LAPORAN PRAKTIKUM ALGORITMA PEMOGRAMAN



Oleh :

Muhammad Alfarizy

NIM 2311532010

ALGORITMA PEMOGRAMAN

PERULANGAN (FOR)

DOSEN PENGAMPU: Dr.Wahyudi, S.T, M.T

FAKULTAS TEKNOLOGI INFORMASI

DEPARTEMEN INFORMATIKA

UNIVERSITAS ANDALAS

1. **Pendahuluan**

Loop for merupakan salah satu struktur pengulangan yang sangat penting dalam pemrograman karena memberikan kemampuan untuk secara efisien menangani tugas-tugas perulangan. Dengan menggunakan loop for, kita dapat melakukan iterasi atau perulangan dengan jelas dan terkontrol, yang memungkinkan kita mengeksekusi suatu blok kode berulang kali sesuai kebutuhan. Praktikum ini dirancang dengan tujuan mendalaminya konsep dasar loop for dan mendemonstrasikan penerapannya dalam pemrograman dengan keterampilan yang perulangan dalam penanganan tugas pemrograman yang melibatkan perulangan terstruktur.

Loop for juga merupakan landasan penting untuk meningkatkan kejelasan dan keterbacaan kode. Dengan menggunakan loop for, kita dapat menulis kode yang lebih singkat dan lebih mudah dipahami, mempermudah pemeliharaan dan pengembangan program secara keseluruhan. Kemampuan untuk mengimplementasikan loop for dengan baik memungkinkan pengembang untuk mengoptimalkan proses dalam program agar program bisa berjalan dengan baik, serta singkat, padat, dan juga jelas.

1. **Tujuan**
2. Memahami serta menerapkan konsep dasar dan sintaksis loop "for" dalam bahasa pemrograman Java.
3. Menggunakan loop "for" untuk melakukan operasi matematika sederhana, seperti menjumlahkan deret angka serta menghasilkan pola pola sesuai dengan kebutuhan yang kita inginkan.
4. **Posedur Kerja**

a. Alat dan bahan

* + 1. Komputer atau laptop
    2. Aplikasi Eclipse
    3. Wifi

b. Langkah

* + 1. Menyusun algoritma yang diperlukan sesuai dengan kebutuhan yang kita inginkan.

1. **Pembahasan**
2. Mencetak angka dari 1 hingga 10.
3. *int i = 1*
4. *i <= 10*: berguna untuk perulangan selama I kurang dari atau sama dengan 10
5. *i++*: berguna untuk pernyataan iterasi
6. Perulangan for untuk mencetak nilai variabel ‘i’ ke konsol 1 - 10. Bedanya, setiap nilai ‘i’ akan di cetak di samping nilai sebelumnya dengan menggunakan ‘*System.out.print’* sehingga menghasilkan output horizontal.
7. *int i = 1*
8. *i <= 10* : berguna sebagai perulangan selama nilai dari I kurang atau sama dengan 10
9. *i++* : berguna sebagai pernyataan iterasi
10. *System.out.print(i + " ")* : Mencetak nilai *i*

1. Perulangan for untuk menjalankan iterasi 1 - 10. Nilai variabel j diakumulasi dengan nilai variabel i.
2. *int j, i;* : Mendeklarasikan variabel j dan variabel i
3. *j=0;*
4. *for(i = 1; i <= 10; i++) { ... }* : Perulangan for
5. *j = j + i;* : Melakukan akumulasi nilai variabel
6. *if (i < 10) { ... }* :
7. *System.out.print("jumlah = " + j);* : Mencetak jumlah total akumulasi.

1. Memasukkan nilai batas melalui konsol.
2. *Scanner input = new Scanner(System.in);* : Mendeklarasikan objek Scanner untuk menerima input dari pengguna.
3. *int Input = input.nextInt();* : Mengambil input nilai dari pengguna.
4. *input.close();* : Menutup objek Scanner setelah selesai mengambil input.

1. Menggunakan nested loop untuk membentuk pola bintang segitiga siku-siku. Yaitu dengan cara memasukkan loop pertama untuk mengontrol baris segitiga, lalu loop kedua untuk mencetak bintang nya.

1. Kesimpulan

a. Perulangan For

Perulangan for mirip dengan membaca setiap item dalam daftar tugas dan melakukan tindakan yang sama untuk setiap item tersebut. Perulangan for berguna ketika kita memiliki daftar tugas atau sekelompok hal yang perlu dilakukan. Contohnya bisa seperti mencetak angka dari 1 hingga 5 atau memproses setiap elemen dalam sebuah list. Perulangan for membantu kita melakukan tugas yang sama untuk setiap elemen dalam daftar tersebut. Misalnya, jika kita memiliki daftar buah, kita dapat melakukan tugas tertentu untuk setiap buah dalam daftar itu.

b. Input Pengguna

Input pengguna dalam algoritma pemrograman adalah cara untuk mendapatkan data dari pengguna (user) sehingga program dapat berinteraksi dengan mereka. Ini memungkinkan program untuk menerima informasi dari pengguna secara langsung. Jadi, input pengguna adalah cara untuk memberdayakan program agar dapat menanggapi perintah atau informasi yang berasal dari pengguna.

c. Nested Loop

Nested loopNested loop sangat berguna ketika kita memiliki tugas-tugas yang terkait satu sama lain dan kita perlu menangani kombinasi semua kemungkinan tugas. Meskipun sangat bermanfaat, perlu diperhatikan bahwa penggunaan nested loop harus dihindari jika memungkinkan, karena dapat membuat program menjadi lebih sulit dipahami dan kurang efisien. adalah konsep di mana satu perulangan ditempatkan di dalam perulangan lain.

d. Akumulasi dan Output Khusus

Akumulasi merujuk pada proses menambahkan atau mengumpulkan nilai dari suatu variabel seiring berjalannya waktu atau iterasi dalam suatu loop. Ini seringkali melibatkan penggunaan variabel penyimpanan yang secara bertahap "mengakumulasi" hasil dari langkah-langkah atau iterasi tertentu. Output khusus merujuk pada cara khusus di mana hasil program atau informasi ditampilkan kepada pengguna. Ini bisa mencakup tata letak atau format tertentu untuk mempresentasikan hasil secara lebih rapi atau sesuai dengan kebutuhan pengguna. akumulasi dan output khusus, sering digunakan dalam pemrograman untuk mengelola data dan memberikan hasil yang lebih informatif atau mudah dipahami bagi pengguna.

e. Kesesuaian Output

Kesesuaian output merujuk pada sejauh mana hasil yang dihasilkan oleh program sesuai dengan yang diinginkan atau diharapkan. Dalam bahasa yang mudah dipahami, kita bisa mengibaratkan ini seperti mencocokkan hasil yang kita harapkan dengan hasil yang sebenarnya seperti membandingkan apa yang kita inginkan dari program. Kesesuaian output sangat penting untuk memastikan bahwa program berfungsi dengan benar. Jika hasil yang dihasilkan oleh program tidak sesuai dengan yang diharapkan, itu bisa menjadi tanda adanya bug atau kesalahan dalam kode. Output yang sesuai memastikan bahwa program memberikan informasi yang benar kepada pengguna atau sistem yang menggunakan program tersebut. Output yang salah dapat mengarah pada keputusan yang salah atau informasi yang tidak akurat. Jadi, kesesuaian output adalah tentang memastikan bahwa hasil yang dihasilkan oleh program sesuai dengan apa yang diinginkan atau diharapkan, sehingga program dapat diandalkan dan memberikan hasil yang benar.