: <https://docs.oracle.com/javase/tutorial/java/nutsandbolts/while.html>. [Diakses: 07-Nov-2025].

[5] Edureka, "Java Do While Loop," 2025. [Daring]. Tersedia pada: <https://www.edureka.co/blog/java-do-while-loop/#>. [Diakses: 07-Nov-2025].