

**LAPORAN PRAKTIKUM**  
**Modul 1**  
**“Pengenalan Bahasa C++ Bagian Pertama”**



**Disusun Oleh:**  
**Benedictus Qsota Noventino Baru - 2311104029**  
**S1SE07A**

**Dosen :**  
**Yudha Islami Sulistya, S.Kom., M.Cs**

**PROGRAM STUDI S1 SOFTWARE ENGINEERING**  
**FAKULTAS INFORMATIKA**  
**INSTITUT TEKNOLOGI TELKOM**  
**PURWOKERTO**  
**2024**

1. (Input/Output) Tuliskan kode berikut dan jalankan. a) Masukkan nama lengkap anda dan nim anda. Screenshot kode dan hasilnya, lalu tempelkan pada jawaban. b) Masukkan nama pertama anda dan nim anda. Screenshot kode dan hasilnya, lalu tempelkan pada jawaban.

```
main.cpp x
1  #include <iostream>
2
3  using namespace std;
4
5  int main(){
6      string nama, nim;
7      cout << "Siapa nama anda? ";
8      cin >> nama;
9      cout << "Berapa nim anda? ";
10     cin >> nim;
11     cout << "Nama saya:" << nama << endl;
12     cout << "NIM saya:" << nim << endl;
13     return 0;
14 }
```

Jawab :

```
#include <iostream>

using namespace std;

int main ()
{
    string nama, nim;
    cout << "Siapa nama anda ?";
    cin >> nama;
    cout << "Berapa nim anda ?";
    cin >> nim;
    cout << "Nama saya : " << nama << endl;
    cout << "NIM saya : " << nim << endl;
    return 0;
}
```

```
Siapa nama anda ?Novennn
Berapa nim anda ?2311104029
Nama saya : Novennn
NIM saya : 2311104029
PS C:\Users\noven\Desktop\Struktur Data Praktek>
```

Kodingan ini adalah program sederhana dalam C++ yang meminta input dari pengguna berupa nama dan NIM (Nomor Induk Mahasiswa), lalu menampilkan kembali input tersebut. Program dimulai dengan mendeklarasikan variabel `nama` dan `nim` dengan tipe data `string`. Setelah itu, program menampilkan pesan "Siapa nama anda?" dan "Berapa nim anda?" untuk meminta pengguna memasukkan nilai untuk variabel `nama` dan `nim`. Input yang diterima akan disimpan ke dalam variabel yang sesuai menggunakan operator `cin`. Terakhir, program menampilkan output berupa "Nama saya: [nama]" dan "NIM saya: [nim]" menggunakan `cout`, dan program berakhir dengan `return 0` yang menunjukkan bahwa program selesai dijalankan dengan sukses.

2. (Operasi aritmatika) Tuliskan kode berikut dan jalankan. Screenshot kode dan hasilnya, lalu tempelkan pada jawaban.

```
main.cpp x
1  #include <iostream>
2
3  using namespace std;
4
5  int main(){
6      int bil1 = 3, bil2 = 4, hasil1;
7      float bil3 = 3.0, bil4 = 4.0, hasil2;
8      hasil1 = bil1 + bil2;
9      cout << hasil1 << endl;
10     hasil1 = bil1 - bil2;
11     cout << hasil1 << endl;
12     hasil1 = bil1 * bil2;
13     cout << hasil1 << endl;
14     hasil1 = bil1 / bil2; // integer division
15     cout << hasil1 << endl;
16     hasil1 = bil2 / bil1; // integer division
17     cout << hasil1 << endl;
18     hasil1 = bil1 % bil2; // modulo
19     cout << hasil1 << endl;
20     hasil1 = bil2 % bil1; // modulo
21     cout << hasil1 << endl;
22     hasil2 = bil3 / bil4;
23     cout << hasil2 << endl;
24     return 0;
25 }
```

