**LAPORAN PRAKTIKUM 6**

**Algoritma Pemrograman**



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

Nama: Muhammad Rayhan Anugrah  
NIM: 2311533019

**FAKULTAS TEKNOLOGI INFORMASI**

**DEPARTEMEN INFORMATIKA**

**UNIVERSITAS ANDALAS**

**2024**

1. **Pendahuluan**
2. Tujuan Praktikum

 Memahami konsep dasar perulangan menggunakan while dan do-while dalam pemrograman.

 Menerapkan perulangan while untuk menjalankan blok kode selama kondisi tertentu bernilai true.

 Mengimplementasikan perulangan do-while untuk memastikan blok kode dijalankan minimal satu kali meskipun kondisi awal tidak terpenuhi.

 Menguji perbedaan antara perulangan while dan do-while dalam hal cara pemeriksaan kondisi dan eksekusi blok kode.

 Mengembangkan pemahaman tentang penggunaan perulangan untuk memecahkan masalah interaktif atau dinamis, seperti permainan atau aplikasi berbasis input pengguna.

1. Kajian Teori

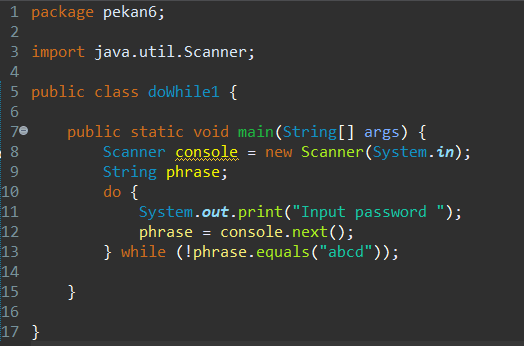
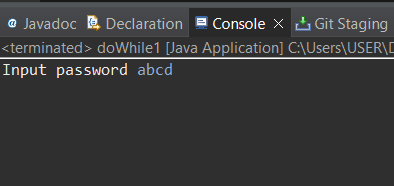
Perulangan (looping) merupakan salah satu konsep dasar dalam pemrograman yang memungkinkan eksekusi instruksi secara berulang-ulang dengan kondisi tertentu. Dua jenis perulangan yang sering digunakan dalam banyak bahasa pemrograman, termasuk Java, adalah perulangan while dan do-while. Keduanya memiliki prinsip dasar yang sama, yakni mengeksekusi blok kode berulang kali selama kondisi tertentu terpenuhi. Namun, terdapat perbedaan mendasar dalam cara kedua perulangan ini memeriksa kondisi dan menentukan kapan perulangan akan berhenti.

Perulangan while bekerja dengan memeriksa kondisi terlebih dahulu sebelum mengeksekusi blok kode. Artinya, perulangan ini hanya akan dilanjutkan jika kondisi yang diberikan bernilai true. Jika kondisi tersebut bernilai false pada awal eksekusi, maka blok kode di dalam perulangan tidak akan dijalankan sama sekali. Hal ini menjadikan perulangan while cocok digunakan dalam situasi di mana kita tidak tahu berapa kali perulangan akan terjadi, namun kita ingin terus melakukan proses tertentu selama kondisi tertentu tetap terpenuhi. Contoh umum penggunaan while adalah ketika program menunggu input atau kondisi yang berubah selama program berjalan.

Sementara itu, perulangan do-while sedikit berbeda karena memeriksa kondisi setelah blok kode dijalankan. Ini berarti bahwa blok kode di dalam perulangan do-while akan selalu dijalankan minimal sekali, meskipun kondisi yang diberikan awalnya tidak memenuhi syarat. Perulangan ini berguna dalam situasi di mana kita ingin memastikan bahwa setidaknya satu kali eksekusi dilakukan, meskipun kondisi penghentian baru diperiksa setelah eksekusi pertama. Penggunaan yang tepat dari perulangan do-while dapat ditemukan dalam aplikasi yang membutuhkan interaksi dengan pengguna, di mana program harus menampilkan informasi atau meminta input terlebih dahulu sebelum memutuskan apakah akan melanjutkan atau berhenti.

Kedua jenis perulangan ini sangat berguna dalam pengembangan perangkat lunak, khususnya dalam pembuatan program yang melibatkan proses yang berulang, seperti pengolahan data, permainan, atau interaksi dengan pengguna. Pemahaman yang baik mengenai cara kerja dan perbedaan antara while dan do-while memungkinkan programmer untuk memilih jenis perulangan yang paling sesuai dengan kebutuhan program.

