# ELEMEN BAHASA PEMROGRAMAN C++
# dan
# MEMULAI MEMBUAT PROGRAM

Setiap proses dalam program komputer pada dasarnya akan memanipulasi objek atau data yang tersimpan di dalam memori, yang tentunya perlu diberi nama. Setiap objek dapat berupa variabel (peubah), konstanta, fungsi atau prosedur. Dan setiap data yang berupa variabel atau konstanta memiliki tipe data tertentu. Tipe data menentukan nilai yang dapat dimilikinya dan operasi yang dapat dilakukannya. Tipe data dapat dikelompokkan menjadi tipe data dasar yaitu tipe data yang dapat langsung dipakai, dan tipe data bentukan yang merupakan turunan dari tipe data dasar.

Variabel, konstanta, tipe data, ekspresi atau operator, dan nilai itulah yang merupakan elemen-elemen dasar pemrograman. Sebelum membahas satu persatu elemen-elemen tersebut di atas, perhatikan terlebih dahulu contoh program C++ dalam gambar berikut ini :

```
//*********************************************************
// This is a simple C++ program. It displays four lines
// of text, including the sum of two numbers.
//*********************************************************

#include <iostream>

using namespace std;

int main()
{
    int num;

    num = 6;

    cout << "My first C++ program." << endl;
    cout << "The sum of 2 and 3 = " << 5 << endl;
    cout << "7 + 8 = " << 7 + 8 << endl;
    cout << "Num = " << num << endl;

    return 0;
}
```

**Sample Run**: (When you compile and execute this program, the following four lines are displayed on the screen.)

```
My first C++ program.
The sum of 2 and 3 = 5
7 + 8 = 15
Num = 6
```

Penjelasan program :

1. Penulisan "//" digunakan untuk memberikan keterangan program. Perintah ini tidak akan dieksekusi saat program dijalankan (run). Program di atas bertujuan untuk menjumlahkan dua bilangan.
2. Pada baris pertama terdapat perintah ***#include <iostream>*** yang digunakan untuk memanggil dan memanipulasi perintah input -output (***cin*** - ***cout***).
3. Maksud perintah berikutnya ***using namespace std;*** perintah ini digunakan untuk mendeklarasikan kepada compiler bahwa user akan menggunakan semua fungsi/class/file yang terdapat dalam paket ***namespace std*.**
4. Pada baris ke 7 terdapat perintah int main() yang merupakan judul dari fungsi utama. Pada baris ke 8 terdapat tanda buka kurawal pembuka { yang menandai dari awal dari main program (fungsi utama). Tanda kurawal penutup } terdpaat pada baris terakhir dari program, yang menandai akhir

main program. Perhatikan bahwa dalam C ++, tanda << adalah operator, yang disebut the stream insertion operator, contoh : cout << "My first C++ program." << endl; Itu adalah contoh pernyataan keluaran C ++, yang menyebabkan komputer mengevaluasi ekspresi setelah pasangan simbol << dan menampilkan hasilnya di layar.

5. Biasanya program C ++ berisi berbagai jenis ekspresi seperti aritmatika dan string. Misalnya, 7 + 8 adalah ekspresi aritmatika. Dan apapun yang berada dalam tanda kutip ganda adalah string. Misalnya, "Program C ++ pertama saya." dan "7 + 8 =" adalah string.
6. Perhatikan pernyataan output ini : cout << "Jumlah 2 dan 3 =" << 5 << endl; Pernyataan tersebut terdiri dari dua ekspresi. Ekspresi pertama adalah "Jumlah 2 dan 3 =" , dan ekspresi kedua terdiri dari angka 5. Ekspresi "Jumlah 2 dan 3 =" adalah string yang akan ditampilkan apa adanya sesuai yang tertulis. Ekspresi kedua, yang terdiri dari angka 5 dievaluasi menjadi 5. Dengan demikian, output dari pernyataan di atas adalah: Jumlah 2 dan 3 = 5.
7. Pernyataan berikutnya : cout << "7 + 8 =" << 7 + 8 << endl; Dalam pernyataan ini, ekspresi "7 + 8 =" merupakan string, yang akan menghasilkan keluaran atau output persis seperti yang tertulis. Pada pernyataan kedua, 7 + 8, ekspresi ini terdiri dari angka 7 dan 8 yang dikenali sebgai operator aritmatika oleh C ++. Oleh karena itu, hasil dari ekspresi 7 + 8 adalah jumlah dari 7 dan 8, yaitu 15. Jadi, output dari pernyataan sebelumnya adalah: 7 + 8 = 15.
8. Berikutnya : cout << "Num =" << num << endl; Pernyataan ini terdiri dari string "Num =", yang mengevaluasi dirinya sendiri, dan kata num. Pernyataan num = 6; menetapkan nilai 6 hingga num. Oleh karena itu, ekspresi num, setelah yang kedua <<, dievaluasi menjadi 6. Sekarang berarti output dari pernyataan sebelumnya adalah: Num = 6
9. Pernyataan terakhir, yaitu ***return 0***; mengembalikan nilai 0 ke sistem operasi ketika program berakhir.
10. Catatan tentang kata kunci (***keywords***) merupakan simbol kata. Beberapa simbol kata termasuk dalam keywords antara lain: int, float, double, char, const, void, return.
11. ***Reserved words*** atau kata - kata yang dicadangkan juga disebut *keywords* atau kata kunci. Huruf-huruf yang membentuk kata khusus selalu lebih kecil. Seperti simbol khusus, masing-masing dianggap sebagai simbol tunggal. Selain itu, simbol kata tidak dapat didefinisikan ulang dalam program apa pun; yaitu, mereka tidak dapat digunakan untuk apa pun selain penggunaan yang dimaksudkan. MIsal : kata "***class***" tidak dapat dipakai sebagai nama variabel dalam program C++, karena ***class*** merupakan salah satu ***keywords.***

