

LAPORAN AKHIR PRAKTIKUM
MATA KULIAH
MODUL 3
JUDUL MODUL : PERULANGAN FOR DAN WHILE



Nama : Kurnia Fajar Rahyudi Putra
No. BP : 2211512013
Hari/Tanggal : Senin/ 22 Mei 2023
Shift : 1

Dosen : Dodon Yendri, M.Kom

LABORATORIUM KOMPUTER DAN JARINGAN
DEPARTEMEN TEKNIK KOMPUTER
FAKULTAS TEKNOLOGI INFORMASI
UNIVERSITAS ANDALAS
PADANG
2023

BAB I

PENDAHULUAN

1.1 Tujuan

1. Memperkenalkan kepada mahasiswa struktur kontrol perulangan/looping
2. Memahami dan mengerti penggunaan struktur kontrol perulangan FOR ..., WHILE
3. Dapat menerapkan struktur kontrol perulangan for ..., while ... untuk menyelesaikan masalah dalam kehidupan sehari-hari.

1.2 Landasan Teori

1.2.1. Definisi Perulangan

Dalam bahasa C++ tersedia suatu fasilitas yang digunakan untuk melakukan proses yang berulang-ulang sebanyak keinginan kita. Misalnya saja, bila kita ingin menginput dan mencetak bilangan dari 1 sampai 100 bahkan 1000, tentunya kita akan merasa kesulitan. Namun dengan struktur perulangan proses, kita tidak perlu menuliskan perintah sampai 100 atau 1000 kali, cukup dengan beberapa perintah saja. Struktur perulangan dalam bahasa C mempunyai bentuk yang bermacam-macam. Sebuah/kelompok instruksi diulang untuk jumlah pengulangan tertentu. Baik yang terdefinisi sebelumnya ataupun tidak. Struktur pengulangan terdiri atas dua bagian :

- (1). Kondisi pengulangan yaitu ekspresi boolean yang harus dipenuhi untuk melaksanakan pengulangan;
- (2). Isi atau badan pengulangan yaitu satu atau lebih pernyataan (aksi) yang akan diulang.

1.2.2. Statement “for”

Pernyataan for digunakan untuk melakukan looping. Pada umumnya looping yang dilakukan oleh for telah diketahui batas awal, syarat looping dan perubahannya. Selama kondisi terpenuhi, maka

pernyataan akan terus dieksekusi. Bentuk umum pernyataan for sebagai berikut :

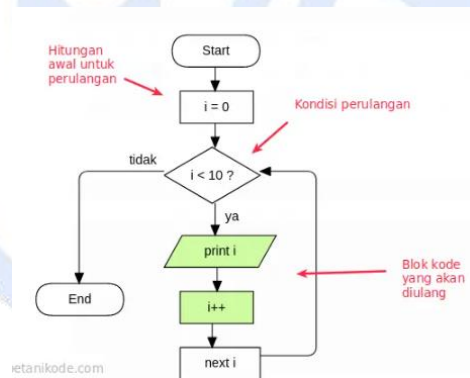
```
for (inisialisasi; syarat pengulangan; pengubah nilai pencacah)
```

Bila pernyataan didalam for lebih dari satu maka pernyataan-pernyataan tersebut harus diletakkan didalam tanda kurung.

```
for ( inisialisasi; syarat pengulangan; pengubah nilai pencacah ) {  
    pernyataan / perintah;  
    pernyataan / perintah;  
    pernyataan / perintah;  
}
```

Kegunaan dari masing-masing argumen for diatas adalah :

- Inisialisasi: merupakan bagian untuk memberikan nilai awal untuk variabel-variabel tertentu.
- Syarat Pengulangan: memegang kontrol terhadap pengulangan, karena bagian ini yang akan menentukan suatu pengulangan diteruskan atau dihentikan.
- Pengubah Nilai Pencacah: mengatur kenaikan atau penurunan nilai pencacah.



Flowchart pengulangan for

1.2.3. Statement “while”

Perulangan WHILE banyak digunakan pada program yang terstruktur. Perulangan ini banyak digunakan bila jumlah perulangannya belum diketahui. Proses perulangan akan terus

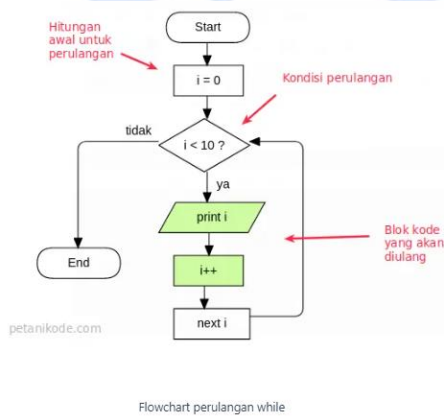
berlanjut selama kondisinya bernilai benar ($\neq 0$) dan akan berhenti bila kondisinya bernilai salah ($=0$).

Bentuk umum perulangan while sebagai berikut:

```
while ( syarat )  
    Pernyataan / perintah ;
```

Bentuk umum perulangan while, dengan lebih dari perintah / pernyataan, sebagai berikut:

```
while ( syarat ) {  
    Pernyataan / perintah ;  
    Pernyataan / perintah ;  
}
```



1.2.4. Statement “do while”

Pernyataan perulangan do-while merupakan bentuk perulangan yang melaksanakan perulangan terlebih dahulu dan pengujian perulangan dilakukan dibelakang. Bentuk umum perulangan do-while, sebagai berikut :

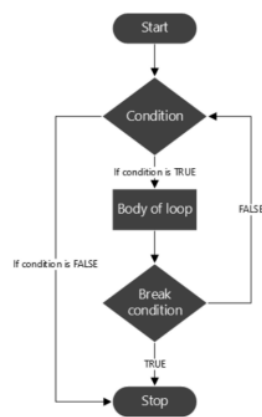
```
do  
    pernyataan / perintah ;  
while ( syarat );
```

Bentuk umum perulangan do-while, dengan lebih dari perintah / pernyataan, sebagai berikut:

```
do {  
    Pernyataan / perintah;  
    Pernyataan / perintah;  
} while ( syarat )
```

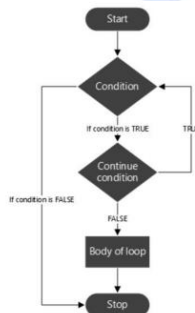
1.2.5. Pernyataan “break”

Pernyataan break telah dibahas pada pernyataan pengambilan keputusan switch. Pernyataan break ini berfungsi untuk keluar dari struktur switch. Selain itu pernyataan break berfungsi keluar dari perulangan (for, while dan do-while). Jika pernyataan break dikerjakan, maka eksekusi akan dilanjutkan ke pernyataan yang terletak sesudah akhir dari badan perulangan (loop).