1. **Pembahasan**
2. Class DoWhile1

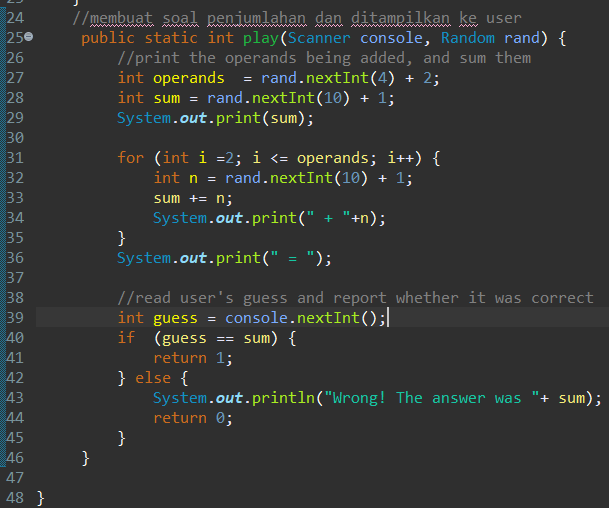
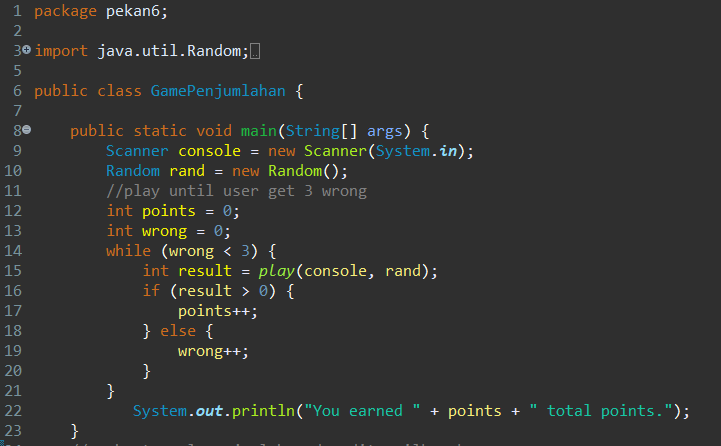


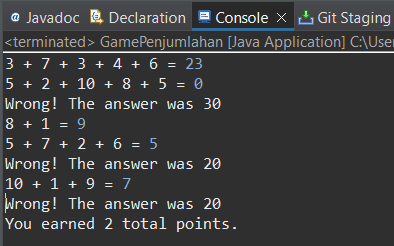
Program ini merupakan implementasi dari struktur kontrol perulangan do-while yang berada dalam package pekan6. Program dirancang sebagai sistem autentikasi sederhana yang meminta pengguna untuk memasukkan password. Pada awal program, sebuah objek Scanner dibuat untuk menerima input dari pengguna melalui console dengan nama variabel 'console'. Program menggunakan variabel String bernama 'phrase' untuk menyimpan input password yang dimasukkan oleh pengguna.

Dalam struktur do-while, program akan pertama kali menampilkan pesan "Input password " kepada pengguna menggunakan System.out.print. Setelah itu, program akan mengambil input dari pengguna menggunakan method next() dari objek Scanner dan menyimpannya dalam variabel phrase. Perulangan akan terus berlanjut selama password yang dimasukkan tidak sama dengan "abcd" (diimplementasikan menggunakan !phrase.equals("abcd")). Penggunaan operator negasi (!) pada kondisi while menunjukkan bahwa perulangan akan berlanjut selama string yang dimasukkan tidak sama dengan "abcd".

Program ini memanfaatkan keunikan struktur do-while dimana blok kode akan dieksekusi minimal satu kali sebelum pengecekan kondisi while dilakukan. Hal ini membuat program akan selalu meminta input password setidaknya satu kali. Program akan berhenti ketika pengguna berhasil memasukkan password yang benar yaitu "abcd". Implementasi ini mendemonstrasikan penggunaan praktis dari perulangan do-while dalam membuat sistem validasi input sederhana.

