LAPORAN PRATIKUM

PEMROGRAMAN ALGORITMA PEMROGRAMAN

PERULANGAN WHILE DAN DO WHILE

Disusun Oleh:

Gina Ramadhani

Nim: 2511533014

Dosen Pengampu: DR. Wahyudi, S.T, M.T

Asisten Pratikum: Rahmad Dwi Rizki Olders

DAPERTEMEN INFORMATIKA

FAKULTAS TEKNOLOGI INFORMASI

UNIVERSITAS ANDALAS

KATA PENGANTAR

Laporan praktikum ini disusun sebagai bentuk tanggung jawab penulis atas pelaksanaan kegiatan praktikum pada mata kuliah Algoritma dan Pemrograman, khususnya mengenai materi Perulangan While dan Do While dalam Bahasa Pemrograman Java. Penyusunan laporan ini dimaksudkan untuk membantu penulis memperdalam pemahaman terhadap konsep-konsep dasar pemrograman, khususnya bagaimana perulangan dapat digunakan untuk mengulang proses tertentu secara efisien dan sesuai dengan kondisi yang diinginkan.

Selain sebagai dokumentasi resmi dari kegiatan praktikum, laporan ini juga berfungsi sebagai sarana pembelajaran tambahan. Dengan menulis laporan ini, penulis berkesempatan untuk melatih keterampilan menulis secara sistematis, mencatat hasil percobaan dengan rapi, dan menyajikan informasi teknis secara jelas. Hal ini penting tidak hanya untuk memenuhi persyaratan praktikum, tetapi juga untuk mengembangkan kemampuan analisis dan komunikasi ilmiah yang akan berguna dalam proses belajar selanjutnya.

Penulis menyadari bahwa laporan ini masih jauh dari sempurna. Masih mungkin terdapat kekurangan dalam penyusunan kata, penjelasan konsep, atau penyajian hasil percobaan. Oleh karena itu, penulis sangat mengharapkan kritik dan saran yang membangun dari pembaca maupun dosen pembimbing agar laporan berikutnya dapat lebih baik, jelas, dan mudah dipahami. Dengan demikian, laporan ini tidak hanya menjadi bukti pelaksanaan praktikum, tetapi juga sarana untuk meningkatkan kualitas belajar dan kemampuan akademik penulis secara menyeluruh.

Padang, 2025

Penulis

DAFTAR ISI

BAB I

PENDAHULUAN

1.1 Latar Belakang

Perulangan merupakan salah satu konsep dasar dalam pemrograman yang penting untuk dipahami, karena membantu programmer mengeksekusi perintah secara berulang tanpa harus menulis kode yang sama berulang-ulang. Dalam bahasa pemrograman Java, perulangan memungkinkan program berjalan lebih efisien, mengurangi kesalahan akibat duplikasi kode, dan mempermudah pengelolaan program yang kompleks.

Di Java, terdapat beberapa jenis perulangan, namun yang paling umum digunakan adalah while dan do-while**.** Perulangan while akan mengeksekusi suatu blok kode selama kondisi tertentu terpenuhi, dan pengecekan kondisi dilakukan sebelum blok kode dijalankan. Hal ini berarti jika kondisi awal tidak terpenuhi, blok kode mungkin tidak dijalankan sama sekali. Sementara itu, perulangan do-while akan mengeksekusi blok kode setidaknya satu kali karena pengecekan kondisi dilakukan setelah kode dijalankan. Jenis perulangan ini cocok digunakan ketika suatu proses harus dijalankan minimal satu kali sebelum dilakukan evaluasi kondisi.

Memahami kedua jenis perulangan ini sangat penting, karena perulangan digunakan secara luas dalam berbagai aplikasi, mulai dari pengolahan data, simulasi permainan, hingga program interaktif. Melalui praktikum ini, penulis diharapkan dapat memahami bagaimana cara kerja perulangan while dan do-while, serta mampu menerapkannya dalam pembuatan program sederhana menggunakan Java.

1.2 Tujuan Praktikum

1. Memahami konsep dasar perulangan dalam bahasa pemrograman Java.
2. Mampu membedakan cara kerja perulangan while dan do-while beserta penerapannya.