Jawab :

```
int main () {
    int bil1 = 3, bil2 = 4, hasil1;
    float bil3 = 3.0, bil4 = 4.0, hasil2;
    hasil1 = bil1 + bil2;
    cout << hasil1 << endl;
    hasil1 = bil1 - bil2;
    cout << hasil1 << endl;
    hasil1 = bil1 * bil2;
    cout << hasil1 << endl;
    hasil1 = bil1 / bil2; // integer division
    cout << hasil1 << endl;
    hasil1 = bil2 / bil1; // integer division
    cout << hasil1 << endl;
    hasil1 = bil1 % bil2; // modulo
    cout << hasil1 << endl;
    hasil1 = bil2 % bil1; // modulo
    cout << hasil1 << endl;
    hasil2 = bil3 / bil4;
    cout << hasil2 << endl;
    return 0;
}
```

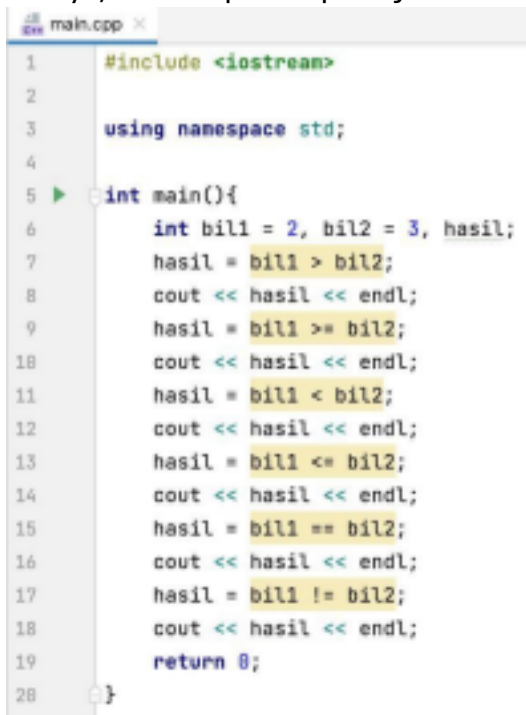
```
7
-1
12
0
1
3
1
0.75
PS C:\Users\noven\Desktop\Struktur Data Praktek>
```

Kodingan ini melakukan berbagai operasi aritmatika menggunakan variabel bertipe integer dan float. Pertama, variabel `bil1` dan `bil2` dideklarasikan sebagai bilangan bulat (3 dan 4), sementara `bil3` dan `bil4` sebagai bilangan desimal (3.0 dan 4.0). Program kemudian melakukan beberapa operasi aritmatika, mencetak hasilnya satu per satu.

1. Penjumlahan `bil1 + bil2` disimpan ke `hasil1` dan dicetak.
2. Pengurangan `bil1 - bil2` dicetak berikutnya.
3. Perkalian `bil1 * bil2` juga dicetak.
4. Operasi pembagian integer `bil1 / bil2` menghasilkan hasil pembagian bilangan bulat (integer division), di mana angka desimal dibuang.
5. Begitu pula pembagian `bil2 / bil1` mengikuti aturan integer division.
6. Operasi modulo (`bil1 % bil2` dan `bil2 % bil1`) menghitung sisa pembagian kedua bilangan bulat.
7. Terakhir, pembagian `bil3 / bil4` menggunakan tipe float dicetak dengan hasil desimal.

Program ini menunjukkan perbedaan antara operasi aritmatika dengan tipe data integer dan float, serta penggunaan modulo untuk sisa pembagian.

