LAPORAN PRAKTIKUM ALGORITMA PEMROGRAMAN

PEKAN KE - 6

(WHILE LOOP, DO WHILE LOOP, SENTINEL LOOP, METHOD)



**Oleh :** **ABDUR ROHIM** **NIM 2411533015**

**MATA KULIAH ALGORITMA DAN PEMROGRAMAN**

**DOSEN PENGAMPU :**

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

**FAKULTAS TEKNOLOGI INFORMASI** **DEPARTEMEN INFORMATIKA**

**UNIVERSITAS ANDALAS**

# PADANG

**2024**

1. **Pendahuluan**

## While loop

## *While loop* merupakan salah satu bentuk perulangan dalam pemrograman yang berfungsi untuk mengeksekusi sekumpulan perintah selama suatu kondisi masih bernilai benar (True). Perulangan ini akan terus berlanjut selama kondisi tersebut terpenuhi dan akan berhenti ketika kondisi menjadi salah (False).

## 2.Sentinel Loop

*Sentinel loop* adalah jenis perulangan (loop) dalam pemrograman yang bergantung pada nilai tertentu (disebut sentinel) untuk menentukan kapan perulangan harus berhenti. Sentinel bertindak sebagai "penanda akhir," sehingga perulangan akan terus berjalan hingga nilai sentinel terdeteksi.

## 3.Do While Loop

## *Do-while loop* merupakan jenis perulangan dalam pemrograman yang menjamin blok kode dijalankan minimal satu kali, tanpa memandang kondisi awal. Setelah menjalankan blok kode pertama, kondisi akan dievaluasi untuk memutuskan apakah perulangan akan diteruskan.

## 4.Method

*Method* merupakan serangkaian instruksi atau fungsi yang didefinisikan dalam sebuah kelas dalam pemrograman berorientasi objek (OOP). Method ini digunakan untuk menggambarkan aksi atau perilaku yang dapat dilakukan oleh objek yang dibuat dari kelas tersebut

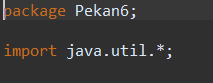
**B.Tujuan**

Tujuan dari pelaksanaan praktikum ini adalah untuk memahami dan mengimplementasikan dasar-dasar dari *while loop, sentinel loop,* dan *do while loop* pada bahasa pemrograman Java.

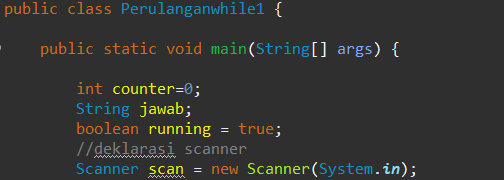
**C.Langkah-Langkah Pengerjaan**

## a.While Loop

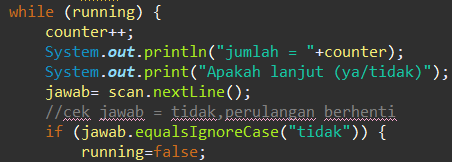
* 1. Buat package dan class baru. Berikan nama package sebagai pekan6 dan nama class sebagai perulanganWhile1
  2. Import *Scanner* untuk memungkinkan input dari pengguna



* 1. Inisialisasikan varibel yang akan digunakan yaitu varibel “counter” dengan *data type* int, variabel “jawab” dengan *data type* String, variabel ”running” dengan *data type* Boolean. Kemudian inisialisasikan *Scanner* dengan memberikan nama variabel “scan”.



* 1. Kemudian, tuliskan *while loop* dengan menulis “while” yang diikuti dengan kondisi yang diperlukan di dalam tanda kurung. Dalam kurung kurawal tuliskan perintah untuk menghitung iterasi dan menampilkan pesan di terminal/layar output.
  2. Tambahkan *Scanner* untuk menerima input dari pengguna dan memeriksa apakah perulangan harus berhenti.