3. Melatih kemampuan membuat program sederhana yang menggunakan struktur perulangan.
4. Mengembangkan keterampilan analisis dan logika pemrograman melalui penerapan perulangan dalam kasus nyata, seperti simulasi permainan atau pengolahan data.
5. Meningkatkan keterampilan menulis laporan praktikum secara sistematis dan sesuai kaidah akademik.

1.3 Manfaat Praktikum

1. Memperkuat pemahaman konsep perulangan
   Praktikum ini membantu mahasiswa memahami cara kerja perulangan while dan do-while, sehingga mampu membedakan kegunaan dan penerapannya dalam berbagai situasi.
2. Melatih kemampuan pemrograman praktis
   Dengan langsung membuat program menggunakan perulangan, mahasiswa dapat mengasah keterampilan menulis kode, menganalisis logika, dan menerapkan teori ke dalam praktik.
3. Meningkatkan kemampuan berpikir logis
   Proses perencanaan program dan pengaturan alur perulangan mengajarkan mahasiswa berpikir sistematis dan logis dalam memecahkan masalah.
4. Meningkatkan keterampilan dokumentasi
   Praktikum mendorong mahasiswa untuk menulis laporan yang rapi dan terstruktur, sehingga mampu menyampaikan hasil praktikum dengan jelas dan sesuai kaidah akademik.
5. Persiapan untuk pemrograman yang lebih kompleks
   Pemahaman perulangan dasar menjadi fondasi penting untuk belajar konsep pemrograman lanjutan, seperti array, pengolahan data, dan pembuatan aplikasi interaktif.

BAB II

PEMBAHASAN

2.1 Praktikum “Sentine1Loop”

```java
package pekan6_2511533014;

import java.util.Scanner;

public class Sentine1Loop_2511533014 {
    public static void main(String[] args) {
        Scanner console = new Scanner (System.in);
        int sum = 0;
        int number=12;   // "dummy value", anything but 0

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

    }

}
```

Gambar 2.1 Kode program praktikum Sentine1Loop

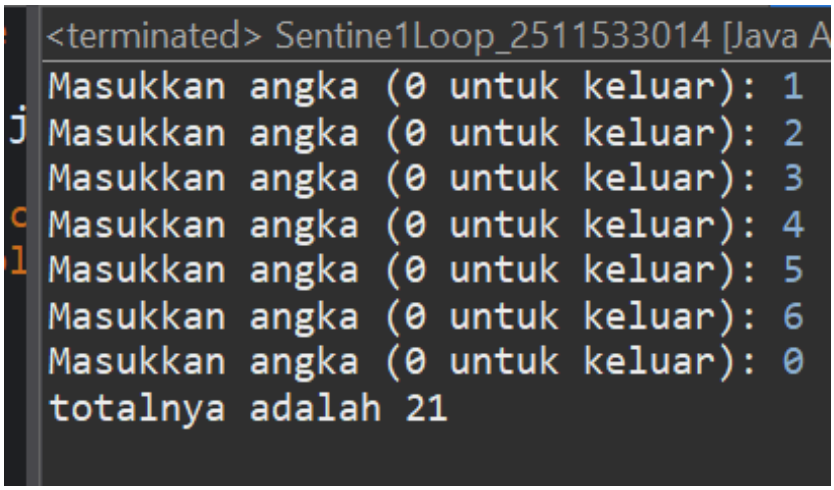
Program ini menggunakan loop sentinel (perulangan dengan nilai penghenti) untuk menjumlahkan serangkaian angka yang dimasukkan oleh pengguna. Perulangan akan berhenti embal pengguna memasukkan angka 0, dan kemudian program akan menampilkan total dari semua angka yang telah dijumlahkan.

Berikut penjelsan langkah-langkah kerjanya:

1. Program dimulai dan menjalankan fungsi utama (main).
2. Program menyiapkan alat untuk membaca input dari pengguna melalui keyboard.
3. Dua embali dibuat: satu untuk menyimpan total penjumlahan, dan satu lagi untuk menyimpan angka yang dimasukkan pengguna. Variabel angka diberi nilai awal selain nol agar perulangan bisa dimulai.
4. Program masuk ke perulangan yang akan terus berjalan selama angka yang dimasukkan bukan nol.

