LAPORAN PRATIKUM PEKAN 6
ALGORITMA DAN PEMROGRAMAN

"PERULANGAN WHILE"

Dosen Pengampu:

DR. Wahyudi, S.T., M.T.

Asisten Lab:

Muhammad Zaki Al Hafiz

Disusun Oleh:

Hamdi Sidqi Alifi

2511531009

Fakultas Teknologi Informasi

Departemen Informatika

Universitas Andalas

2025

KATA PENGANTAR

Puji dan syukur senantiasa penulis panjatkan kepada Allah SWT yang telah melimpahkan rahmat dan karunia-Nya, sehingga penulis dapat menyelesaikan laporan ini guna memenuhi laporan pratikum mata kuliah Algoritma Pemorograman, dengan judul: “Perulangan While”. Laporan ini berisikan hasil analisis dari praktikum pemograman java penulis di Praktikum Kelas D yang dilaksanakan pada hari Rabu, tanggal 5 November 2025.

Pada kesempatan ini, Penulis ingin mengucapkan terima kasih kepada Bapak DR. Wahyudi. .S.T.M.T, dosen pengampu mata kuliah Algoritma Pemograman, yang telah memberikan tugas ini. Penulis juga ingin mengucapkan terima kasih kepada pihak-pihak yang telah mendukung serta membantu penulis dalam penyelesaian laporan pratikum Algoritma Pemrograman.

Penulis sadar bahwa laporan ini masih memiliki beberapa kekurangan dikarenakan terbatasnya pengalaman dan pengetahuan yang penulis miliki. Oleh karena itu, penulis harap adanya bentuk saran dan kritik yang membangun dari berbagai pihak.

Akhir kata, penulis berharap laporan ini dapat bermanfaat bagi perkembangan dunia pendidikan.

DAFTAR ISI

BAB I
PENDAHULUAN

1.1 Latar Belakang

Pada pertemuan sebelumnya kita telah mempelajari mengenai konsep perulangan *for-loop* mengguanakan *syntax for*. Namun, bagaimana jika jumlah perulangan yang diperlukan belum diketahui atau jumlah perulangan didasari pada input yang akan diberikan oleh pengguna?

Permasalahan tersebut dapat diatasi dengan mengguanakan *syntax while*, karena dengan mengguanakan *syntax* ini, *user* (pengguna) hanya harus menyatakan *conditional statement*nya saja.

1.2 Tujuan

1. Menjelaskan konsep perulangan *while-loop* dalam bahasa pemrograman Java.
2. Menjelaskan penerapan konsep perulangan *while-loop* dalam bahasa pemrograman Java.

1.3 Manfaat Pratikum

1. Memahami konsep perulangan *while-loop* dalam bahasa pemrograman Java.
2. Mampu memahami dan menerapkan konsep perulangan *while-loop* dalam bahasa Pemrograman Java.

BAB II
PEMBAHASAN

2.1 Teori

While-loop digunakan saat kita tidak/belum mengetahui berapa kali baris kode tersebut akan dijalankan. Dalam menggunakan *while-loop* kita hanya memerlukan 1 statement, yang mana sebagai berikut :

```
while (conditional) {
    // your program here
    }
```

Dimana *conditional* adalah sebuah *statement* yang berfungsi sebagai indikator jalannya program. Dengan kata lain, selama nilai *conditional statement* ini terpenuhi, maka program tersebut akan terus berjalan.

2.2 Kode Pemrograman

1. Program perulanganWhile1

```java
package pekan6_2511531009;
import java.util.Scanner;
public class perulanganWhile1_2511531009 {

    public static void main(String[] args) {

        int counter = 0;
        String jawab;
        boolean running = true;
        //deklarasi scanner
        Scanner scan = new Scanner(System.in);
        while (running) {
            counter++;
            System.out.println("jumlah = "+counter);
            System.out.println("Apakah lanjut? (ya / tidak)");
            jawab = scan.nextLine();
            //cek jawab = tidak, perulangan berhenti
            if (jawab.equalsIgnoreCase("tidak")) {
                running = false;
            }
        }
        scan.close();
        System.out.println("Anda sudah melakukan perulangan sebanyak "+counter+" kali");
    }

}
```

Penjelasan :

1. Dengan menggunakan *syntax while* pada *line* (12), program akan membaca apakah running bernilai *true*, jika ya maka lanjut perulangan
2. Di dalam perulangan *while* terdapat program jumlah dan pertanyaan apakah pengguana ingin melanjutkan program(lihat *line* (12))
3. Jawaban diberikan oleh pengguna, dan hanya membaca “tidak” selain dari *string* itu maka dianggap seperti *string* “ya”

Output :

```
jumlah = 1
Apakah lanjut? (ya / tidak)
ya
jumlah = 2
Apakah lanjut? (ya / tidak)
ya
jumlah = 3
Apakah lanjut? (ya / tidak)
tiDak
Anda sudah melakukan perulangan sebanyak 3 kali
```

Dapat diperhatikan bahwa disini penulis tadak sempurna dalam mengetik kata “tidak”(huruf kecil semua) karena penulis menggunakan *sytax equalsIgnoreCase* pada *line* (18).