| C++ Keywords | | | | | |
|---|---|---|---|---|---|
| *and* | *and_eq* | *asm* | *auto* | *bitand* | *bitor* |
| *bool* | *break* | *case* | *catch* | *char* | *class* |
| *compl* | *const* | *const_cast* | *continue* | *default* | *delete* |
| *do* | *double* | *dynamic_cast* | *else* | *enum* | *explicit* |
| *export* | *extern* | *false* | *float* | *for* | *friend* |
| *goto* | *if* | *inline* | *int* | *long* | *mutable* |
| *namespace* | *new* | *not* | *not_eq* | *operator* | *or* |
| *or_eq* | *private* | *protected* | *public* | *register* | *reinterpret_cast* |
| *return* | *short* | *signed* | *sizeof* | *static* | *static_cast* |
| *struct* | *switch* | *template* | *this* | *throw* | *true* |
| *try* | *typedef* | *typeid* | *typename* | *union* | *unsigned* |
| *using* | *virtual* | *void* | *volatile* | *wchar_t* | *while* |
| *xor* | *xor_eq* | | | | |

Dalam penulisan program secara umum, terdiri dari 2 bagian utama, yaitu bagian deklarasi & bagian statement. Bagian deklarasi merupakan bagian program untuk mendefinisikan atau mendeklarasikan variabel, konstanta, dan tipe datanya. Selain itu bagian deklarasi dapat juga digunakan untuk memberi nilai awal suatu variable, dengan kata lain deklarasi digunakan untuk memperkenalkan suatu nama kepada *compiler*.

Deklarasi di dalam Bahasa Pascal, dimana tipe data disebutkan setelah penamaan variabel atau konstantanya. Di bawah ini variabel yang dideklarasikan adalah r dan luas yang bertipe data integer (bilangan bulat), dan max yang merupakan konstanta bernilai tetap 100.

```
var  r, luas : real;
const max = 100;
```

Dalam Bahasa C++ yang merupakan pengembangan dari Bahasa Pascal, tipe data disebutkan sebelum nama variabelnya. Di bawah ini variabel yang dideklarasikan adalah num dan count yang bertipe data integer, dan num diberi nilai awal 6. Kemudian sum dan product yang bertipe data float (bilangan decimal atau pecahan), terakhir variabel ch yang bertipe data karakter.

```
int num, count;
num = 6;
float sum, product;
char ch;
```

Sedangkan bagian statement merupakan bagian program yang berisi rangkaian perintah sesuai notasi algoritmanya. Rangkaian perintah tersebut pada umumnya dituliskan setelah pendeklarasian variabel.

## A. Variabel dan Konstanta

Dalam dunia pemrograman, istilah variabel merujuk pada suatu tempat yang digunakan untuk menampung data di dalam memori yang mempunyai nilai yang dapat berubah – ubah selama proses program. Suatu tempat atau lokasi di dalam memori tersebut menyimpan data yang akan diolah. Bisa diasumsikan bahwa sebuah varibel yang dideklarasikan dalam program, merupakan sebuah wadah yang dipesan oleh user untuk menampung nilai atau data yang sifatnya dapat berubah-ubah atau tidak tetap selama program dijalankan. Singkatnya variabel merupakan objek yang nilainya tidak tetap, dapat berubah sesuai proses yang dikenai terhadapnya. Nama variabel harus didefinisikan tipe datanya dalam bagian deklarasi.