1.2.6. Pernyataan “continue”

Pernyataan continue digunakan untuk mengarahkan eksekusi ke iterasi (proses) berikutnya pada loop yang sama, dengan kata lain mengembalikan proses yang sedang dilaksanakan ke-awal loop lagi, tanpa menjalankan sisa perintah dalam loop tersebut.



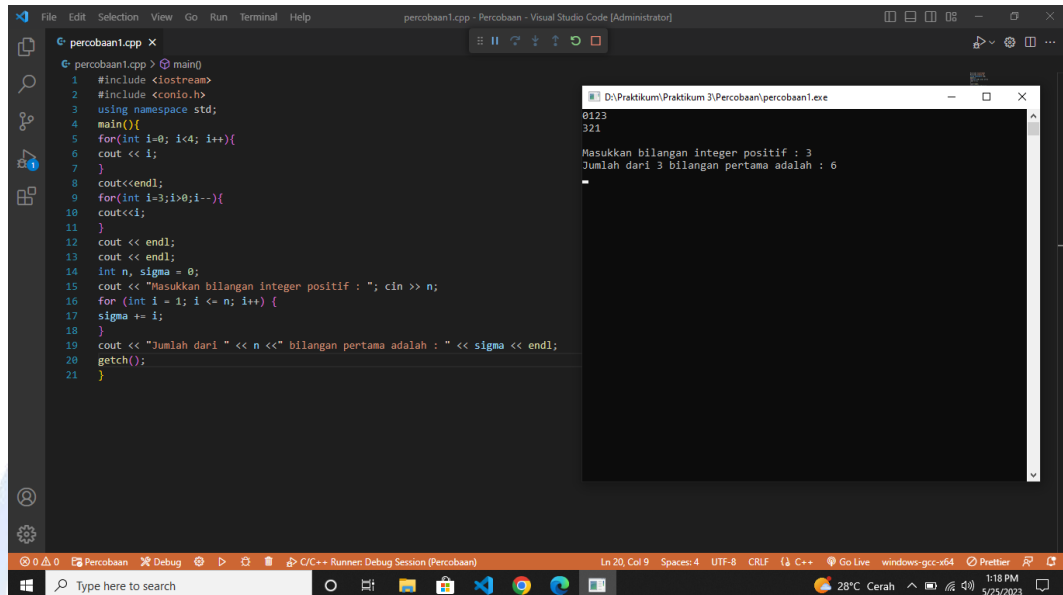
1.2.7. Nested loop

Pernyataaan Nested Loop adalah suatu perulangan didalam perulangan yang lainnya.

BAB II

PERCOBAAN

2.1 Percobaan 3.1

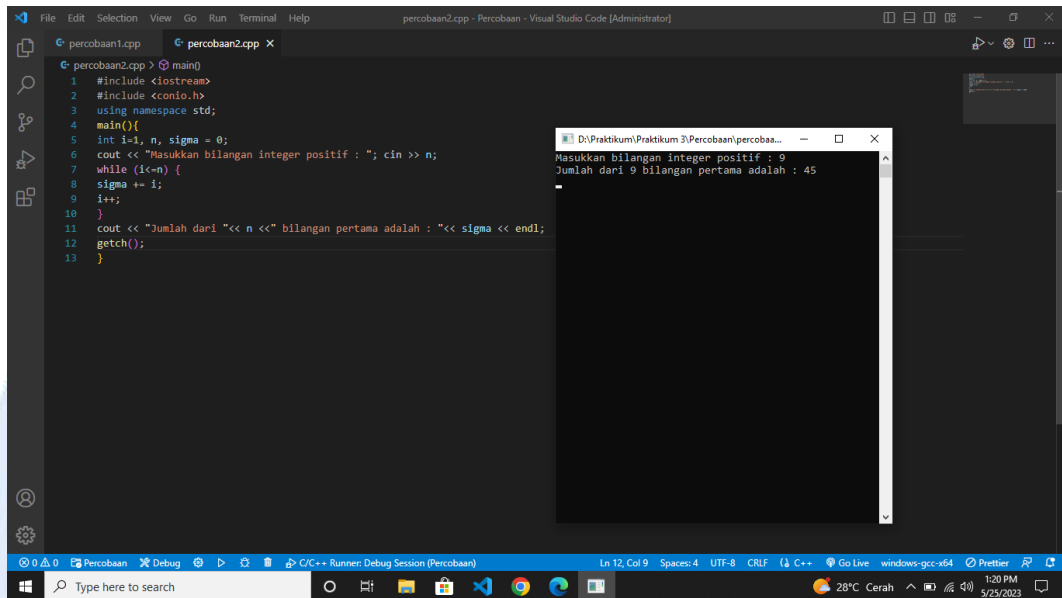


The screenshot shows the Visual Studio Code editor with a C++ file named `percobaan1.cpp`. The code includes `<iostream>` and `<conio.h>`, uses the `std` namespace, and defines a `main` function. It contains two `for` loops: the first loops from `i=0` to `i<4` (4 iterations), and the second loops from `i=3` down to `i>0` (3 iterations). Both loops output their values with `cout << i;` and `cout << endl;`. After the loops, two variables are declared: `int n, sigma = 0;`. A prompt is shown: `cout << "Masukkan bilangan integer positif : "; cin >> n;`. A `for` loop then calculates the sum of integers from 1 to `n` using `sigma += i;`. Finally, the result is printed: `cout << "Jumlah dari " << n << " bilangan pertama adalah : " << sigma << endl;`. The output window shows the execution results: `0123`, `321`, the prompt `Masukkan bilangan integer positif : 3`, and the result `Jumlah dari 3 bilangan pertama adalah : 6`.