1. Class GamePenjumlahan

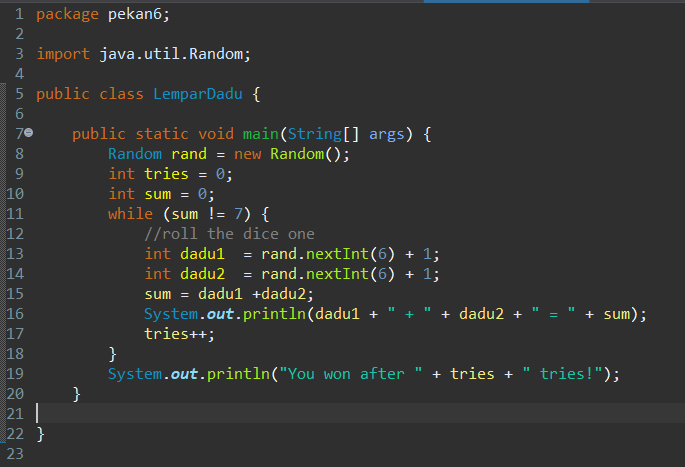
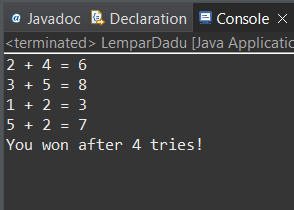




Program ini adalah sebuah game penjumlahan sederhana yang diimplementasikan dalam package pekan6. Program dirancang untuk menguji kemampuan pengguna dalam melakukan operasi penjumlahan dengan menggunakan angka acak. Di awal program, dideklarasikan dua objek utama yaitu Scanner untuk menerima input dari pengguna dan Random untuk menghasilkan angka acak. Program menggunakan dua variabel counter penting yaitu 'points' untuk menghitung jumlah jawaban benar dan 'wrong' untuk menghitung jumlah jawaban salah.

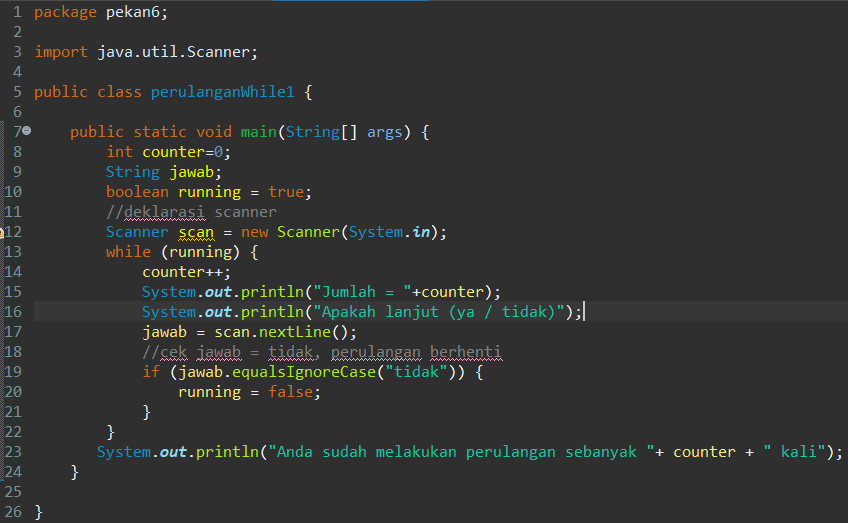
Struktur utama program menggunakan perulangan while yang akan terus berjalan selama jumlah jawaban salah (wrong) kurang dari 3. Dalam setiap iterasi, program memanggil method play() yang mengembalikan nilai hasil permainan. Jika method play() mengembalikan nilai positif, artinya jawaban benar dan points bertambah. Sebaliknya, jika mengembalikan nilai 0, berarti jawaban salah dan counter wrong bertambah. Permainan akan berakhir ketika pengguna telah salah menjawab sebanyak 3 kali, dan program akan menampilkan total points yang berhasil dikumpulkan.