Contoh :
Deklarasi :

```
int nilai, nomer;
float ipk;
string nama;
char gender;
```

Sedangkan konstanta adalah objek yang nilainya tetap, tidak berubah selama program dijalankan, notasinya menggunakan ***const***. Contoh : const phi = 3.14; const kode = “xyz123”; const nilai_max = 100; dan sebagainya.

Penulisan variabel dan konstanta dalam Bahasa C++, juga dalam bahasa pemrograman lainnya, pada umumnya mempunyai aturan yang sama yaitu :

- karakter pertama harus berupa huruf
- karakter kedua dan seterusnya dapat berupa angka atau *underscore* ( _ )
- tidak mengandung spasi, operator, tanda baca, dan karakter khusus lainnya.

## B. Tipe Data

Secara umum, terdapat dua jenis tipe data yaitu tipe data dasar dan tipe data bentukan. Yang termasuk dalam tipe data dasar adalah bilangan bulat (int), bilangan riil atau pecahan (float atau real), logika, karakter, dan string, dapat dilihat pada tabel di bawah ini. Sedangkan tipe data bentukan disusun dari beberapa tipe data dasar seperti tipe terstruktur, contoh struct atau record.

| Name | Size | Range |
|---|---|---|
| Char | 1 byte | signed: -128 to 127<br>unsigned: 0 to 255 |
| short int (short) | 2 bytes | signed: -32768 to 32767<br>unsigned: 0 to 65535 |
| Int | 4 bytes | signed: -2147483648 to 2147483647<br>unsigned: 0 to 4294967295 |
| long int (long) | 4 bytes | signed: -2147483648 to 2147483647<br>unsigned: 0 to 4294967295 |
| Bool | 1 byte | true or false |
| Float | 4 bytes | +/- 3.4e +/- 38 (~7 digits) |
| Double | 8 bytes | +/- 1.7e +/- 308 (~15 digits) |
| long double | 8 bytes | +/- 1.7e +/- 308 (~15 digits |
| wchar_t | 2 or 4 bytes | 1 wide character |

### Tipe Data Dasar

Tiga tipe data dasar sederhana dalam bahasa C++ yang umum dipakai antara lain bilangan bulat, bilangan logika, dan karakter seperti dalam tabel berikut ini :

| Data Type | Values | Storage (in bytes) |
|---|---|---|
| int | -2147483648 to 2147483647 | 4 |
| bool | true and false | 1 |
| char | -128 to 127 | 1 |

Berikut macam-macam tipe dasar dan penjelasannya :

- Logika

  Tipe data logika disebut juga dengan Boolean, hanya mengenal dua nilai yaitu True (benar) dan False (salah). True dapat dinyatakan dengan angka 1, dan False dapat dinyatakan dengan angka 0, atau sebaliknya bergantung kesepakatan yang dibuat. Operator logika yang umum digunakan antara lain : ***and, or, not,*** dan ***xor.***

  Misal terdapat dua variabel X dan Y bertipe Boolean, dilakukan operasi logika ***and, or, not,*** dan ***xor*** terhadap kedua variabel tersebut. Maka cara mendeklarasikan kedua variabel tersebut adalah sebagai berikut :

Deklarasi:
***X , Y : Boolean;***

Operator ***and*** akan menghasilkan nilai benar jika variabel X dan Y **keduanya** bernilai benar, operator ***or*** akan menghasilkan nilai benar jika **salah satu** dari kedua variabel tersebut bernilai benar, dan operator ***xor*** akan bernilai benar jika X dan Y saling berlawanan nilai kebenarannya. Sedangkan ***not*** merupakan kebalikan nilai yang dikandung variabel tersebut.
Hasil operasi logika terhadap kedua variabel tersebut digambarkan dalam **tabel kebenaran** sebagai berikut :

| **X** | **Y** | **X *and* Y** | **X *or* Y** | **X *xor* Y** | ***not*** | |
|---|---|---|---|---|---|---|
| True | False | False | True | True | X = True | Not (X) = False |
| False | True | False | True | True | X = False | Not (X) = True |
| False | False | False | False | False | Y = True | Not (Y) = False |
| True | True | True | True | False | Y = False | Not (Y) = True |

Contoh operasi logika lainnya adalah jika X bernilai true, Y bernilai false, dan Z bernilai true, maka :