3. (Operasi perbandingan) Tuliskan kode berikut dan jalankan. Screenshot kode dan hasilnya, lalu tempelkan pada jawaban.



```
1  #include <iostream>
2
3  using namespace std;
4
5  int main(){
6      int bil1 = 2, bil2 = 3, hasil;
7      hasil = bil1 > bil2;
8      cout << hasil << endl;
9      hasil = bil1 >= bil2;
10     cout << hasil << endl;
11     hasil = bil1 < bil2;
12     cout << hasil << endl;
13     hasil = bil1 <= bil2;
14     cout << hasil << endl;
15     hasil = bil1 == bil2;
16     cout << hasil << endl;
17     hasil = bil1 != bil2;
18     cout << hasil << endl;
19     return 0;
20 }
```

Jawab :

```
int main () {
    int bil1 = 2, bil2 = 3, hasil;
    hasil = bil1 > bil2;
    cout << hasil << endl;
    hasil = bil1 >= bil2;
    cout << hasil << endl;
    hasil = bil1 < bil2;
    cout << hasil << endl;
    hasil = bil1 <= bil2;
    cout << hasil << endl;
    hasil = bil1 == bil2;
    cout << hasil << endl;
    hasil = bil1 != bil2;
    cout << hasil << endl;
    return 0;
}
```

```
0
0
1
1
0
1
PS C:\Users\noven\Desktop\Struktur Data Praktek>
```

Kodingan ini melakukan beberapa operasi perbandingan (comparison operations) menggunakan dua variabel integer, yaitu **bil1** (2) dan **bil2** (3). Program ini mengevaluasi hasil dari operasi perbandingan dan menyimpannya dalam variabel **hasil**, kemudian menampilkannya. Hasil dari setiap operasi perbandingan akan berupa 1 jika benar (true), atau 0 jika salah (false).

Berikut adalah operasi perbandingan yang dilakukan:

1. **bil1 > bil2**: Mengecek apakah **bil1** lebih besar dari **bil2**. Karena 2 tidak lebih besar dari 3, hasilnya adalah 0.
2. **bil1 >= bil2**: Mengecek apakah **bil1** lebih besar atau sama dengan **bil2**. Karena 2 tidak lebih besar atau sama dengan 3, hasilnya juga 0.
3. **bil1 < bil2**: Mengecek apakah **bil1** lebih kecil dari **bil2**. Karena 2 lebih kecil dari 3, hasilnya adalah 1.
4. **bil1 <= bil2**: Mengecek apakah **bil1** lebih kecil atau sama dengan **bil2**. Karena 2 lebih kecil dari 3, hasilnya 1.
5. **bil1 == bil2**: Mengecek apakah **bil1** sama dengan **bil2**. Karena 2 tidak sama dengan 3, hasilnya 0.
6. **bil1 != bil2**: Mengecek apakah **bil1** tidak sama dengan **bil2**. Karena 2 tidak sama dengan 3, hasilnya 1.

Program ini menunjukkan cara kerja operator logika perbandingan, yang sering digunakan dalam pengambilan keputusan (decision-making) di dalam program.

4. (Operasi logika) Tuliskan kode berikut dan jalankan. Screenshot kode dan hasilnya, lalu tempelkan pada jawaban.

```
main.cpp x
1  #include <iostream>
2
3  using namespace std;
4
5  int main(){
6      int bil1 = 2, bil2 = 3, hasil;
7      hasil = bil1 <= bil2 and bil1 < bil2;
8      cout << hasil << endl;
9      hasil = bil1 >= bil2 or bil1 < bil2;
10     cout << hasil << endl;
11     hasil = not(bil1 >= bil2) or bil1 < bil2;
12     cout << hasil << endl;
13     return 0;
14 }
```

Jawab :

```
int main () {
    int bil1 = 2, bil2 = 3, hasil;
    hasil = bil1 <= bil2 and bil1 < bil2;
    cout << hasil << endl;
    hasil = bil1 >= bil2 and bil1 < bil2;
    cout << hasil << endl;
    hasil = not(bil1 >= bil2) or bil1 < bil2;
    cout << hasil << endl;
    return 0;
}
```

```
1
0
1
PS C:\Users\noven\Desktop\Struktur Data Praktek>
```