1. **for(int i=0; i<4; i++)** adalah loop for yang akan mengulang iterasi sebanyak 4 kali. Pada setiap iterasi, nilai dari variabel `i` akan ditampilkan menggunakan `cout << i;`. `cout<<endl;` digunakan untuk memindahkan kursor ke baris baru setelah loop pertama selesai.
2. **for(int i=3;i>0;i--)** adalah loop for yang akan mengulang iterasi sebanyak 3 kali. Pada setiap iterasi, nilai dari variabel `i` akan ditampilkan menggunakan `cout << i;`. `cout << endl;` digunakan untuk memindahkan kursor ke baris baru setelah loop kedua selesai.
3. **int n, sigma = 0;** mendeklarasikan dua variabel yaitu `n` dan `sigma`, dengan tipe data integer. Variabel `n` akan digunakan untuk menyimpan bilangan integer positif yang dimasukkan pengguna, dan variabel `sigma` akan digunakan untuk menjumlahkan bilangan-bilangan dari 1 hingga `n`. `cout << "Masukkan bilangan integer positif : "; cin >> n;` digunakan untuk menampilkan pesan kepada pengguna agar memasukkan bilangan integer positif. Nilai yang dimasukkan akan disimpan dalam variabel `n` menggunakan `cin >> n;`.

4. **for (int i = 1; i <= n; i++)** adalah loop for yang akan mengulang iterasi sebanyak n kali. Pada setiap iterasi, nilai dari variabel i akan ditambahkan ke variabel sigma menggunakan **sigma += i**; **cout << "Jumlah dari " << n << " bilangan pertama adalah : " << sigma << endl**; digunakan untuk menampilkan jumlah dari n bilangan pertama yang dihitung sebelumnya.

2.2 Percobaan 3.1



The screenshot shows the Visual Studio Code editor with a C++ file named `percobaan2.cpp`. The code is as follows:

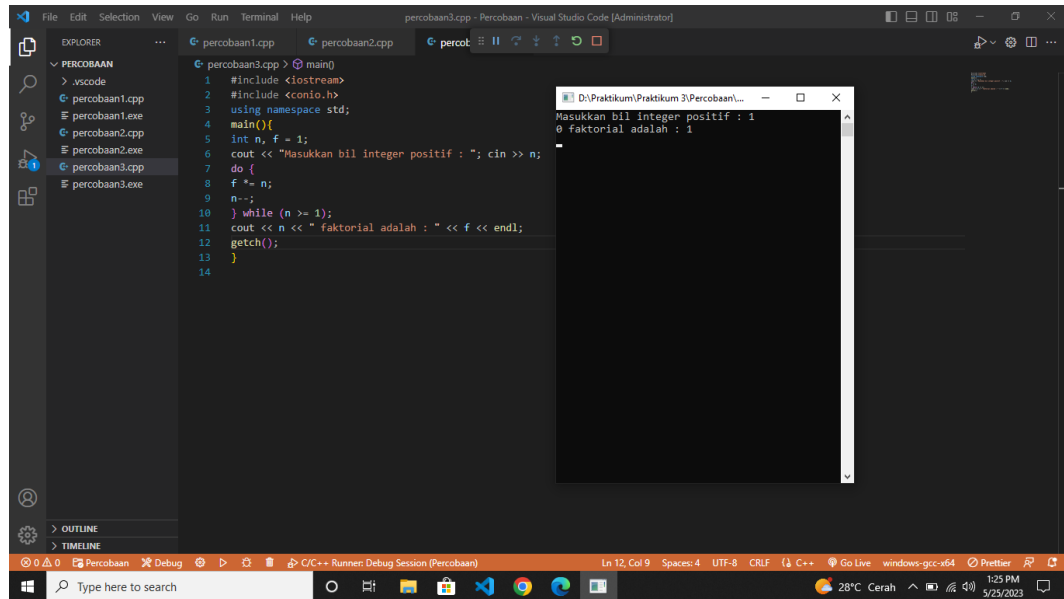
```
1 #include <iostream>
2 #include <conio.h>
3 using namespace std;
4 main()
5 {
6     int i=1, n, sigma = 0;
7     cout << "Masukkan bilangan integer positif : "; cin >> n;
8     while (i<=n) {
9         sigma += i;
10        i++;
11    }
12    cout << "Jumlah dari "<< n << " bilangan pertama adalah : "<< sigma << endl;
13    getch();
14 }
```

The output window shows the program's execution:

```
Masukkan bilangan integer positif : 9
Jumlah dari 9 bilangan pertama adalah : 45
```

1. Pada awal program dideklarasikan nilai **i = 1**, **n**, dan **sigma**, kemudian program menampilkan pesan kepada user menggunakan **cout << "Masukkan bilangan Integer positif : "**, dan meminta user memasukan sebuah bilangan yang disimpan kedalam **cin >> n**;
2. **while (i<=n)** adalah sebuah loop while yang akan berjalan selama nilai i kurang dari atau sama dengan n. Loop ini digunakan untuk menjumlahkan bilangan-bilangan dari 1 hingga n. Pada setiap iterasi, nilai dari variabel i akan ditambahkan ke variabel sigma menggunakan, dan nilai i akan ditingkatkan sebesar 1 dengan **i++**;
3. **cout << "Jumlah dari " << n << " bilangan pertama adalah : " << sigma << endl**; digunakan untuk menampilkan jumlah dari n bilangan pertama yang dihitung sebelumnya.

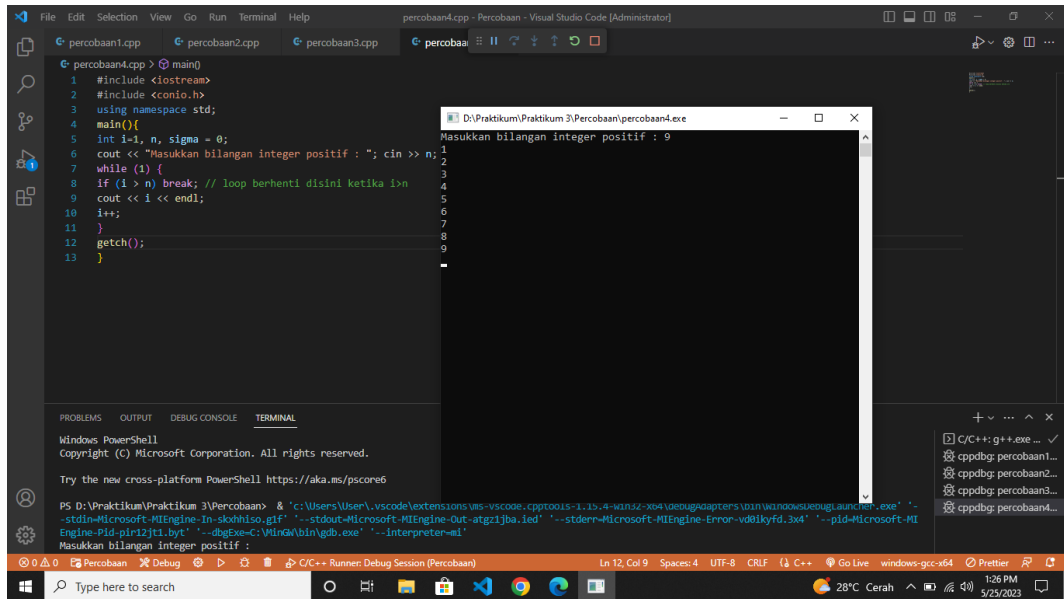
2.3 Percobaan 3.3



```
1 #include <iostream>
2 #include <conio.h>
3 using namespace std;
4 main(){
5     int n, f = 1;
6     cout << "Masukkan bil integer positif : "; cin >> n;
7     do {
8         f *= n;
9         n--;
10    } while (n >= 1);
11    cout << n << " faktorial adalah : " << f << endl;
12    getch();
13 }
```