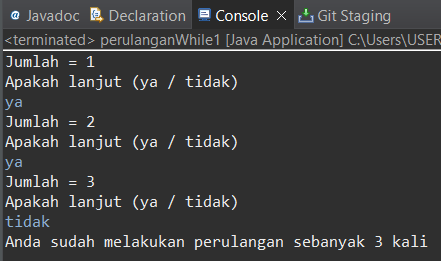
Method play() berfungsi sebagai inti dari permainan yang menghasilkan soal penjumlahan secara acak. Method ini menggunakan dua parameter yaitu objek Scanner dan Random. Pertama, method menentukan jumlah operand secara acak antara 2 sampai 5 (rand.nextInt(4) + 2). Kemudian, program menghasilkan angka pertama antara 1 sampai 10 dan menyimpannya sebagai nilai awal sum. Selanjutnya, menggunakan perulangan for, program menambahkan operand-operand berikutnya yang juga dihasilkan secara acak antara 1 sampai 10. Setiap operand ditampilkan dengan format yang sesuai (dipisahkan dengan tanda +). Setelah semua operand ditampilkan, program meminta input jawaban dari pengguna. Jika jawaban sesuai dengan nilai sum, method mengembalikan nilai 1 (benar). Jika salah, program menampilkan jawaban yang benar dan mengembalikan nilai 0.

1. Class LemparDadu

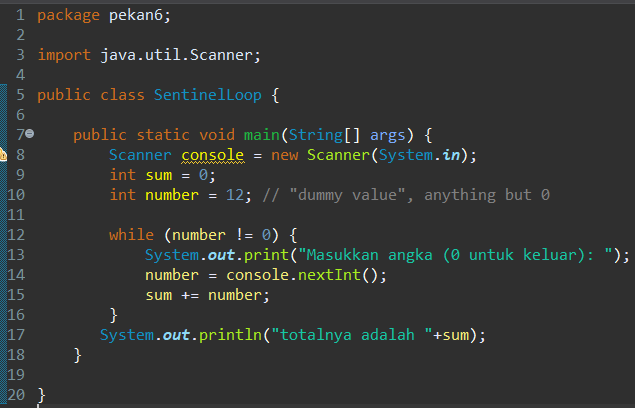
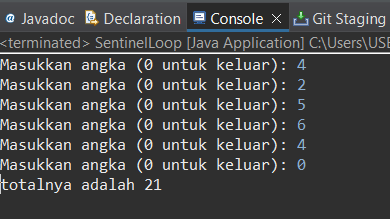
Kode program di atas merupakan implementasi sederhana dari permainan lempar dadu yang bertujuan untuk mendapatkan jumlah angka dadu sebesar tujuh. Program ini dimulai dengan mengimpor kelas Random dari pustaka Java, yang digunakan untuk menghasilkan angka acak. Di dalam metode main, objek Random dibuat untuk menghasilkan angka acak yang merepresentasikan hasil lemparan dua dadu. Program kemudian menginisialisasi dua variabel, yaitu tries untuk menghitung jumlah percobaan lemparan dan sum untuk menyimpan hasil penjumlahan dari kedua dadu.

Selama jumlah dadu tidak sama dengan tujuh, program akan terus melakukan lemparan dadu. Setiap kali dadu dilempar, dua angka acak dihasilkan, masing-masing untuk dadu pertama dan kedua, dengan rentang dari 1 hingga 6. Hasil dari kedua dadu tersebut dijumlahkan dan ditampilkan ke layar. Setelah mendapatkan jumlah tujuh, program akan keluar dari loop dan menampilkan pesan yang menunjukkan berapa kali percobaan dilakukan sebelum mencapai hasil yang diinginkan. Program ini mencerminkan penggunaan kontrol alur dengan loop while dan pemanfaatan kelas Random untuk menghasilkan angka acak, serta memberikan gambaran sederhana tentang bagaimana simulasi permainan dapat diterapkan dalam pemrograman Java.

1. Class PerulanganWhile1



Kode program di atas merupakan contoh penggunaan perulangan dengan struktur kontrol while dalam bahasa pemrograman Java. Program ini dimulai dengan mendeklarasikan variabel counter yang berfungsi untuk menghitung jumlah iterasi perulangan, serta variabel jawab untuk menyimpan input dari pengguna. Selain itu, terdapat variabel boolean running yang diinisialisasi dengan nilai true, menandakan bahwa perulangan akan terus berlangsung. Program menggunakan kelas Scanner untuk mengambil input dari pengguna melalui konsol. Di dalam loop while, program akan mencetak nilai counter yang menunjukkan jumlah iterasi yang telah dilakukan. Selanjutnya, program menanyakan kepada pengguna apakah mereka ingin melanjutkan perulangan dengan memberikan pilihan "ya" atau "tidak". Jika pengguna menjawab "tidak" (dengan memanfaatkan metode equalsIgnoreCase untuk mengabaikan perbedaan huruf besar dan kecil), maka variabel running diubah menjadi false, yang menyebabkan loop berhenti. Setelah perulangan selesai, program akan mencetak jumlah total iterasi yang telah dilakukan. Kode ini menggambarkan bagaimana perulangan dapat digunakan untuk mengulang suatu proses berdasarkan kondisi yang ditentukan oleh input pengguna, serta menunjukkan interaksi dasar antara program dan pengguna.