Kodingan ini menggunakan operator logika **and**, **or**, dan **not** untuk mengevaluasi kondisi dari variabel **bil1** (2) dan **bil2** (3). Setiap operasi logika menghasilkan nilai **true** (1) atau **false** (0), yang kemudian disimpan dalam variabel **hasil** dan dicetak menggunakan **cout**. Berikut penjelasan operasi pada masing-masing baris:

1. **hasil = bil1 <= bil2 and bil1 < bil2;**

Operasi ini mengecek apakah **bil1** lebih kecil atau sama dengan **bil2** **dan** **bil1** lebih kecil dari **bil2**. Karena 2 memang lebih kecil atau sama dengan 3 dan juga lebih kecil dari 3, kedua kondisi ini benar, sehingga hasilnya adalah **1** (true).

2. **hasil = bil1 >= bil2 and bil1 < bil2;**

Operasi ini mengecek apakah **bil1** lebih besar atau sama dengan **bil2** **dan** **bil1** lebih kecil dari **bil2**. Kondisi pertama (**bil1 >= bil2**) salah karena 2 tidak lebih besar atau sama dengan 3, sementara kondisi kedua benar. Karena operator **and** memerlukan kedua kondisi benar untuk menghasilkan true, hasil dari operasi ini adalah **0** (false).

3. **hasil = not(bil1 >= bil2) or bil1 < bil2;**

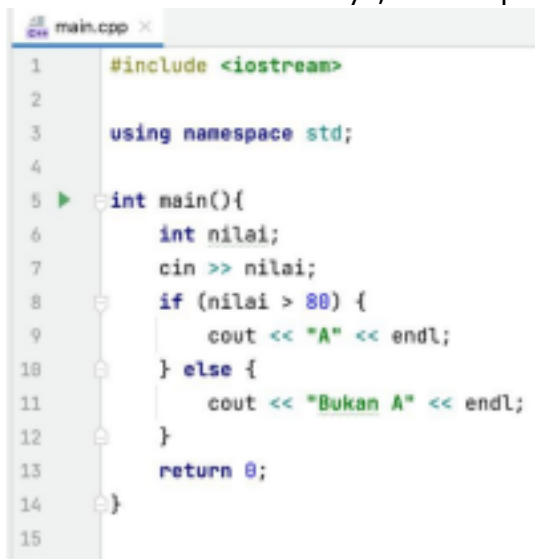
Operasi ini pertama-tama mengevaluasi **not(bil1 >= bil2)**, yang membalik hasil dari **bil1 >= bil2**. Karena **bil1 >= bil2** salah (false), maka **not(false)**

menjadi true (1). Kondisi kedua (`bil1 < bil2`) juga benar (true), sehingga dengan operator `or`, hasilnya akan **true** (1) jika salah satu atau kedua kondisi benar. Jadi hasil akhir adalah **1**.

Program ini memperlihatkan cara kerja logika gabungan menggunakan **and**, **or**, dan **not**, yang berguna dalam pemrograman untuk membuat keputusan berdasarkan kondisi yang kompleks.

Penggunaan struktur kontrol

5. (Percabangan if-else) Tuliskan kode berikut dan jalankan. Masukkan input 80, 81, dan 79. Screenshot kode dan hasilnya, lalu tempelkan pada jawaban.

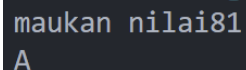


```
1  #include <iostream>
2
3  using namespace std;
4
5  int main(){
6      int nilai;
7      cin >> nilai;
8      if (nilai > 80) {
9          cout << "A" << endl;
10     } else {
11         cout << "Bukan A" << endl;
12     }
13     return 0;
14 }
15
```

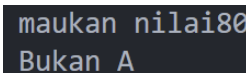
Jawab :



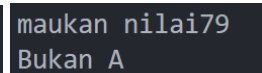
```
int main () {
    int nilai;
    cout << "masukkan nilai"; cin >> nilai;
    if (nilai > 80 ) {
        cout << "A" << endl;
    } else {
        cout << "Bukan A" << endl;
    }
    return 0;
}
```



```
masukkan nilai81
A
```



```
masukkan nilai80
Bukan A
```



```
masukkan nilai79
Bukan A
```