2. Program Lempardadu

```java
package pekan6_2511531009;
import java.util.Random;
public class Lempardadu_2511531009 {

    public static void main(String[] args) {
        Random rand = new Random();
        int tries = 0;
        int sum = 0;
        while (sum != 7) {
            // roll the dice once
            int dadu1 = rand.nextInt(6) + 1;
            int dadu2 = rand.nextInt(6) + 1;
            sum = dadu1 + dadu2;
            System.out.println(dadu1 + " + "+ dadu2 + " = " + sum);
            tries++;
            }
            System.out.println("You won after "+tries+" tries");
    }

}
```

Penjelasan

a. Pada *line* (9), kita menyatakan bahwa jika nilai variabel *sum* tidak 7 maka program akan terus berlanjut, keberlanjutan tersebut ditandai dengan adanya penggunaan *syntax while* pada *line* tersebut. Perlu diperhatikan bahwa variabel *sum* harus diberikan nilai selayaknya yang penulis lakukan pada *line* (8).

b. Nilai *sum* tersebut didapatkan dari hasil penjumlahan 2 variabel (*dadu1* dan *dadu2*). Dengan tiap-tiap variabel mensimulasikan bagimana peluang dadu bekerja dengan menggunakan salah satu utilitas *java* yaitu *Random*. Karena anggota ruang sampel dadu ada enam, yaitu {1,2,3,4,5,6}, maka pada line penarikan angka acak kita jumlahkan dengan 1, karena komputer menghitung dari 0.

c. Setiap lemparan dua mata dadu akan dihitung dalam 1 kali cobaan. Diinisialisasikan pula variabel *tries* yang nilaiya selalu bertambah dalam tiap kali lemparan. Dan pada *line* (17) akan mencetak sebuah *line* berapa kali kita melakukan lemparan guna mencari jumlah 7 dari dua dadu.

Output :

```
2 + 4 = 6
4 + 5 = 9
4 + 4 = 8
2 + 5 = 7
You won after 4 tries
```

```
4 + 2 = 6
3 + 4 = 7
You won after 2 tries
```

```
4 + 3 = 7
You won after 1 tries
```

Seperti yang dapat dilihat pada hasil output, program akan terus berjalan selama angka 7 belum didapatkan dari hasil pelemparan dadu. Jika pemjulahan telah mencapai angka 7 pada pelemparan pertama, maka perulangan while akan langsung berhenti.

3. Program GamePenjumlahan

```
1  package pekan6_2511531009;
2  import java.util.*;
3  public class GamePenjumlahan_2511531009 {
4
5      public static void main(String[] args) {
6          Scanner console = new Scanner(System.in);
7          Random rand = new Random();
8          // play until user gets 3 wrong
9          int points = 0;
10         int wrong = 0;
11         while (wrong < 3) {
12             int result = play(console, rand);   // play one game
13             if (result > 0) {
14                 points++;
15             }
16             else {
17                 wrong++;
18             }
19         }
20         System.out.println("You earned "+points+" total points.");
21     }
```

Pada program ini menggunakan 2 *package* yaitu *Random* dan *Scanner*, namun kita bisa membungkusnya dalam satu *line* saja, seperti yang penulis lakukan pada *line* (2).

Dengan menggunakan *while-loop*, kita bisa membuat perulangan selama nilai *conditional statement*nya terpenuhi. Pada program ini program akan terus berjalan selama *wrong* (variabel untuk indikasi kesalahan) masih belum bernilai 3. Untuk menaikkan nilai *wrong* kondisi *if* pada line (13) tidak boleh terpenuhi, yang berupa nilai variabel *result* yang didapatkan dari fungsi *play(console, rand).*

```
22      // membuat soal penjumlahan dan ditampilkan ke user
23      public static int play(Scanner console, Random rand) {
24          // print the operands being added, and sum them
25          int operands = rand.nextInt(4) + 2;
26          int sum = rand.nextInt(10) + 1;
27          System.out.print(sum);
28          for (int  i = 2; i <= operands; i++) {
29              int n = rand.nextInt(10) + 1;
30              sum += n;
31              System.out.print(" + "+ n);
32          }
33          System.out.print(" = ");
34
35          // read user's guess and report wheter it was correct or not
36          int guess = console.nextInt();
37          if (guess == sum) {
38              return 1;
39          }
40          else {
41              System.out.println("Wrong! The answer was "+ sum);
42              return 0;
43          }
44      }
45  }
```