5. Program menampilkan pesan agar pengguna memasukkan angka, dengan pemberitahuan bahwa angka 0 digunakan untuk keluar.
6. Pengguna mengetikkan sebuah angka, lalu program menyimpannya.
7. Angka yang dimasukkan ditambahkan ke total penjumlahan.
8. Program memeriksa embali kondisi perulangan:
    - Jika angka yang baru dimasukkan bukan nol, proses embali ke langkah 5.
    - Jika angka nol, perulangan berhenti.
9. Setelah keluar dari perulangan, program menampilkan hasil total penjumlahan dari semua angka yang sudah dimasukkan sebelumnya.
10. Program selesai dijalankan.

Dari Langkah-langkah diatas kita akan mendapatkan output:



Gambar 2.2 Output kode program Sentine1Loop

2.1.1 Analisis

- Gambar menampilkan hasil output berupa beberapa kali permintaan input angka dari pengguna dengan pesan “Masukkan angka (0 untuk keluar)”.
- Pengguna memasukkan angka secara berurutan yaitu 1, 2, 3, 4, 5, 6, dan terakhir 0 sebagai tanda untuk menghentikan proses.
- Angka 0 berfungsi sebagai penanda berhenti atau sentinel yang digunakan agar program tidak lagi meminta input selanjutnya.

- Selama angka yang dimasukkan bukan 0, program terus menambahkan setiap angka ke dalam jumlah total.
- Setelah pengguna memasukkan angka 0, program menampilkan hasil akhir berupa pesan “totalnya adalah 21”.
- Nilai total 21 merupakan hasil penjumlahan dari semua angka yang dimasukkan sebelum 0 yaitu 1 + 2 + 3 + 4 + 5 + 6.
- Hasil tersebut menunjukkan bahwa program berjalan dengan benar tanpa error dan berhasil menghitung total dari data yang diinput pengguna.
- Program ini merupakan contoh penggunaan perulangan dengan kondisi berhenti untuk melakukan penjumlahan sejumlah nilai yang tidak ditentukan sebelumnya.

2.2 Praktikum “perulanganWhile1”

```java
package pekan6_2511533014;

import java.util.Scanner;

public class perulanganWhile1_2511533014 {
    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.print("Apakah lanjut (ya / tidak?)");
            jawab= scan.nextLine();
            //cek jawab = tidak, perulangan berhenti
            if (jawab.equalsIgnoreCase("tidak")) {
                running= false;
            }
        }
        System.out.println("Anda sudah melakukan perulangan sebanyak "+counter+" kali");
    }

}

```

Gambar 2.3 Kode program praktikum perulanganWhile1

Program tersebut merupakan program Java yang menggunakan perulangan *while* untuk menghitung jumlah pengulangan berdasarkan input pengguna. Setiap

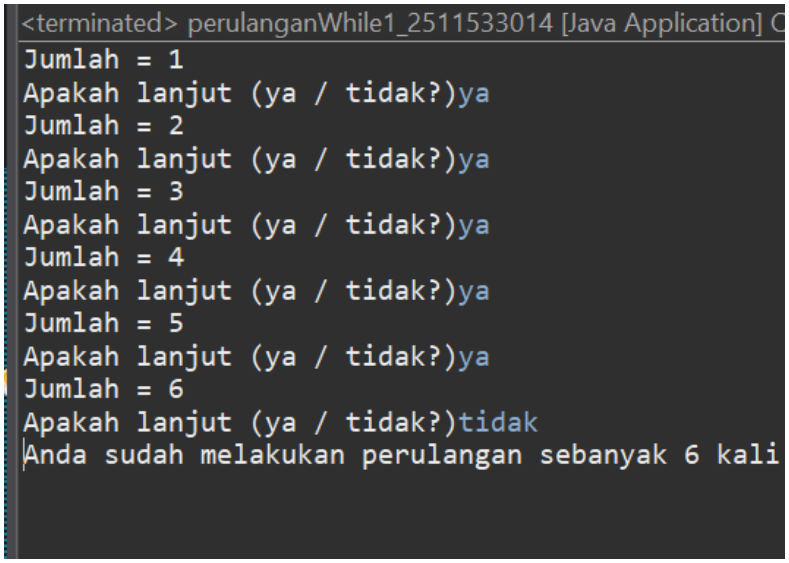
kali perulangan berjalan, nilai *counter* bertambah satu dan pengguna diminta menjawab apakah ingin melanjutkan dengan mengetik “ya” atau berhenti dengan mengetik “tidak”. Jika pengguna menjawab “tidak”, maka perulangan berhenti dan program menampilkan total jumlah perulangan yang telah dilakukan. Program ini menunjukkan penggunaan perulangan *while* dengan kondisi berhenti yang ditentukan oleh input pengguna.