Kodingan ini meminta pengguna untuk memasukkan nilai, kemudian menggunakan struktur kontrol `if-else` untuk memeriksa apakah nilai tersebut lebih besar dari 80. Jika kondisinya benar (nilai lebih besar dari 80), program akan mencetak "A". Jika tidak, program akan mencetak "Bukan A". Berikut penjelasan lebih detail:

1. `int nilai;`  
Variabel `nilai` bertipe integer dideklarasikan untuk menyimpan nilai yang diinputkan oleh pengguna.
2. `cout << "masukkan nilai"; cin >> nilai;`

Program menampilkan pesan "masukkan nilai" dan menunggu input dari pengguna. Nilai yang dimasukkan disimpan dalam variabel `nilai`.

3. **`if (nilai > 80)`**

Kondisi ini mengecek apakah nilai yang dimasukkan lebih besar dari 80. Jika kondisi ini benar (nilai lebih dari 80), program akan mencetak "A".

4. **`else`**

Jika kondisi `if` salah (nilai kurang dari atau sama dengan 80), maka blok `else` dijalankan, yang mencetak "Bukan A".

5. **`return 0;`**

Program selesai dengan pengembalian nilai 0, yang menunjukkan bahwa program berjalan dengan sukses.

Jadi, program ini sederhana untuk memberikan kategori berdasarkan input nilai: jika lebih dari 80, maka mendapat "A", jika tidak, maka "Bukan A".

6. (Perulangan for-to-do) Tuliskan kode berikut dan jalankan. Masukkan 1 dan 10. Screenshot kode dan hasilnya, lalu tempelkan pada jawaban.

```
main.cpp x
1  #include <iostream>
2
3  using namespace std;
4
5  int main(){
6      int a, b, bilangan;
7      cout << "Masukan batas bawah: ";
8      cin >> a;
9      cout << "Masukan batas atas: ";
10     cin >> b;
11     for (bilangan = a; bilangan <= b; bilangan++) {
12         cout << "Bilangan " << bilangan << endl;
13     }
14     return 0;
15 }
```

Jawab :

```
int main () {
    int a, b, bilangan;
    cout << "Masukan batas bawah : ";
    cin >> a;
    cout << "Masukan batas atas : ";
    cin >> b;
    for (bilangan = a; bilangan <= b; bilangan++) {
        cout << "Bilangan " << bilangan << endl;
    }
    return 0;
}
```

```
Masukan batas bawah : 1
Masukan batas atas : 10
Bilangan 1
Bilangan 2
Bilangan 3
Bilangan 4
Bilangan 5
Bilangan 6
Bilangan 7
Bilangan 8
Bilangan 9
Bilangan 10
PS C:\Users\noven\Desktop\Struktur Data Praktek>
```

Kodingan ini meminta pengguna untuk memasukkan dua angka, yaitu batas bawah (`a`) dan



batas atas (*b*), lalu mencetak semua bilangan dari batas bawah hingga batas atas menggunakan loop **for**. Berikut penjelasan lebih rinci dari kodingan ini:

- **Deklarasi variabel:**

Tiga variabel bertipe integer dideklarasikan: *a*, *b*, dan *bilangan*. Variabel *a* akan menyimpan batas bawah, *b* akan menyimpan batas atas, dan *bilangan* akan digunakan untuk iterasi dalam loop.

- **Input batas bawah dan batas atas:**

Program meminta pengguna memasukkan dua nilai:

- a. **cin >> a;** mengambil input dari pengguna dan menyimpannya sebagai batas bawah (*a*).
- b. **cin >> b;** mengambil input dari pengguna dan menyimpannya sebagai batas atas (*b*).