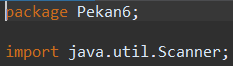
Kemudian, setelah perulangan selesai, tampilkan jumlah iterasi. System.***out***.println("Anda sudah melakukan perulangan sebanyak "+ counter

+ " kali");

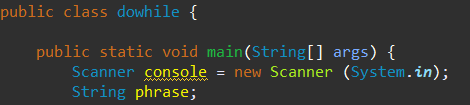
* 1. Klik run untuk menjalani program

## b.Do While Loop

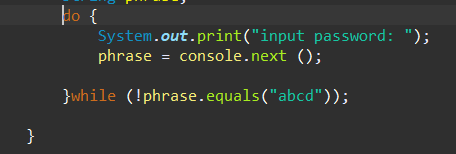
* 1. Buat class baru, berikan nama class sebagai “doWhile1”.
  2. Import *Scanner* dari java.util.Scanner



* 1. Inisialisasikan varibel phrase dengan *data type* String. Kemudian inisialisasikan *Scanner* dengan nama “console”



* 1. Gunakan *do-while loop* untuk meminta input password. Gunakan varibel “phrase” untuk meminta input *string* dan menyimpannya di dalam variabel tersebut. Program akan terus meminta input hingga pengguna memasukkan password yang sesuai yaitu, “abcd”.

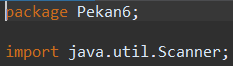


* 1. Klik run untuk menjalani program

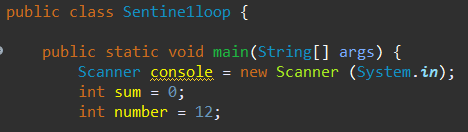
## c.Sentinel Loop

1.Buat class baru. Berikan nama class sebagai SentinelLoop.

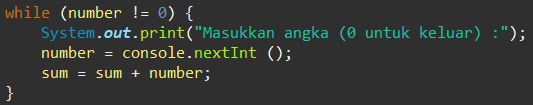
2.Import *Scanner* dari java.util.Scanner



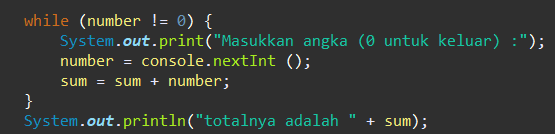
3.Inisialisasikan variabel yang diperlukan yaitu, “sum” dan “num” dengan *data type* int. “sum” diinisialisasikan menjadi 0, dan “number” dengan angka apa saja selain 0.



4.Tuliskan struktur *while loop* untuk memproses input selama nilai sentinel yaitu 0 belum dimasukkan.



5.Setelah perulangan selesai, tampilkan total dari semua angka yang dimasukkan

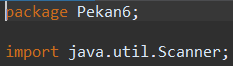


6.Klik run untuk menjalani program

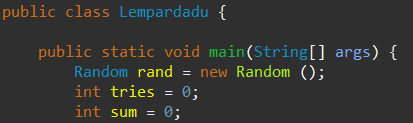
## d.Game Lempar Dadu

1.Buat class baru dengan nama Lempardadu.

2.Import *Random* dari java.util untuk membuat angka acak.

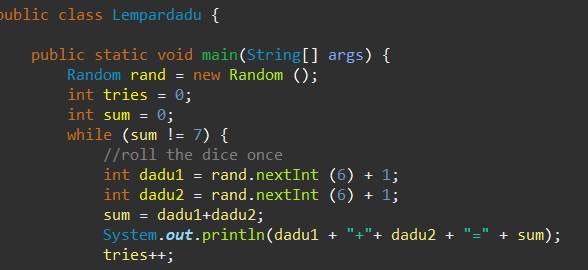


3.Deklarasikan variabel yang diperlukan yaitu, “int tries” untuk menghitung jumlah percobaan. “int sum” untuk menyimpan hasil penjumlahan kedua dadu, dan “Random rand” sebagai *generator* angka acak.



4.Buat perulangan *while* yang terus berjalan sampai nilai sum sama dengan 7.

5.Kemudian, di dalam *while loop* asilkan dua angka acak dari 1 hingga 6 untuk mewakili dua dadu. Jumlahkan kedua angka tersebut dan simpan dalam “sum”. Tampilkan hasil lemparan dadu ke layar. Tingkatkan nilai “tries” setiap kali perulangan dijalankan.



6.Setelah keluar dari perulangan, tampilkan jumlah percobaan yang diperlukan untuk mendapatkan angka 7

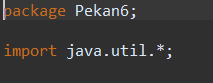


7.Klik run untuk menjalani program

## e.Method

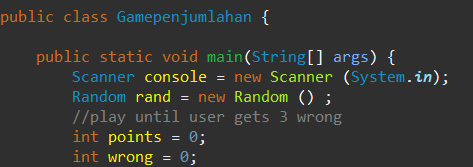
1.Buat class baru dengan nama GamePenjumlahan.