Berikut penjelasan langkah-langkah kerja program berdasarkan urutan barisnya:

1. program diawali dengan mendefinisikan paket tempat file program disimpan.
2. program mengimpor pustaka scanner dari java.util agar bisa membaca input dari keyboard.
3. kelas utama program dibuat dengan nama sesuai file, dan di dalamnya terdapat metode utama main sebagai titik awal eksekusi.
4. variabel counter dideklarasikan dan diinisialisasi dengan nilai 0 untuk menghitung jumlah perulangan.
5. variabel jawab dideklarasikan untuk menyimpan jawaban pengguna.
6. variabel running dideklarasikan dengan nilai awal true agar perulangan dapat dijalankan.
7. objek scanner dibuat untuk menerima input dari pengguna melalui keyboard.
8. program masuk ke dalam perulangan while yang akan terus berjalan selama nilai running bernilai true.
9. setiap kali perulangan berjalan, nilai counter bertambah satu.
10. program menampilkan pesan jumlah perulangan ke layar.
11. program kemudian menanyakan kepada pengguna apakah ingin melanjutkan perulangan atau tidak.
12. pengguna memasukkan jawaban melalui keyboard, yang kemudian disimpan dalam variabel jawab.
13. program memeriksa isi variabel jawab; jika pengguna menjawab “tidak”, maka nilai running diubah menjadi false sehingga perulangan berhenti.

14. jika pengguna menjawab selain “tidak”, perulangan akan dilanjutkan dan kembali ke langkah penambahan counter.
15. setelah pengguna menjawab “tidak” dan perulangan berhenti, program menampilkan pesan akhir berisi jumlah total perulangan yang telah dilakukan.
16. program kemudian selesai dijalankan.

Dari langkah-langkah diatas kita akan mendapatkan output:



Gambar 2.4 Output kode programm perulanganWhile1

2.1.2 Analisis

• Program berjalan dengan baik tanpa error dan menampilkan output sesuai logika perulangan while.
• Setiap kali perulangan dijalankan, nilai counter bertambah satu dan ditampilkan pada layar.
• Pengguna diminta menjawab pertanyaan “apakah lanjut (ya / tidak?)” untuk menentukan apakah perulangan akan dilanjutkan atau dihentikan.
• Jika pengguna menjawab “ya”, maka perulangan terus berjalan dan nilai counter bertambah.
• Jika pengguna menjawab “tidak”, perulangan berhenti dan program menampilkan jumlah total perulangan yang telah dilakukan.

• Output menunjukkan bahwa pengguna menjawab “ya” sebanyak lima kali dan “tidak” pada perulangan keenam, sehingga total perulangan yang dilakukan adalah enam kali.

• Program bekerja sesuai logika dan menunjukkan penerapan perulangan while dengan kondisi berhenti berdasarkan input pengguna.

2.3 Praktikum “Lempardadu”

```java
package pekan6_2511533014;
import java.util.Random;
public class Lempardadu_2511533014 {
    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!");


    }

}
```