1. **int n, f = 1;** adalah deklarasi dua variabel. Variabel n digunakan untuk menyimpan bilangan integer positif yang dimasukkan oleh pengguna, dan variabel f diinisialisasi dengan nilai 1. **cout << "Masukkan bil integer positif : "; cin >> n;** digunakan untuk menampilkan pesan kepada pengguna agar memasukkan bilangan integer positif. Nilai yang dimasukkan oleh pengguna akan disimpan dalam variabel n menggunakan **cin >> n;**.
2. **do { ... } while (n >= 1);** adalah konstruksi do-while loop yang akan terus berjalan selama nilai n lebih besar dari atau sama dengan 1. Loop ini digunakan untuk menghitung faktorial dari n. Pada setiap iterasi, nilai dari variabel n akan dikalikan dengan variabel f menggunakan **f *= n;**, dan nilai n akan dikurangi 1 dengan **n--;**.
3. Setelah loop selesai, program akan menampilkan nilai dari variabel n dan variabel f menggunakan **cout << n << " faktorial adalah : " << f << endl;**. Hal ini akan menampilkan hasil faktorial dari bilangan n yang dihitung sebelumnya.

2.4 Percobaan 3.4



```
1 #include <iostream>
2 #include <conio.h>
3 using namespace std;
4 main()
5 {
6     int i=1, n, sigma = 0;
7     cout << "Masukkan bilangan integer positif : "; cin >> n;
8     while (1) {
9         if (i > n) break; // loop berhenti disini ketika i>n
10        cout << i << endl;
11        i++;
12    }
13 }
```

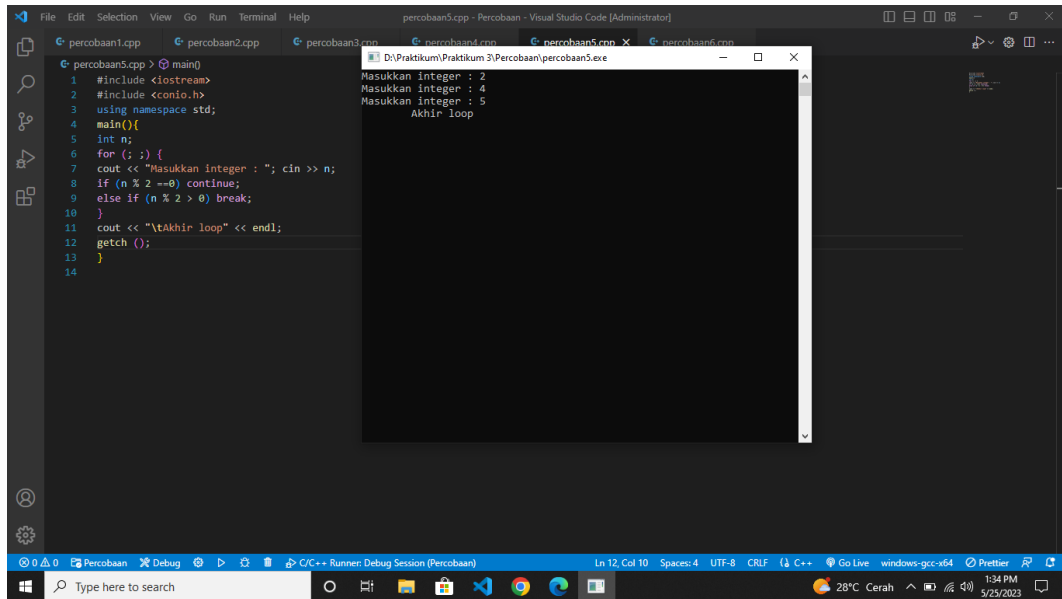
Windows PowerShell
Copyright (c) Microsoft Corporation. All rights reserved.
Try the new cross-platform PowerShell <https://aka.ms/pscore6>

PS D:\Praktikum\Praktikum 3\Percobaan> & "c:\Users\User\.vscode\extensions\ms-vscode.cpptools-1.15.4-win32-x64\debugadapters\win\windowsdebuglauncher.exe" "-.stdin-stdout-Microsoft-MiEngine-In-siohniso.gif" "-.stdout-Microsoft-MiEngine-Out-argz1jba.1ed" "-.stderr-Microsoft-MiEngine-Error-vd8ikyfd.3x4" "-.pid-Microsoft-MiEngine-Pid-pir12jtl.by" "-.dbgexe-c:\Windows\bin\gdb.exe" "-.interpreter-mi"

Masukkan bilangan integer positif :
9
1
2
3
4
5
6
7
8
9

1. **int i=1, n, sigma = 0;** adalah deklarasi beberapa variabel. **cout << "Masukkan bilangan integer positif : "; cin >> n;** digunakan untuk menampilkan pesan kepada pengguna agar memasukkan bilangan integer positif. Nilai yang dimasukkan oleh pengguna akan disimpan dalam variabel **n** menggunakan **cin >> n;**.
2. **while (1) { ... }** adalah loop while yang akan berjalan selama kondisi di dalamnya bernilai true. Dalam hal ini, kondisi yang digunakan adalah **1**, yang akan selalu bernilai true. Oleh karena itu, loop ini akan terus berjalan sampai ada perintah untuk menghentikannya. Di dalam loop, pertama-tama dilakukan pengecekan dengan if statement. **Jika nilai i lebih besar dari n**, maka program akan keluar dari loop menggunakan **break;**. Ini berarti loop akan berhenti saat nilai **i** melebihi **n**. Jika kondisi **if tidak terpenuhi**, maka program akan mencetak nilai **i** menggunakan **cout << i << endl;**, kemudian nilai **i** akan ditingkatkan sebesar 1 dengan **i++;**.