- (X or Y) or Z → hasilnya : (True) or True = True
- (X and Y) and Z → hasilnya : (False) and True = False
- (X or Y) and Z → hasilnya : (True) and True = True
- (X and Y) or Z → hasilnya : (False) or True = True
- X and (Y or Z) → hasilnya : True and (True) = True
- not (X and Z) → hasilnya : not (True) = False
- (Z xor Y) and Y → hasilnya : (True) and False = False

- Bilangan bulat atau integer
  Tipe data ini tidak mengandung nilai pecahan atau desimal, seperti 119, 7, -12, 620011, dan sebagainya. Tipe ini memiliki keterurutan, artinya nilai sebelumnya (predecessor) dan nilai sesudahnya (successor) dapat diketahui. Contoh predecessor dari 10 adalah 9, dan successornya adalah 11. Jika x adalah variabel bertipe integer, maka definisi keterurutan secara formalnya predecessor (a) = x – 1, dan successor (a) = x + 1.

  Deklarasi variabel bertipe data integer adalah sebagai berikut :
  ***d, e : integer ;***
- Bilangan riil atau pecahan
  Tipe data ini mengandung pecahan decimal, seperti 0.25, 5.7, 31.0, 0.0008, 1.70004300E-3, dan lain-lain. Perhatikan bilangan riil harus mengandung tanda titik (.) sebagai penanda decimal. Bilangan riil juga dapat dituliskan dengan notasi E yang berarti pangkat 10. Contoh 1.70004300E-3 berarti $1.70004300 \times 10^{-3}$.

  Deklarasi variabel bertipe data riil (real atau float) adalah sebagai berikut:
  ***g, h : real***;

- Karakter
  Semua huruf dalam alfabet, tanda baca '.' ',' '?' dan lainnya, simbol khusus seperti '&' '%' '#' '@' dan lainnya, operator aritmatik ', dan angka yang diapit tanda petik seperti '0' '12'; termasuk dalam tipe data karakter. Begitu juga karakter kosong (null) yaitu karakter yang panjangnya nol.

  Deklarasi variabel bertipe karakter (char) adalah sebagai berikut :
  ***c, d : char;***

- String
  String merupakan sekumpulan karakter dengan panjang tertentu. Pada hakikatnya string bukan tipe data dasar murni, karena disusun dari elemen-elemen bertipe karakter. Tapi tipe ini sering dipakai dalam pemrograman, yang kemudian diperlakukan sebagai tipe data dasar.

  Deklarasi variabel bertipe karakter (char) adalah sebagai berikut :
  ***nama : string;***

**Tipe Data Bentukan**

Berbeda dengan tipe data dasar yang sudah dijelaskan di atas, tipe data bentukan merupakan tipe data yang dapat dibentuk sendiri atau didefinisikan sendiri sesuai kebutuhan pemrogram (user defined data type). Tipe data bentukan dapat terdiri lebih dari satu tipe data dasar, atau terdiri dari sekumpulan variabel dengan tipe sama atau berbeda yang berupa rekaman (record). Rekaman (record) disusun oleh satu atau lebih field, tiap field menyimpan tipe dasar tertentu. Nama record ditentukan sendiri oleh pemrogram, yang selanjutnya menjadi nama tipe baru.

Tipe bentukan ini juga dikenal dengan tipe terstruktur, sesuai namanya, tipe ini menyimpan lebih dari satu variabel bertipe sama maupun berbeda. Pendeklarasian tipe terstruktur dalam C++ menggunakan kata kunci ***struct.*** Berikut ini bentuk umum deklarasi dan contohnya :

**Bentuk umum:**
```
struct nama_struktur
{
   tipe_data variabel1;
   tipe_data variabel2;
   ...
};
```

**Contoh:**
```
struct nilai_mhs
{
    char nim[10];
    float nilai_tugas, nilai_kuis, nilai_uts, nilai_uas;
};
```

Atau nama variabel dipisah dari deklarasi tipe recordnya dimana nama variabel bisa lebih dari satu seperti pada fungsi main, perhatikan bentuk umum dan contohnya berikut ini :

**Bentuk umum:**

```
typedef  struct
{
    tipefield1 namafield1;
    tipefield2 namafield2;
    … …
    tipefieldn namafieldn;
} Namatipestruct;
namatipestruct  namavar;
```

**contoh:**

```
typedef struct
{   int tanggal, bulan, tahun;
} data_lahir tgl_lhr;

typedef struct
{   char nama[25];
  data_lahir tgl_lhr;
} data_pasien;
data_pasien info_pasien;
```