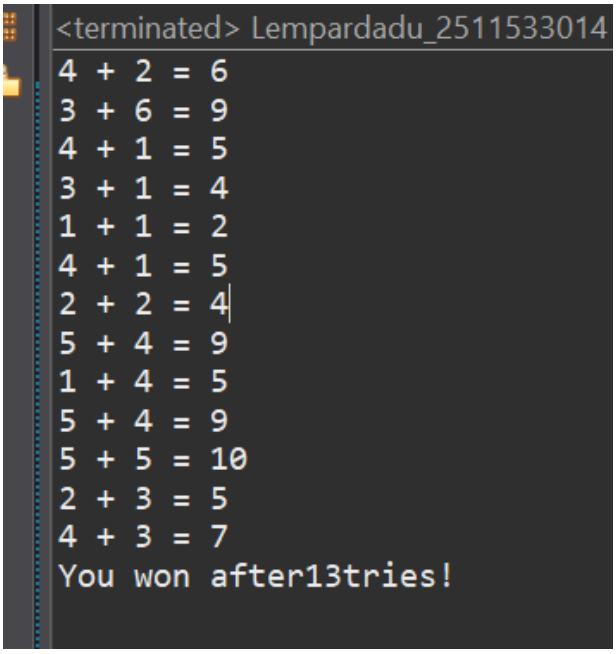
Gambar 2.5 Kode program praktikum Lempardadu

Program tersebut merupakan simulasi pelemparan dua dadu menggunakan bahasa java. program menggunakan kelas random untuk menghasilkan angka acak antara satu hingga enam, lalu menjumlahkan kedua hasil dadu. perulangan while terus berjalan sampai jumlah kedua dadu sama dengan tujuh. setiap kali melempar, program menampilkan hasil dadu dan menghitung jumlah percobaan. ketika jumlah tujuh muncul, perulangan berhenti dan program menampilkan jumlah percobaan yang dibutuhkan untuk menang.

Berikut langkah-langkah kerjanya:

1. program dimulai dengan membuat objek random yang berfungsi menghasilkan angka acak untuk dua buah dadu.
2. dua variabel, yaitu tries dan sum, diinisialisasi dengan nilai awal nol; tries digunakan untuk menghitung jumlah percobaan dan sum untuk menyimpan hasil penjumlahan dua dadu.
3. program masuk ke dalam perulangan while dengan kondisi sum tidak sama dengan tujuh, artinya program akan terus mengulang sampai jumlah kedua dadu bernilai tujuh.
4. di dalam perulangan, program menghasilkan dua angka acak antara satu hingga enam untuk mewakili nilai dadu pertama dan dadu kedua.
5. kedua nilai tersebut dijumlahkan dan hasilnya disimpan ke dalam variabel sum.
6. hasil dari kedua dadu serta jumlahnya ditampilkan ke layar agar pengguna dapat melihat setiap hasil pelemparan.
7. setiap kali perulangan terjadi, nilai tries bertambah satu untuk mencatat berapa kali dadu dilempar.
8. jika jumlah kedua dadu sama dengan tujuh, kondisi while menjadi salah sehingga perulangan berhenti.
9. setelah keluar dari perulangan, program menampilkan pesan kemenangan yang berisi jumlah percobaan yang dibutuhkan untuk mendapatkan hasil tujuh.
10. program berakhir setelah menampilkan hasil akhir tersebut.

Dari langkah-langkah keja kode program di atas kita mendapatkan output seperti ini:



Gambar 2.6 Output kode program Lempardadu

2.1.3 Analisis

• Program berjalan dengan baik tanpa error dan menghasilkan output sesuai logika permainan dadu.

• Setiap kali perulangan dijalankan, dua angka acak antara 1 hingga 6 dihasilkan sebagai hasil lemparan dua dadu.

• Hasil dari kedua dadu dijumlahkan dan ditampilkan ke layar bersama nilai masing-masing dadu.

• Proses pelemparan terus dilakukan hingga jumlah kedua dadu bernilai 7.

• Variabel percobaan digunakan untuk menghitung berapa kali lemparan dilakukan sebelum mendapatkan hasil 7.

• Setelah kondisi tercapai, program menampilkan pesan bahwa pemain menang beserta jumlah total percobaan yang dibutuhkan.