2.5 Percobaan 3.5



The screenshot shows the Visual Studio Code editor with a C++ file named `percobaan5.cpp`. The code is as follows:

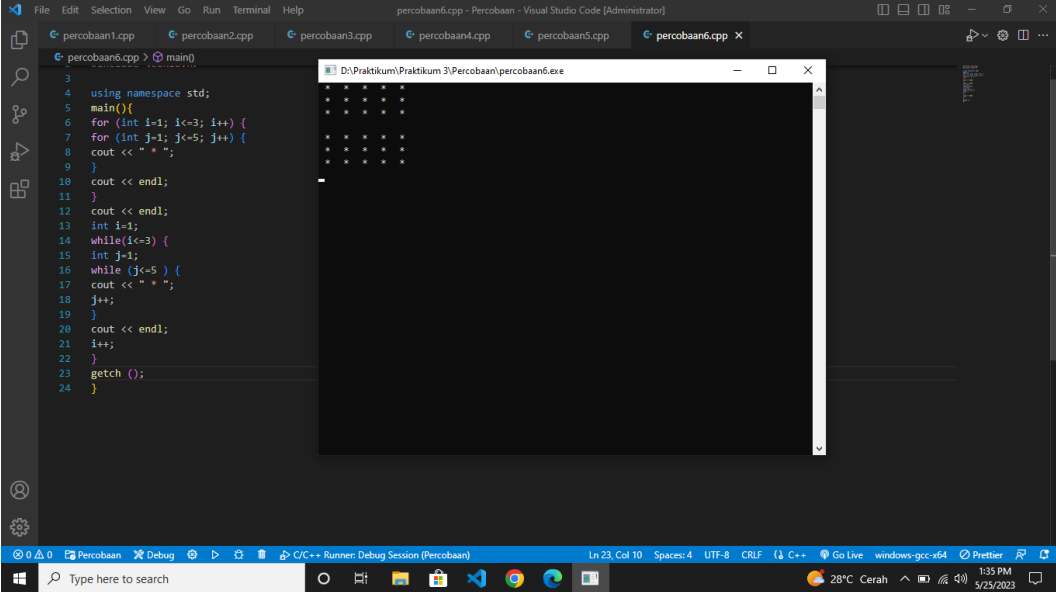
```
1 #include <iostream>
2 #include <conio.h>
3 using namespace std;
4 main()
5 {
6     int n;
7     for (; ) {
8         cout << "Masukkan integer : "; cin >> n;
9         if (n % 2 == 0) continue;
10        else if (n % 2 > 0) break;
11    }
12    cout << "\tAkhir loop" << endl;
13    getch();
14 }
```

The output window shows the program's execution:

```
Masukkan integer : 2
Masukkan integer : 4
Masukkan integer : 5
\tAkhir loop
```

1. Pada awal program terdapat deklarasi `n` untuk menyimpan nilai integer yang dimasukan pengguna.
2. Terdapat sebuah loop `for` yang tidak memiliki kondisi awal, kondisi perulangan, dan perubahan iterasi. Artinya loop ini akan berhenti ketika diberi perintah untuk berhenti.
3. Di dalam loop, pertama-tama program akan menampilkan pesan kepada pengguna agar memasukkan sebuah bilangan integer menggunakan `cout << "Masukkan integer : ";`. Setelah itu, program akan mengambil input dari pengguna dan menyimpannya dalam variabel `n` menggunakan `cin >> n;`
4. Setelah pengguna memasukkan bilangan, dilakukan pengecekan dengan `if` statement. Jika nilai **`n modulo 2` (`n % 2`) sama dengan 0**, yang berarti `n` adalah bilangan genap, maka program akan melanjutkan ke iterasi berikutnya menggunakan `continue;`. Ini berarti program akan kembali ke awal loop tanpa mengeksekusi perintah-perintah di bawahnya. Jika kondisi `if` pertama tidak terpenuhi, maka program akan melakukan pengecekan pada kondisi `else if`. Jika nilai **`n modulo 2 lebih besar dari 0`**, yang berarti `n` adalah bilangan ganjil, maka program akan keluar dari loop menggunakan `break;`. Ini berarti program akan melanjutkan eksekusi ke perintah-perintah setelah loop.
5. Setelah loop selesai, program akan mencetak pesan "Akhir loop" diikuti dengan baris baru menggunakan `cout << "\tAkhir loop" << endl;`.

2.6 Percobaan 3.6



The screenshot shows the Visual Studio Code editor with a C++ file named `percobaan6.cpp`. The code uses nested loops to print a pattern of stars. The output window shows the result of the program execution, which is a 3x5 grid of stars.

```
3 using namespace std;
4 main(){
5     for (int i=1; i<=3; i++) {
6         for (int j=1; j<=5; j++) {
7             cout << " * ";
8         }
9         cout << endl;
10    }
11    cout << endl;
12    int i=1;
13    while(i<=3) {
14        int j=1;
15        while (j<=5 ) {
16            cout << " * ";
17            j++;
18        }
19        cout << endl;
20        i++;
21    }
22    }
23    getch ();
24 }
```

```
* * * * *
* * * * *
* * * * *

* * * * *
* * * * *
* * * * *
```