Untuk mengakses atau memanggil sebuah variabel yang berada di dalam record, menggunakan bentuk umum "**namavar.namafield**" (dipisah dengan atnda titik). Perhatikan contoh dalam program sederhana berikut ini :

```
#include <iostream>
#include <string.h>

typedef struct
{ int tanggal, bulan, tahun;
} data_lahir tgl_lhr;

typedef struct
{ char nama[25];
data_lahir tgl_lhr;
} data_pasien;
data_pasien info_pasien;

int main()
{
strcpy(info_pasien.nama,"Budi");
info_pasien.tgl_lhr.tanggal = 10;
info_pasien.tgl_lhr.bulan = 2;
info_rekan.tgl_lahir.tahun = 2015;
cout << "Nama  Pasien : " <<info_pasien.nama;
cout << "\nTanggal Lahir Pasien :";
cout << "-" << info_pasien.tgl_lhr.bulan;
cout << "-" << info_pasien.tgl_lhr.tahun;
}
```

## C. Ekspresi : Operand dan Operator

Perubahan nilai di dalam proses komputer sampai menjadi keluaran dilakukan melalui suatu perhitungan atau komputasi, yang dinyatakan dalam suatu ekspresi. Ekspresi terdiri dari operand dan operator, operand dapat berupa konstanta atau variabel yang dioperasikan dengan operator tertentu. Sedangkan operator berupa simbol atau lainnya yang menyatakan proses apa yang akan dilakukan pada suatu operand.
Contoh dari operator adalah +, -, *, /, ^ (perpangkatan), div (pembagian yang menghasilkan bilangan bulat atau integer), mod (sisa hasil pembagian integer), >, <, not, and, or, xor, dan lainnya.

Terdapat tiga macam ekspresi yaitu numerik, relasional, dan string**.**

- **Ekspresi numerik :**
  Pada ekspresi numerik, tipe operand dan hasilnya bertipe numerik, dan yang perlu diperhatikan dalam penulisan ekspresi numerik adalah terdapat tingkatan (hirarki) operator. Operator yang mempunyai tingkatan lebih tinggi dikerjakan terlebih dahulu, tetapi dapat berubah karena penggunaan tanda kurung. Berikut tingkatan operator aritmetika dimulai dari yang tertinggi :
  1. ^
  2. div, mod
  3. /, *
  4. +,-

  Contoh-contoh ekspresi numerik dengan keterangan urutan pengerjaannya, baik **biner** (ekspresi dengan dua operand) maupun **uner** (ekspresi dengan satu operand) :
  - a * (b + c) → di dalam tanda kurung dikerjakan dulu
  - a ^ b * c – d → ((a ^ b) *c )) - d
  - i + j * k – 5 + l*m → I + (j*k) – 5 + (l*m)
  - (d + e) div 2
  - a mod 2
  - -a * (b+c) →**-a** adalah ekspresi uner

- **Ekspresi Relasional**
  Ekspresi relasional kadang disebut juga dengan ekspresi Boolean karena hasil ekspresinya bertipe boolean (true atau false). Ekspresi ini menggunakan operator **<, <=, >,>=,=, ==, !=** (tidak sama dengan), **not, and, or**, dan **xor**.

  Contoh ekspresi boolean, dengan deklarasi variabel-variabel bertipe boolean dan integer di bawah ini :
  *hasil, nilai = Boolean*
  *x , y = integer*

  misal hasil bernilai false, nilai bernilai true, x bernilai 4, dan y bernilai 6, maka hasil ekspresi boolean berikut ini adalah:
  *not nilai → false*
  *hasil and nilai →false*
  *nilai or hasil → true*
  *x < 10 → false*
  *nilai or (x=y) → true*

- **Ekspresi String**
  Ekspresi string adalah ekspresi menggabungkan dua buah string dengan operator "+" (operator penyambungan atau penggabungan atau *concatenation*).
  Contoh :
  *'Jl. Majapahit' + 'Sidoarjo' → Jl. Majapahit Sidoarjo*
  *'NIM' + ' 20200013' →NIM 20200013*

**D. Nilai : Input dan Output**

Nilai merupakan besaran dari tipe data yang terdefinisi (tipe dasar atau tipe bentukan). Nilai dapat berupa data yang disimpan di dalam sebuah variabel (peubah), konstanta, hasil perhitungan, atau nilai yang dikirim oleh sebuah fungsi. Pada dasarnya algoritma memanipulasi nilai yang tersimpan di dalam sebuah variabel, yaitu mengisinya ke variabel lain, memakainya untuk perhitungan, membaca nilai dari perangkat / piranti masukan, dan atau menulisnya ke piranti keluaran.