2.Import *Scanner* dari java.util.Scanner dan Random dari java.util.Random



3.Inisialisasikan *Scanner* dengan nama “console”. Kemudian inisialisasikan Random dengan nama “rand”

4.Kemudian, inisialisasikan variabel “points”, “wrong” dengan *data type* int dan nilai 0.

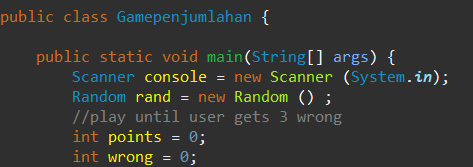


5.Tuliskan *while loop* dengan kondisi True jika variabel “wrong” lebih kecil dari

3. Kemudian inisialisasikan variabel “result” dengan *data type* int dan *assign*

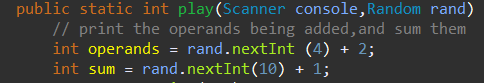
dengan *public class* “play”.

6.Kemudian, gunakan *if statement* dengan kondisi True jika “result” lebih besar dari 0. Di dalam *if statement,* ingkrementasikan variabel “points” jika kondisi True, dan ingkrementasikan variabel “wrong” jika kondisi False. Setelah keluar dari perulangan, outputkan jumlah poin.

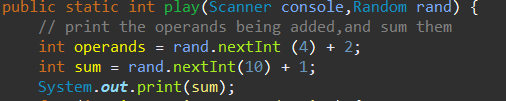


7.Setelah itu, inisialisasikan *public static int* di luar *public static void* dengan nama “play” dengan parameter *Scanner* console, dan *Random* rand.

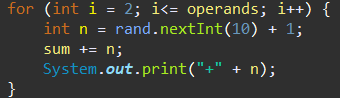
8.Kemudian, di dalam *public class* “play”, inisialisasikan variabel “operands” dengan *data type* int, dan *assign* ke dalam variabelnya rand.nextInt(4) + 2. Setelah itu, inisialisasikan variabel “sum” dengan *data type* yang sama, kemudian *assign* rand.nextInt(10) + 1.



9.Setelah itu, outputkan “sum”



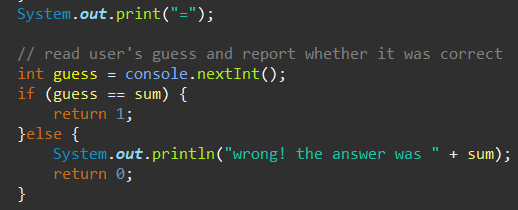
10. Tuliskan *for loop* yang dimulai dari 2 dengan kondisi lebih kecil sama dengan “operands”. Dalam *for loop,* inisialisasikan variabel “n” dengan *data type* int dan *assign-*kan rand.nextInt(10) + 1. Kemudian, ingkerementasikan variabel “sum” dengan variabel “n”. Outputkan variabel “n” dengan tanda “+”.



11.Kemudian, di luar *for loop,* outputkan tanda “=”. Setelah itu inisialisasikan variabel “guess” dengan *data type* int dan *assign-*kan variabel tersebut dengan *Scanner* untuk menerima input integer.

* 1. Setelah itu, gunakan *if/else statement* untuk memeriksa apakah jawaban *user*

sesuai dengan variabel “sum”. Jika benar, maka *return* 1, jika salah, *return* 2.



**C.Kesimpulan**

Dari praktikum yang telah dilakukan, maka dapat disimpulkan bahwa pada bahasa pemrograman Java, *while loop, do-while loop*, dan *sentinel loop* memiliki peran penting dalam menjalankan tugas yang bersifat repetitif dengan fleksibilitas tertentu. *While loop* memastikan perulangan berjalan selama kondisi bernilai benar, *do-while loop* menjamin eksekusi blok kode setidaknya satu kali sebelum memeriksa kondisi, sedangkan *sentinel loop* berhenti berdasarkan nilai khusus yang menjadi tanda (*sentinel*). Selain itu, implementasi **method** membantu membagi program menjadi bagian-bagian kecil yang lebih terstruktur, memungkinkan penggunaan ulang kode, dan meningkatkan keterbacaan program.