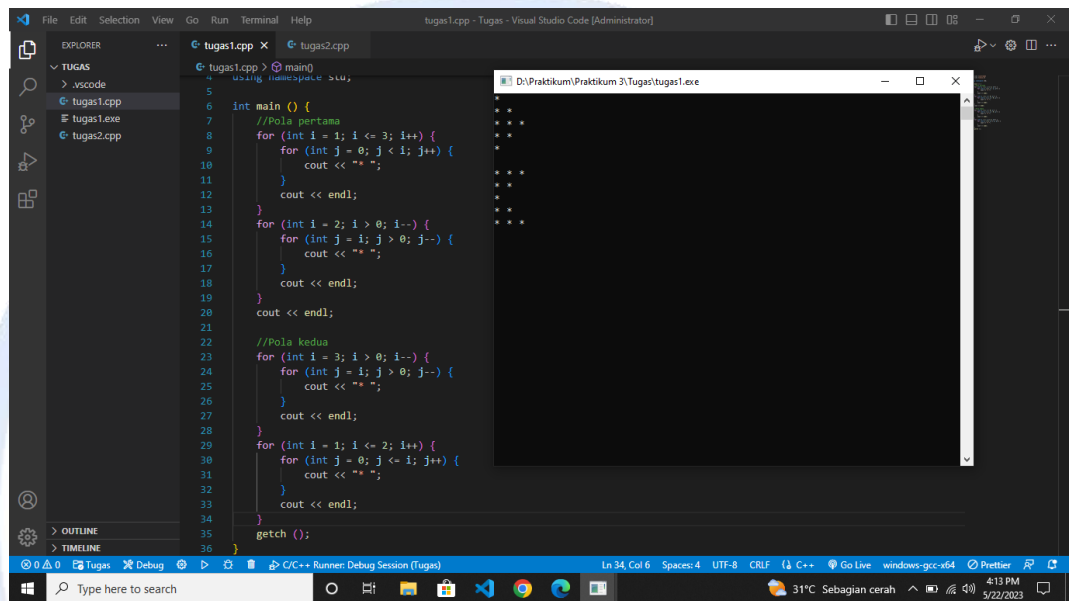
1. Pertama, terdapat loop for yang akan berjalan sebanyak 3 kali (**i=1; i<=3; i++**). Loop ini berfungsi untuk mencetak baris bintang. Di dalam loop for pertama, terdapat loop for kedua yang juga akan berjalan sebanyak 5 kali (**j=1; j<=5; j++**). Loop ini berfungsi untuk mencetak kolom bintang di setiap baris. Di dalam loop for kedua, program mencetak satu bintang dengan perintah **cout << " * "**. Setelah mencetak 5 bintang di dalam loop for kedua, program mencetak baris baru dengan perintah **cout << endl**. Hal ini akan menghasilkan pola bintang dengan 5 kolom dan 3 baris.
2. Setelah loop for pertama selesai, program mencetak baris kosong menggunakan **cout << endl**. Hal ini digunakan untuk memberikan jarak antara dua pola bintang yang dicetak menggunakan loop for dan loop while.
3. Kemudian, terdapat loop while yang akan berjalan sebanyak 3 kali (**i<=3**). Loop ini bertanggung jawab untuk mencetak baris bintang. Di dalam loop while, terdapat loop while kedua yang juga akan berjalan sebanyak 5 kali (**j<=5**). Loop ini bertanggung jawab untuk mencetak kolom bintang di setiap baris. Di dalam loop while kedua, program mencetak satu bintang dengan perintah **cout << " * "**. Setelah mencetak 5 bintang di dalam loop while kedua, program mencetak baris baru dengan perintah **cout << endl**. Hal ini akan menghasilkan pola bintang dengan 5 kolom dan 3 baris.

4. Setelah loop while selesai, program mencetak baris kosong menggunakan **cout << endl;**. Hal ini digunakan untuk memberikan jarak antara pola bintang menggunakan loop for dan loop while.