2.4 Praktikum “GamePenjumlahan”

```java
package pekan6_2511533014;
import java.util.Scanner;
public class GamePenjumlahan_2511533014 {
    public static void main(String[] args) {
        Scanner console = new Scanner (System.in);
        Random rand = new Random ();
        // play until user gets 3 wrong
        int points = 0;
        int wrong = 0;
        while (wrong < 3) {
            int result = play(console, rand);  // play one game
            if (result > 0) {
                points++;
            } else {
                wrong++;
            }
        }
        System.out.println("You earned " + points + "total points.");
    }
    // membuat soal penjumlahan dan ditampilkan ke user
    public static int play (Scanner console, Random rand) {
        //print the operands being added, and sum them
        int operands = rand.nextInt (4) + 2;
        int sum = rand.nextInt (10) + 1;
        System.out.println(false);

    for (int i = 2; i <= operands; i++) {
        int n = rand.nextInt(10) + 1;
        sum += n;
        System.out.print("+ " + n);
    }
    System.out.print(" = ");
```

```java
    // read user's guess and report whether it was correct
    int guess = console.nextInt();
    if (guess == sum) {
        return 1;
    } else {
        System.out.println("Wrong! The answer was " + sum);
        return 0;
    }
    }
}
```

Gambar 2.7 Kode program praktikum GamePenjumlahan

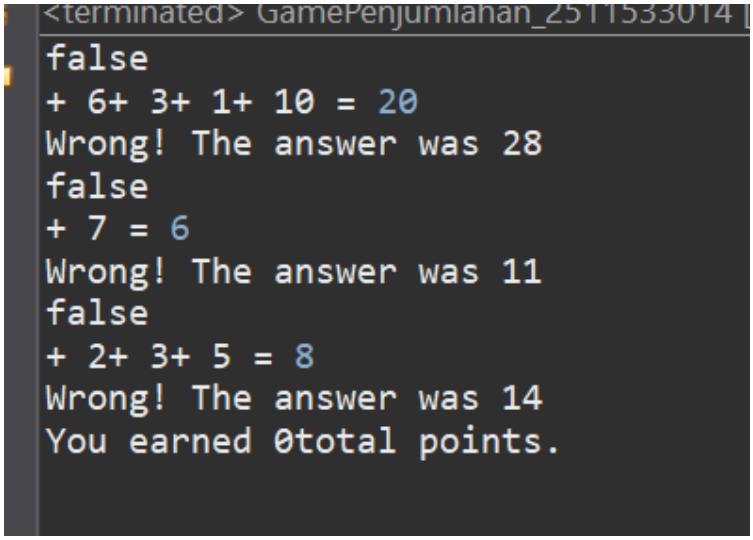
Program GamePenjumlahan merupakan permainan sederhana berbasis teks yang melatih kemampuan penjumlahan pengguna. Program menggunakan kelas Scanner untuk membaca input dan Random untuk menghasilkan angka acak. Dalam permainan, pengguna akan diberikan soal penjumlahan acak yang terdiri dari dua hingga lima bilangan. Jika jawaban benar, poin bertambah, sedangkan jika salah, jumlah kesalahan meningkat. Permainan berakhir setelah pengguna salah tiga kali,

kemudian program menampilkan total poin yang diperoleh. Program ini bertujuan untuk memberikan latihan berhitung secara interaktif dan menyenangkan.

Berikut penjelasan langkah-langkah kerjanya:

1. Program dimulai dan mengimpor dua pustaka yaitu Scanner untuk membaca input pengguna dan Random untuk menghasilkan angka acak.
2. Di dalam metode main dibuat objek Scanner dan Random.
3. Dua variabel disiapkan yaitu points untuk menyimpan jumlah jawaban benar dan wrong untuk menghitung jumlah kesalahan.
4. Program menjalankan perulangan while selama nilai wrong kurang dari 3.
5. Di dalam perulangan metode play dipanggil untuk menampilkan satu soal penjumlahan acak.
6. Metode play menentukan banyaknya bilangan yang dijumlahkan antara 2 hingga 5 angka dan menghasilkan angka acak dari 1 sampai 10.
7. Soal penjumlahan ditampilkan di layar misalnya 4 + 7 + 2 =.
8. Pengguna memasukkan jawaban melalui keyboard.
9. Program membandingkan jawaban pengguna dengan hasil penjumlahan yang benar.
10. Jika jawaban benar metode play mengembalikan nilai 1 dan points bertambah satu.
11. Jika jawaban salah program menampilkan jawaban yang benar mengembalikan nilai 0 dan menambah jumlah kesalahan wrong.
12. Perulangan terus berlanjut sampai pengguna salah sebanyak tiga kali.
13. Setelah perulangan berhenti program menampilkan total points yang diperoleh pengguna.
14. Program selesai dijalankan.