Contoh 1 :
Mengisi nilai ke variabel lain dan memakainya untuk perhitungan : misalkan num1, num2, dan num3 adalah tiga buah variabel bertipe integer, dan pernyataan berikut dieksekusi secara berurutan :

1. num1 = 18;
2. num1 = num1 + 27;
3. num2 = num1;
4. num3 = num2 / 5;
5. num3 = num3 / 4;

Tabel berikut menunjukkan nilai-nilai variabel setelah eksekusi setiap pernyataan. Tanda tanya artinya nilai yang belum diketahui - A? Menunjukkan bahwa nilainya tidak diketahui. Warna oranye di dalam kotak menunjukkan bahwa nilai variabel itu diubah.

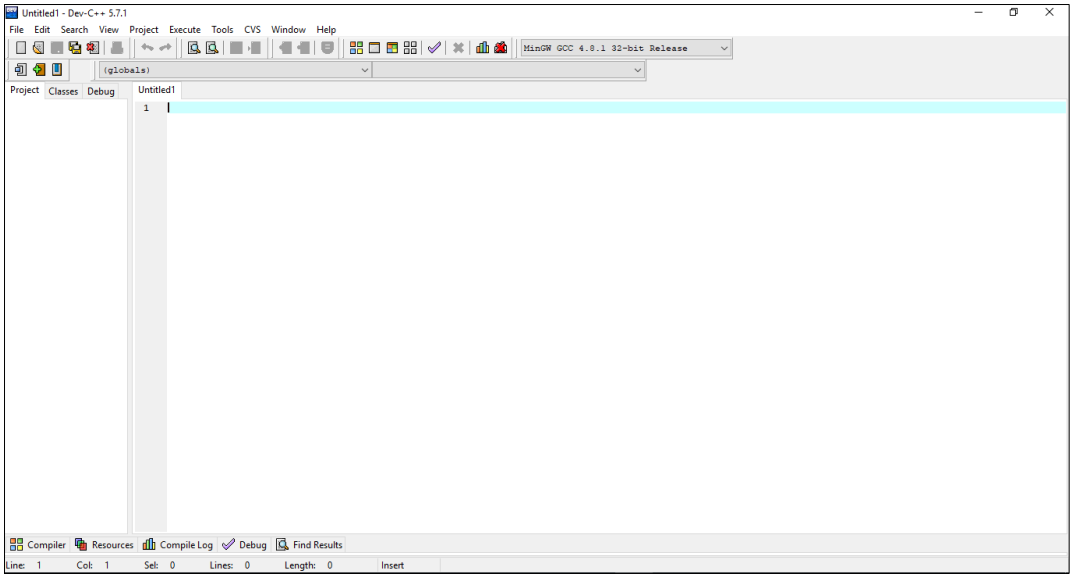
| | Values of the Variables | | | Explanation |
|---|---|---|---|---|
| Before Statement 1 | ? num1 | ? num2 | ? num3 | |
| After Statement 1 | 18 num1 | ? num2 | ? num3 | |
| After Statement 2 | 45 num1 | ? num2 | ? num3 | **num1 + 27 = 18 + 27 = 45.** This value is assigned to **num1**, which replaces the old value of **num1**. |
| After Statement 3 | 45 num1 | 45 num2 | ? num3 | Copy the value of **num1** into **num2**. |
| After Statement 4 | 45 num1 | 45 num2 | 9 num3 | **num2 / 5 = 45 / 5 = 9.** This value is assigned to **num3**. So **num3 = 9**. |
| After Statement 5 | 45 num1 | 45 num2 | 2 num3 | **num3 / 4 = 9 / 4 = 2.** This value is assigned to **num3**, which replaces the old value of **num3**. |

Contoh 2 :
Membaca nilai dari piranti masukan (keyboard) dan menulisnya ke piranti keluaran (monitor) :