BAB III

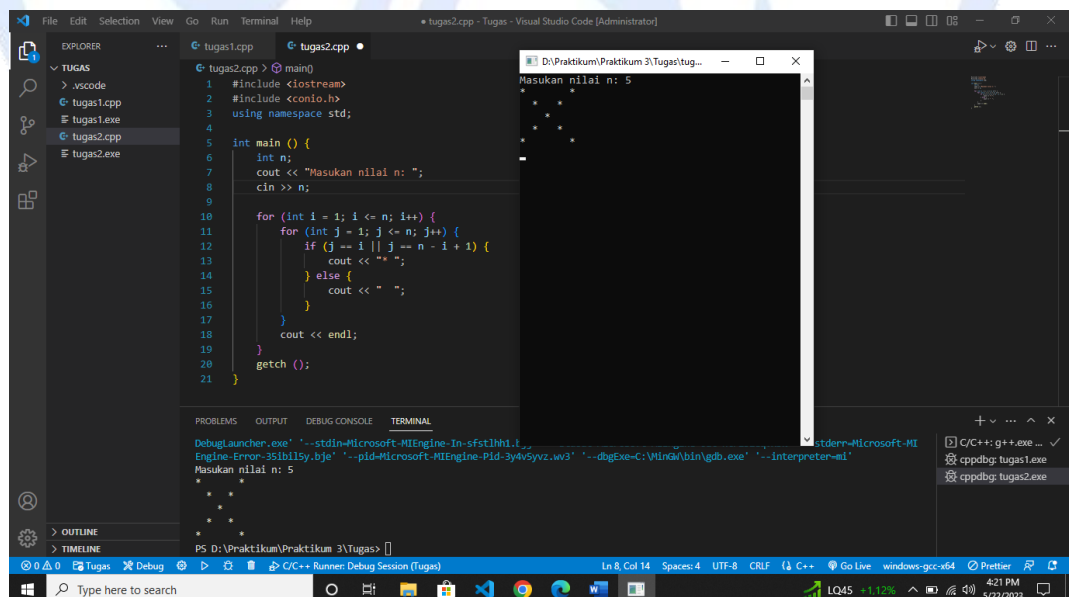
TUGAS PRAKTIKUM

3.1 Tugas 1



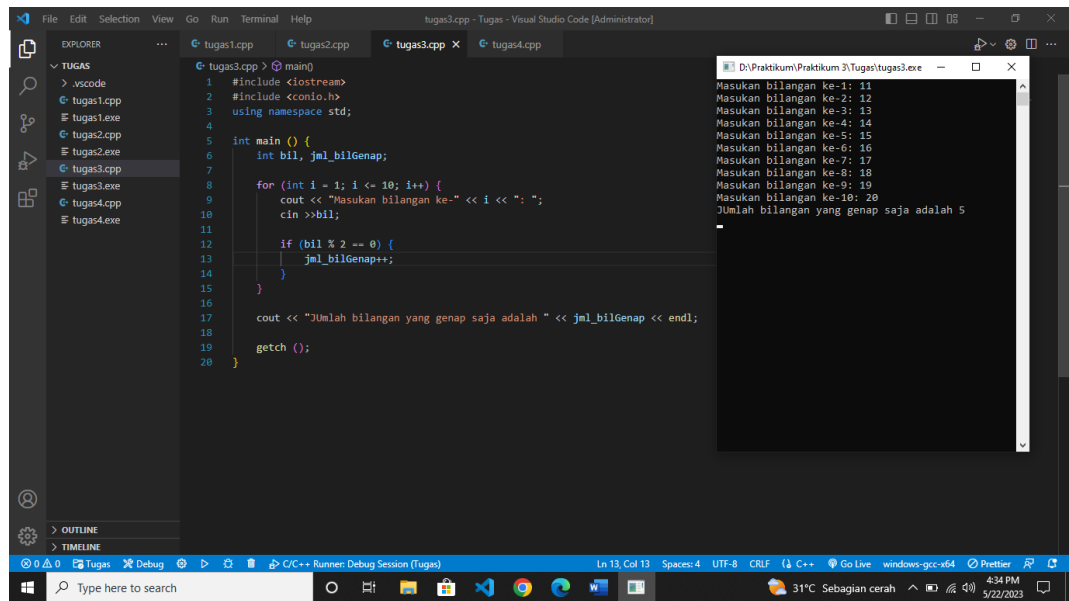
```
1 #include <iostream>
2 #include <conio.h>
3 using namespace std;
4
5 int main () {
6     //Pola pertama
7     for (int i = 1; i <= 3; i++) {
8         for (int j = 0; j < 1; j++) {
9             cout << " * ";
10        }
11        cout << endl;
12    }
13    for (int i = 2; i > 0; i--) {
14        for (int j = i; j > 0; j--) {
15            cout << " * ";
16        }
17        cout << endl;
18    }
19    cout << endl;
20
21    //Pola kedua
22    for (int i = 3; i > 0; i--) {
23        for (int j = 1; j > 0; j--) {
24            cout << " * ";
25        }
26        cout << endl;
27    }
28    for (int i = 1; i <= 2; i++) {
29        for (int j = 0; j <= i; j++) {
30            cout << " * ";
31        }
32        cout << endl;
33    }
34    getch ();
35 }
```

3.2 Tugas 2

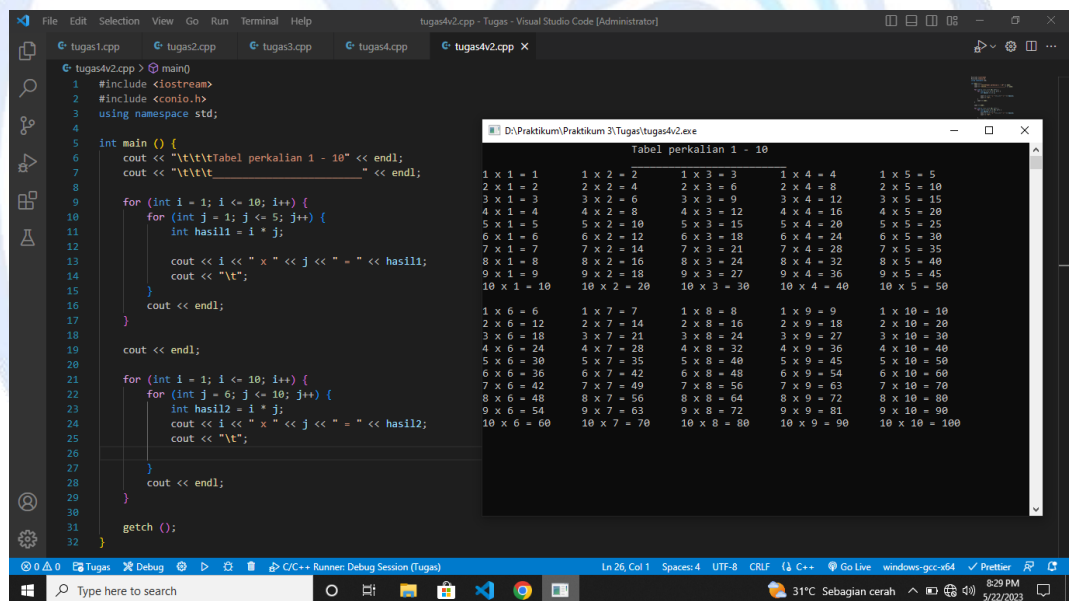


```
1 #include <iostream>
2 #include <conio.h>
3 using namespace std;
4
5 int main () {
6     int n;
7     cout << "Masukan nilai n : ";
8     cin >> n;
9
10    for (int i = 1; i <= n; i++) {
11        for (int j = 1; j <= n; j++) {
12            if (j == i || j == n - i + 1) {
13                cout << " * ";
14            } else {
15                cout << " ";
16            }
17        }
18        cout << endl;
19    }
20    getch ();
21 }
```

3.3 Tugas 3



3.4 Tugas 4



BAB IV ANALISA

4.1 Tugas x

4.1.1 Poin A

Untuk menampilkan masing-masing pola, dibutuhkan nested loop, dimana pada program tersebut variabel *i* digunakan untuk mengontrol jumlah baris, dan variabel *j* untuk mengontrol kolom pada masing-masing pola.

4.1.2 Poin B

Untuk membuat pola pertama, dibutuhkan dua buah nested loop dengan variabel *i* dan variabel *j*. Pada nested loop pertama, variabel *i* digunakan untuk mengontrol jumlah baris yaitu sebanyak 3 baris, setiap iterasi maka akan dibuat loop yang baru untuk mengontrol kolom dengan variabel *j*, jumlah kolom yaitu lebih kecil dari jumlah baris, jika pada baris pertama (*i* = 1) maka jumlah kolom adalah 0, dst.

Pada nested loop kedua, variabel *i* digunakan untuk mengontrol jumlah baris yaitu sebanyak 2 baris, setiap iterasi maka akan dibuat loop yang baru untuk mengontrol kolom dengan variabel *j*, jumlah kolom yaitu sama besar dengan jumlah baris.

4.1.3 Poin C

Untuk membuat pola pertama, dibutuhkan dua buah nested loop dengan variabel *i* dan variabel *j*. Pada nested loop pertama, variabel *i* digunakan untuk mengontrol jumlah baris yaitu sebanyak 3 baris, setiap iterasi maka akan dibuat loop yang baru untuk mengontrol kolom dengan variabel *j*, jumlah kolom yaitu sama besar dengan jumlah baris.

Pada nested loop kedua, variabel *i* digunakan untuk mengontrol jumlah baris yaitu sebanyak 2 baris, setiap iterasi maka akan dibuat loop yang baru untuk mengontrol kolom dengan variabel *j*, jumlah kolom yaitu kecil sama dengan jumlah baris.