Disini kita menggunakan pemanggilan *method function*, dan *function* yang dipanggil ialah play(console, rand) yang telah kita buat pada *line* (12) sebelumnya. Disini akan memprogram suatu baris penjumlahan, jika penjumlahan kita benar maka *point* akan bertambah karena nilai return adalah *true* (1), jika tidak maka akan *false* (0)

```
10 + 9 + 3 = 22
7 + 7 + 7 + 8 + 9 = 38
5 + 7 + 1 + 3 = 16
1 + 4 + 3 = 8
3 + 4 + 6 = 12
Wrong! The answer was 13
9 + 9 + 10 + 2 + 10 = 3
Wrong! The answer was 40
10 + 9 + 8 = 3
Wrong! The answer was 27
You earned 4 total points.
```

4. Program SentinelLoop

```java
package pekan6_2511531009;
import java.util.Scanner;
public class SentinelLoop_2511531009 {

    public static void main(String[] args) {
        Scanner console = new Scanner(System.in);
        int sum = 0;
        int number = 12;    // "dummy value", anything is allowed except 0

        while (number != 0) {
            System.out.print("Masukkan angka (0 untuk keluar): ");
            number = console.nextInt();
            sum = sum + number;

        }
        console.close();
        System.out.println("totalnya adalah "+ sum);
    }

}
```

Pada program ini, kita bertujuan untuk menciptakan kalkulator penjumlahan, jika angka yang diinput bukanlah angka nol maka kalkulasi akan terus berjalan hingga pengguna memasukkan angka nol.

Sebelum itu nilai *number* harus dimasukkan suatu value terlebih dahulu sebelum digunakan untuk menjumlahkan nilai, karena nilai *number* berpengaruh pada *conditional statement* pada *line* (10).

```
Masukkan angka (0 untuk keluar): 10
Masukkan angka (0 untuk keluar): 10
Masukkan angka (0 untuk keluar): 10
Masukkan angka (0 untuk keluar): 0
totalnya adalah 30
```

5. Program doWhile1

```
1  package pekan6_2511531009;
2  import java.util.Scanner;
3  public class doWhile1_2511531009 {
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             }
12             while (!phrase.equals("abcd"));
13         console.close();
14     }
15
16 }
```

Disini program yang diinginkan adalah memasukkan *password*, jika *password* tidak sesuai dengan *key* yang berada pada *line* (12) maka nilai ini akan terus berulang.

Perbedaan penggunaan perulangan *do-while* dengan perulangan *while* ialah bahwa *do-while-loop* akan terbentuk seminimalnya satu kali. Perhatikan bahwa pada line (12) program tidak mencari nilai "abcd" melainkan nilai komplemen dari "abcd" (selain "abcd"). Artinya jika kita berhasil memasukkan *password* dalam sekali coba maka program akan berjalan seminimalnya satu kali, program yang terjadi disini ialah pencetakkan line "Input Passoword : "

```
Input Password : aaaa
Input Password : adasda
Input Password : dasdasd
Input Password : abcd
```

BAB III
KESIMPULAN

3.1 Kesimpulan

Penggunaan *syntax* perulangan *while* memiliki peran yang lebih fleksibel daripada penggunaan *for-loop* karena *conditional statement*nya tidak selalu bergantung pada iterasi yang digunakan pada program, melainkan hanya berdasarkan nilai kondisionlanya saja.

3.2 Saran

Berdasarkan praktikum yang telah dilakukan mengenai while-*loop syntax* dalam bahasa pemrograman *Java*. Penulis harap kita sebagai mahasiswa informatika harus dapat memahami dan mampu menerapkan konsep *while-loop* dalam membuat sebuah kode program.

DAFTAR PUSTAKA

Hsu, Jonathan, diperbaharui oleh Whitfield, Brennan. (2025). *How to Pick Between a For Loop and While Loop*. builtin.com. Diakses pada tanggal 11 November 2025, dari https://builtin.com/software-engineering-perspectives/for-loop-vs-while-loop

Dicoding Indonesia. (2025). *Apa Itu Looping? Kenali Jenis-Jenisnya*. builtin.com. Diakses pada tanggal 11 November 2025, dari https://www.dicoding.com/blog/apa-itu-looping-kenali-jenis-jenisnya/#:~:text=Kelebihan%20Looping%20While%20*%20Cocok%20untuk%20situasi,secara%20dinamis.%20*%20Lebih%20fleksibel%20dibanding%20for.