Dari langkah-langkah keja kode program di atas kita mendapatkan output:



Gambar 2.8 Output kode program GamePenjumlahan

2.1.4 Analisis

- Program berjalan dengan baik tanpa error dan menghasilkan output sesuai logika permainan penjumlahan.
- Pada setiap perulangan program menampilkan beberapa bilangan acak yang harus dijumlahkan oleh pengguna.
- Pengguna kemudian memasukkan hasil penjumlahan sebagai jawabannya.
- Program membandingkan jawaban pengguna dengan hasil penjumlahan yang benar.
- Jika jawaban pengguna salah program menampilkan pesan Wrong The answer was beserta hasil penjumlahan yang benar.
- Nilai false yang muncul di awal setiap soal berasal dari perintah System.out.printlnfalse yang ada di dalam metode play dan sebenarnya tidak diperlukan dalam tampilan program.
- Permainan terus berlangsung sampai pengguna salah menjawab sebanyak tiga kali.
- Setelah tiga kali salah perulangan berhenti dan program menampilkan pesan akhir You earned 0 total points karena tidak ada jawaban yang benar selama permainan.

- Secara keseluruhan logika permainan berjalan sesuai dengan rancangan yaitu memberikan soal penjumlahan acak mengevaluasi jawaban menghitung poin dan mengakhiri permainan setelah tiga kesalahan.

2.5 Praktikum “doWhile1”

```
1  package pekan6_2511533014;
2
3  import java.util.Scanner;
4
5  public class doWhile1_2511533014 {
6
7      public static void main(String[] args) {
8          Scanner console = new Scanner (System.in);
9              String phrase;
10             do {
11                 System.out.print("input Password: ");
12                 phrase = console.next();
13             } while (!phrase.equals("abcd"));
14
15     }
16
17 }
18
```

Gambar 2.9 Kode program praktikum doWhile1

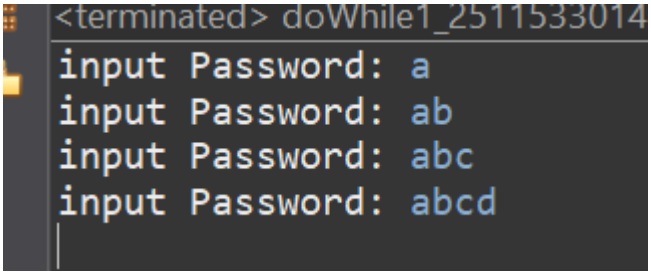
Program ini menggunakan struktur perulangan *do-while* untuk meminta pengguna memasukkan sebuah kata sandi (password). Program akan terus meminta input dari pengguna sampai pengguna mengetikkan password yang benar, yaitu “abcd”. Jika pengguna memasukkan password yang salah, program akan mengulang dan kembali meminta input. Setelah password yang dimasukkan cocok dengan kata “abcd”, perulangan berhenti dan program selesai dijalankan.

Berikut penjelasan langkah-langkah kerjanya:

1. Program dimulai dan menjalankan fungsi utama (main).
2. Program membuat objek *Scanner* untuk membaca input dari pengguna melalui keyboard.
3. Sebuah variabel bertipe *String* dibuat untuk menyimpan input password dari pengguna.
4. Program masuk ke dalam perulangan *do-while*.

5. Di dalam perulangan, program menampilkan pesan “input Password:” agar pengguna mengetikkan password.
6. Pengguna memasukkan sebuah kata, dan kata tersebut disimpan ke dalam variabel *phrase*.
7. Program memeriksa apakah nilai dari *phrase* sama dengan “abcd”.
8. Jika password yang dimasukkan tidak sama dengan “abcd”, maka perulangan akan berjalan lagi dan pengguna diminta memasukkan password kembali.
9. Jika password yang dimasukkan sama dengan “abcd”, maka kondisi perulangan menjadi salah dan program berhenti.
10. Program selesai dijalankan setelah pengguna berhasil memasukkan password yang benar.