4.2 Tugas 2

4.2.1 Poin A

Pada awal program dideklarasikan *n*, kemudian program menampilkan perintah kepada user untuk memasukan sebuah nilai dengan **cout <<**

“Masukan nilai n : “;, kemudian nilai yang dimasukan user akan disimpan dalam **cin >> n;**

4.2.2 Poin B

Dalam program tersebut terdapat nested loop, dalam loop pertama **for (int i = 1; i <= n; i++)** terdapat sebuah loop **for (int j = 1; j <= n; j++)**, setiap iterasi pada loop ini akan masuk ke dalam kondisi if.

4.2.3 Poin C

Jika kondisi **j == i** atau **j == n - i + 1** terpenuhi maka akan menghasilkan output **“* “** (bintang dan satu spasi), jika kondisi tidak terpenuhi maka akan menghasilkan output **“ “** (dua spasi).

4.3 Tugas 3

4.3.1 Poin A

Pada awal program dideklarasikan variabel **bil** dan **jml_bilGenap**

4.3.2 Poin B

Terdapat sebuah loop **for (int i = 1; i <= 10; i++)** untuk membuah perulangan dengan loop dimulai dari 1 dan diakhiri dengan 10. Untuk setiap iterasi akan ditampilkan sebuah perintah **cout << “Masukan bilangan ke- “ << i << “: “;** yang meminta user memasukan sebuah bilangan acak yang kemudian disimpan dalam **cin >> bil;**. Di setiap iterasi, program akan melakukan pengecekan terhadap setiap bilangan yang diinputkan oleh user, jika bilangan merupakan bilangan genap (**bil % 2 == 0**), maka **jml_bilGenap** akan ditambah.

4.3.3 Poin C

Terakhir program akan menampilkan jumlah bilangan yang merupakan bilangan genap saja dengan **cout << “Jumlah bilangan yang genap saja adalah “ << jml_bilGenap << endl;**.

4.4 Tugas 4

4.4.1 Poin A

Program mencetak judul tabel menggunakan **cout << “\t\tTabel perkalian 1 - 10” << endl;** dan garis pemisah menggunakan **cout << “\t\t_____” << endl;**.

4.4.2 Poin B

Pertama, terdapat loop for pertama yang akan berjalan sebanyak 10 kali ($i=1; i \leq 10; i++$). Loop ini bertanggung jawab untuk mencetak baris tabel pertama, yaitu perkalian dari 1 hingga 5. Di dalam loop for pertama, terdapat loop for kedua yang juga akan berjalan sebanyak 5 kali ($j=1; j \leq 5; j++$). Loop ini bertanggung jawab untuk mencetak kolom tabel. Di dalam loop for kedua, program menghitung hasil perkalian antara i dan j dengan menyimpannya dalam variabel `hasil1` menggunakan `int hasil1 = i * j;`. Setelah itu, program mencetak hasil perkalian dengan format `cout << i << " x " << j << " = " << hasil1;`. Tanda `"\t"` digunakan untuk memberikan jarak antar kolom. Setelah mencetak setiap kolom dalam satu baris, program mencetak baris baru dengan `cout << endl;`. Setelah loop for pertama selesai, program mencetak baris kosong menggunakan `cout << endl;`. Hal ini digunakan untuk memberikan jarak antara dua bagian tabel perkalian.

4.4.3 Poin C

Kemudian, terdapat loop for kedua yang akan berjalan sebanyak 10 kali ($i=1; i \leq 10; i++$). Loop ini bertanggung jawab untuk mencetak baris tabel kedua, yaitu perkalian dari 6 hingga 10. Di dalam loop for kedua, terdapat loop for ketiga yang juga akan berjalan sebanyak 5 kali ($j=6; j \leq 10; j++$). Loop ini bertanggung jawab untuk mencetak kolom tabel. Di dalam loop for ketiga, program menghitung hasil perkalian antara i dan j dengan menyimpannya dalam variabel `hasil2` menggunakan `int hasil2 = i * j;`. Setelah itu, program mencetak hasil perkalian dengan format `cout << i << " x " << j << " = " << hasil2;`. Tanda `"\t"` digunakan untuk memberikan jarak antar kolom. Setelah mencetak setiap kolom dalam satu baris, program mencetak baris baru dengan `cout << endl;`. Setelah loop for kedua selesai, program mencetak baris kosong menggunakan `cout << endl;`. Hal ini digunakan untuk memberikan jarak antara tabel perkalian dan akhir program.

BAB V

PENUTUP

5.1 Kesimpulan

Dalam praktikum ini, kami telah mempelajari dan mengimplementasikan loop dalam bahasa pemrograman C++. Loop adalah sebuah konstruksi yang memungkinkan kita untuk mengulang sejumlah pernyataan atau blok kode secara berulang sesuai dengan kondisi yang ditentukan. Dalam looping terdapat beberapa statement, yaitu:

1. Statement for
2. Statement while
3. Statement do while
4. Pernyataan break
5. Pernyataan continue
6. Dan nested loop

Selama praktikum, kami telah berhasil mengimplementasikan dan menguji semua jenis loop yang disebutkan di atas. Kami juga belajar bagaimana menggunakan pernyataan pengontrol loop, seperti break untuk menghentikan loop secara paksa dan continue untuk melanjutkan ke iterasi berikutnya. Melalui praktikum ini, dapat disimpulkan bahwa loop adalah alat yang sangat berguna dalam pemrograman, karena memungkinkan kita untuk mengulang tugas-tugas tertentu secara efisien. Dengan pemahaman yang baik tentang loop, kami dapat menulis kode yang lebih efektif dan mengatasi tantangan pemrograman yang melibatkan pengulangan.

5.2 Saran

Sebelum memulai membuat program, hendaknya mempelajari dan memahami proses dan cara kerja dari setiap materi yang akan dipraktikan agar tidak terjadi error, sehingga program dapat dijalankan dengan lancar.

DAFTAR PUSTAKA

Anonim. (2023). *Modal 3 Praktikum Algoritma dan Pemecahan Masalah*. Padang: Laboratorium Komputer dan Jaringan.

Anonoim. (2023). *Dasar Pemrograman Modul 4 Perulangan*.

Kode, P. (2019, Desember 20). Retrieved from <https://www.petanikode.com/cpp-perulangan/>