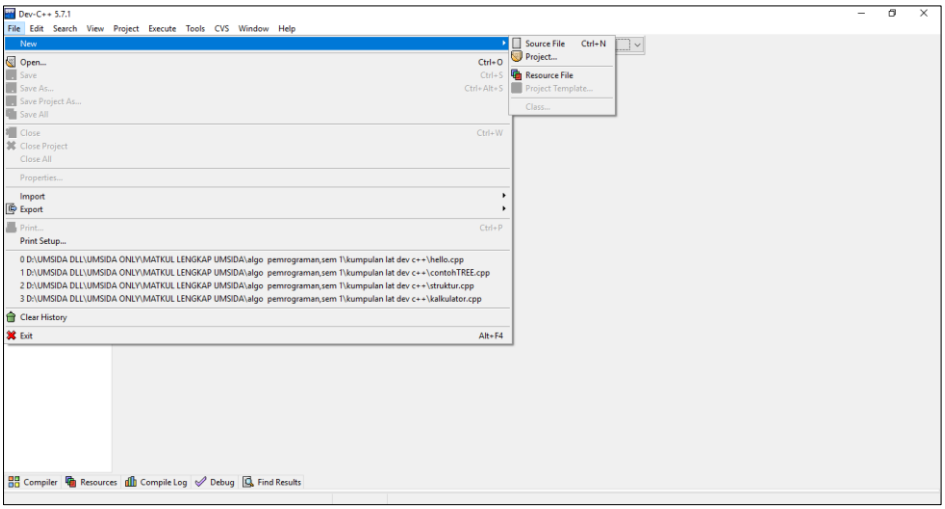
*read (x);* → meminta user untuk memasukkan nilai melalui keyboard
*write (x);* →mencetak isi atau nilai dari variabel x di atas, jika user memasukkan angka 30 maka akan tercetak angka 30 tersebut di monitor

## E. Memulai Membuat Program

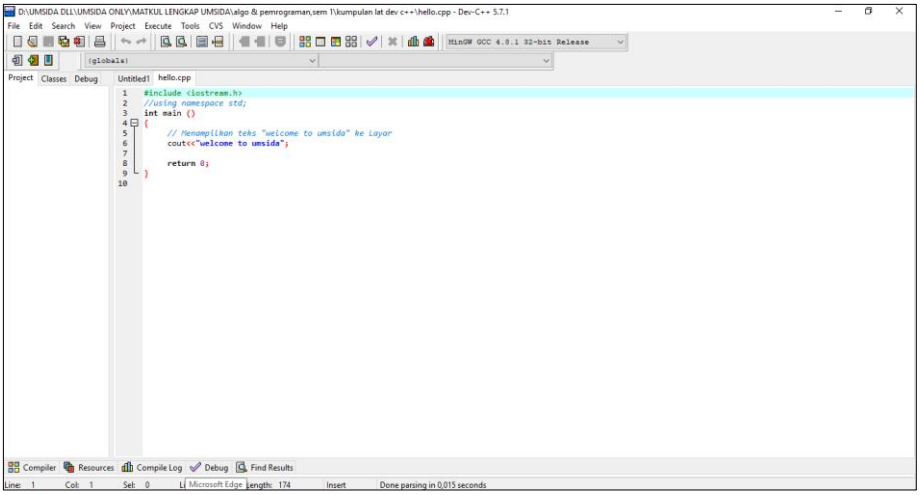
Untuk mengimplementasikan algoritma, struktur program, dan elemen-elemen pemrograman dalam bahasa C++ yang sudah dibahas di atas, diperlukan perangkat lunak compiler yang mendukung implementasi tersebut, antara lain IDE Dev C++ yang selanjutnya disingkat Dev C++ yang dibahas pada buku ini, IDE Visual Studio (Visual C++), dan aplikasi lainnya baik yang gratis maupun yang berbayar.
**Aplikasi Dev C++** merupakan aplikasi gratis untuk bahasa pemrograman C dan C++ dibawah lisensi *General Public License* (GNU). Bagi pemula aplikasi ini sangat mudah digunakan, interface yang sederhana memudahkan pengguna untuk membuat, meng-compile dan menjalankan program dalam satu aplikasi sekaligus, tanpa perlu menginstal library atau plug-in tersendiri. Karena aplikasi tersebut sudah dilengkapi dengan *DM-GCC Compiler* yang merupakan bagian dari *GNU Compiler Collection* (GCC). Aplikasi tersebut dapat diunduh secara gratis di https://sourceforge.net/projects/orwelldevcpp/. Berikut tampilan awalnya:




Selanjutnya pilih menu File – New – Source File, untuk memulai menulis program, seperti tampak dalam tampilan di bawah ini.