1. Class SentinelLoop

Program ini dirancang untuk menghitung total dari serangkaian angka yang dimasukkan oleh pengguna, dengan menggunakan nilai sentinel (dalam hal ini, angka 0) sebagai sinyal untuk menghentikan proses input. Program dimulai dengan mendeklarasikan objek Scanner untuk membaca input dari konsol. Variabel sum diinisialisasi dengan nilai 0 untuk menyimpan total jumlah angka yang dimasukkan, sementara variabel number diatur dengan nilai 12 sebagai "dummy value" yang tidak sama dengan 0, sehingga loop dapat dimulai.

Di dalam loop while, program akan meminta pengguna untuk memasukkan angka, dengan instruksi bahwa memasukkan angka 0 akan mengakhiri proses input. Setiap angka yang dimasukkan oleh pengguna akan ditambahkan ke variabel sum. Loop akan terus berlanjut hingga pengguna memasukkan angka 0. Setelah loop selesai, program akan mencetak total jumlah angka yang telah dimasukkan. Kode ini menunjukkan bagaimana sentinel loop dapat digunakan untuk mengelola input yang tidak terbatas dengan cara yang sederhana dan efisien, serta memberikan pengalaman interaktif kepada pengguna.

1. **Kesimpulan**

Pada praktikum ini, kami telah mempelajari dan mengimplementasikan beberapa konsep perulangan dalam bahasa pemrograman, seperti while, do-while, serta penerapan berbagai jenis game berbasis perulangan, seperti permainan lempar dadu, sentinelloop, dan game penjumlahan.

1. **Perulangan while**: Perulangan ini digunakan untuk menjalankan blok kode selama kondisi tertentu bernilai true. Dalam permainan lempar dadu dan game penjumlahan, perulangan while memungkinkan kita untuk terus meminta input dari pengguna hingga kondisi tertentu tercapai (misalnya, tebakan yang benar atau hasil penjumlahan yang benar). Perulangan ini sangat berguna ketika kita tidak mengetahui berapa kali iterasi yang dibutuhkan sebelumnya, karena kondisi berhenti ditentukan oleh input atau hasil dalam program.
2. **Perulangan do-while**: Berbeda dengan while, perulangan do-while memastikan bahwa blok kode setidaknya dijalankan satu kali, meskipun kondisi awalnya tidak memenuhi syarat. Perulangan ini cocok digunakan dalam situasi di mana kita ingin setidaknya melakukan satu iterasi sebelum memeriksa kondisi.
3. **Permainan Lempar Dadu**: Dalam permainan ini, kita menggunakan perulangan untuk terus meminta pemain menebak hasil lemparan dadu hingga tebakan yang benar ditemukan. Perulangan ini berfungsi untuk mengulang proses hingga pemain mencapai hasil yang benar, yang memungkinkan pengalaman interaktif yang menarik.
4. **Sentinelloop**: Teknik sentinelloop digunakan untuk menghentikan perulangan berdasarkan input khusus dari pengguna, seperti kata kunci "stop" atau "quit". Dalam praktikum ini, sentinelloop memungkinkan kita untuk menghentikan perulangan permainan dengan memberi pilihan kepada pengguna untuk melanjutkan atau berhenti.
5. **Game Penjumlahan**: Permainan penjumlahan menggunakan perulangan while untuk terus meminta pemain memberikan hasil penjumlahan yang benar. Program akan memberi umpan balik apakah jawaban pemain benar atau salah, dan melanjutkan hingga pemain memberikan jawaban yang tepat.

Secara keseluruhan, praktikum ini membantu dalam memahami bagaimana berbagai jenis perulangan bekerja dalam pemrograman dan bagaimana kita dapat menggunakannya untuk membuat program yang lebih dinamis dan interaktif, sesuai dengan kebutuhan program yang kita buat.