- **Loop for:**

Loop ini dimulai dengan *bilangan* = *a* (batas bawah), kemudian akan berjalan selama *bilangan* kurang dari atau sama dengan *b* (batas atas). Pada setiap iterasi, nilai *bilangan* ditambahkan 1 (menggunakan *bilangan++*), dan setiap kali loop dijalankan, program akan mencetak nilai *bilangan*.

- **cout << "Bilangan " << bilangan << endl;**

Pada setiap iterasi, nilai dari *bilangan* akan dicetak. Misalnya, jika *a* adalah 2 dan *b* adalah 5

- **return 0;**

Program diakhiri dengan pengembalian nilai 0 yang menunjukkan bahwa program telah berjalan dengan sukses.

7. (Perulangan while-do) Tuliskan kode berikut dan jalankan. Masukkan pada input bilangan 10. Screenshot kode dan hasilnya, lalu tempelkan pada jawaban.

```
1  #include <iostream>
2
3  using namespace std;
4
5  int main(){
6      int bilangan, asli, jumlah;
7
8      cout << "Masukkan bilangan asli: ";
9      cin >> asli;
10
11     bilangan = 1;
12     jumlah = 0;
13     while (bilangan <= asli) {
14         if (bilangan % 2 == 0) {
15             jumlah += bilangan;
16         }
17         bilangan++;
18     }
19     cout << "Jumlah bilangan genap: " << jumlah << endl;
20     return 0;
21 }
```

Jawab :

```
int main () {
    int bilangan, asli, jumlah;

    cout << "Masukan bilangan asli : ";
    cin >> asli;

    bilangan = 1;
    jumlah = 0;
    while (bilangan <= asli) {
        if (bilangan % 2 == 0) {
            jumlah += bilangan;
        }
        bilangan++;
    }
    cout << "Jumlah bilangan genap : " << jumlah << endl;
    return 0;
}
```

```
Masukan bilangan asli : 10
Jumlah bilangan genap : 30
PS C:\Users\noven\Desktop\Struktur Data Praktek>
```

Kodingan ini meminta pengguna untuk memasukkan sebuah bilangan asli, kemudian menghitung dan menampilkan jumlah semua bilangan genap dari 1 hingga bilangan tersebut menggunakan loop **while**. Berikut penjelasan lebih rinci:

1. **Deklarasi variabel:**

Tiga variabel bertipe integer dideklarasikan:

- o **bilangan**: digunakan sebagai penghitung dalam loop.
- o **asli**: menyimpan bilangan asli yang dimasukkan oleh pengguna.
- o **jumlah**: menyimpan total jumlah bilangan genap yang dijumlahkan selama loop.

2. **Input bilangan asli:**

Program meminta pengguna untuk memasukkan sebuah bilangan asli (misalnya, 10). Nilai ini disimpan dalam variabel **asli**.

3. **Inisialisasi variabel:**

- o **bilangan = 1;**: variabel **bilangan** diinisialisasi dengan nilai 1 untuk memulai loop.
- o **jumlah = 0;**: variabel **jumlah** diinisialisasi dengan 0 karena belum ada bilangan genap yang dijumlahkan pada awalnya.

4. **Loop while:**

Loop ini berjalan selama nilai **bilangan** kurang dari atau sama dengan **asli**. Pada setiap iterasi:

- o Program mengecek apakah **bilangan** adalah genap menggunakan **if (bilangan % 2 == 0)**. Jika hasil modulus 2 dari **bilangan** adalah 0, artinya **bilangan** adalah bilangan genap.
- o Jika bilangan genap, nilai **bilangan** ditambahkan ke variabel **jumlah** menggunakan **jumlah += bilangan;**.
- o **bilangan++;** menaikkan nilai **bilangan** sebesar 1 setelah setiap iterasi, agar program dapat memproses bilangan berikutnya.

5. **Output hasil:**

Setelah loop selesai, program akan menampilkan hasil jumlah bilangan genap yang

ditemukan dari 1 hingga nilai `asli` menggunakan `cout << "Jumlah bilangan genap : " << jumlah << endl;`.