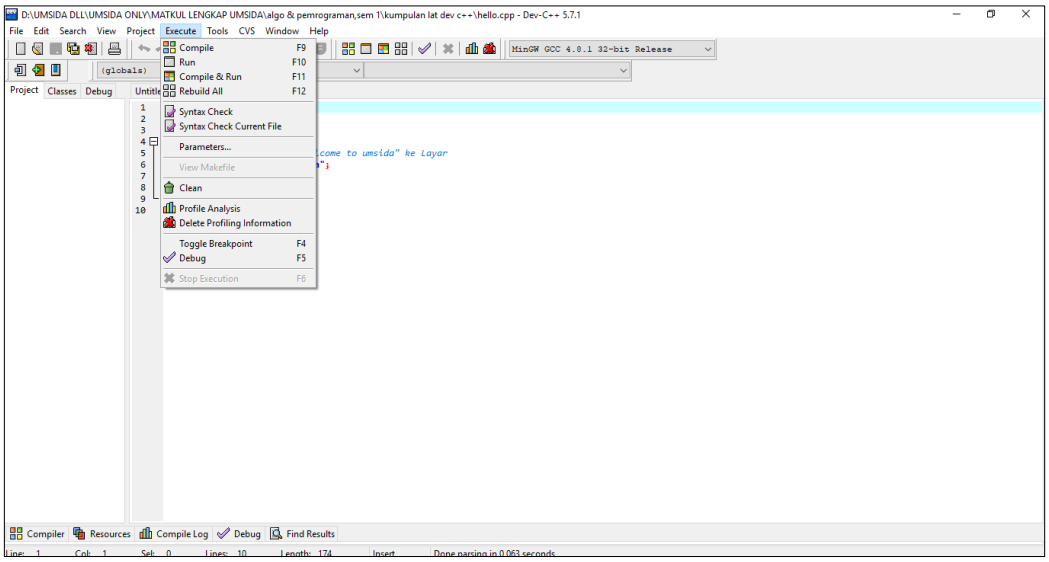
Kemudian bisa langsung menulis perintah-perintah program dalam bahasa C++ pada kolom editor, seperti tampak dalam gambar di bawah ini.



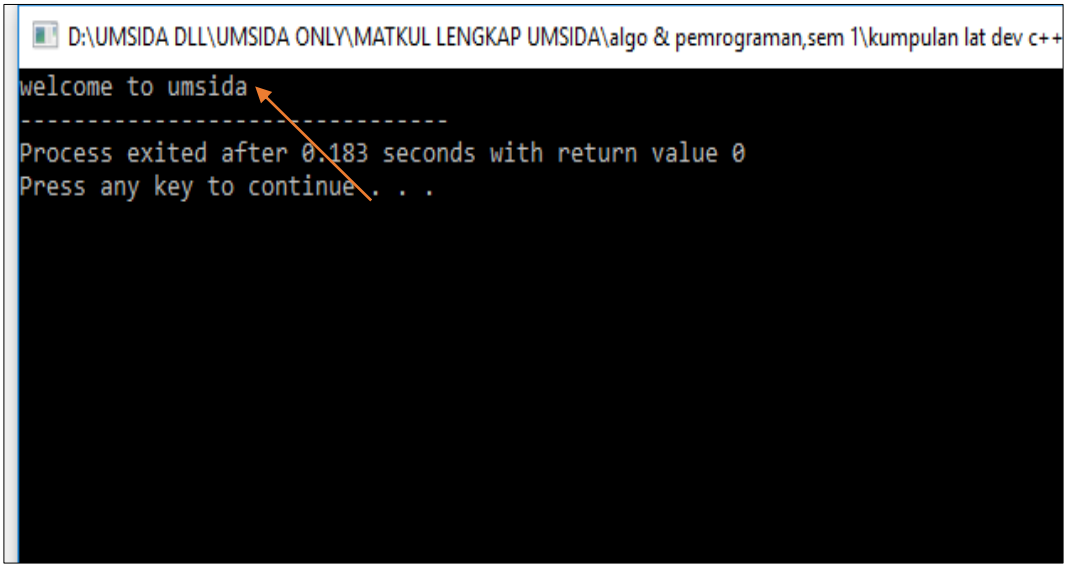
Program pada gambar di atas merupakan program sederhana C++ menampilkan teks "***welcome to umsida***" dengan susunan perintah sebagai berikut :

```
#include <iostream.h>
//using namespace std;
int main ()
{
   // Menampilkan teks "welcome to umsida" ke Layar
   cout<<"welcome to umsida";
   return 0;
}
```

Perhatikan perintah pada tiap barisnya, simak kembali pembahasan perintah program yang telah dijelaskan di atas (bagian awal bab ini).
Selanjutnya simpan dengan nama "**hello**" (ekstensi cpp akan diberikan oleh Dev C++). Selanjutnya program dapat dijalankan dengan memilih menu Compile kemudian menu Run, seperti yang tampak dalam gambar di bawah ini.




Dan berikut tampilan output programnya, perhatikan bagian yang diberi tanda anak panah.




**Sumber :**

Buku ajar ***Algoritma Pemrograman dalam Bahasa C++,*** Uce Indahyanti, UMSIDA PRESS, 2020