Dari langkah-langkah kerja diatas kita mendapatkan output:



Gambar 2.10 Output kode program doWhile1

2.1.5 Analisis

• Program berjalan dengan baik tanpa error dan menghasilkan output sesuai logika pemeriksaan password.

• Setiap kali perulangan dijalankan, program menampilkan pesan *“input Password:”* dan menunggu pengguna mengetikkan input.

• Pengguna beberapa kali memasukkan password yang salah, yaitu “a”, “ab”, dan “abc”, sehingga program terus mengulangi proses input.

• Program menggunakan perulangan *do-while*, sehingga minimal satu kali input akan selalu diminta sebelum pengecekan kondisi dilakukan.

• Setiap kali input tidak sama dengan kata sandi yang benar (“abcd”), program

kembali menampilkan pesan untuk memasukkan password lagi.

• Setelah pengguna memasukkan password yang benar, yaitu “abcd”, kondisi perulangan tidak terpenuhi dan program berhenti.

• Program berakhir setelah password yang dimasukkan cocok dengan nilai yang ditentukan (“abcd”).

BAB III

PENUTUP

3.1 Kesimpulan

Dari hasil praktikum tentang perulangan *while* dan *do-while* dapat disimpulkan bahwa kedua jenis perulangan tersebut memiliki fungsi utama untuk menjalankan serangkaian perintah secara berulang berdasarkan kondisi tertentu. Perbedaan utamanya terletak pada posisi pengecekan kondisi. Pada perulangan *while*, kondisi diperiksa terlebih dahulu sebelum perulangan dijalankan, sehingga jika kondisi awal tidak terpenuhi, perulangan tidak akan dijalankan sama sekali. Sedangkan pada perulangan *do-while*, proses perulangan akan dijalankan minimal satu kali karena pengecekan kondisi dilakukan setelah blok perulangan dijalankan.

Melalui percobaan yang dilakukan, dapat dipahami bahwa perulangan *while* cocok digunakan ketika jumlah pengulangan belum pasti tetapi kondisi awal harus diuji terlebih dahulu, sedangkan *do-while* digunakan ketika setidaknya satu kali eksekusi diperlukan sebelum kondisi diperiksa. Dengan demikian, pemilihan jenis perulangan harus disesuaikan dengan kebutuhan logika program yang akan dibuat.

3.2 Saran

Agar praktikum berikutnya berjalan lebih efektif, disarankan untuk memperhatikan pemilihan jenis perulangan yang sesuai dengan kebutuhan program. Pemahaman yang jelas tentang perbedaan *while* dan *do-while* akan membantu dalam menulis kode yang lebih efisien dan mudah dipahami.

Selain itu, latihan membuat berbagai contoh program menggunakan kedua jenis perulangan akan memperkuat pemahaman konsep dan penerapannya dalam situasi yang berbeda. Terakhir, sebaiknya selalu melakukan pengecekan dan pengujian program secara menyeluruh agar kesalahan logika maupun input dapat terdeteksi, sehingga program berjalan dengan baik sesuai yang diharapkan.

DAFTAR PUSTAKA

[1] H. Schildt, *Java The Complete Reference*, 11th ed. New York: McGraw-Hill Education, 2019.

[2] D. Liang, *Introduction to Java Programming and Data Structures*, 12th ed. Boston: Pearson, 2020.

[3] W. Savitch, *Absolute Java*, 6th ed. Boston: Pearson, 2016.

[4] Oracle, "The Java Tutorials," 2023. [Daring]. Tersedia pada: https://docs.oracle.com/javase/tutorial/. [Diakses: 12-Sep-2025].

[5] GeeksforGeeks, "Java util Random class in Java," 2024. [Daring]. Tersedia pada: https://www.geeksforgeeks.org/java/java-util-random-class-java/. [Diakses: 12-Sep-2025].

[6] W3Schools, "Java User Input Scanner," 2024. [Daring]. Tersedia pada: https://www.w3schools.com/java/java_user_input.asp. [Diakses: 12-Sep-2025].

[7] Oracle, "Java Platform Standard Edition Documentation," 2024. [Daring]. Tersedia pada: https://docs.oracle.com/javase/8/docs/api/. [Diakses: 12-Sep-2025].