BOUNDLESS. OPPORTUNITY.
HORIZON EDUCATION
Shaping the Global Indonesian

ALGORITME

Modul #3 Lembar Kegiatan Mahasiswa

Nama : _____ **Tanggal :** _____

Tingkat : I (SATU), SEMESTER : 1 (SATU)

Pokok Bahasan/ Pembelajaran :
VARIABEL, KONSTATA DAN TYPE DATA

Sasaran Pembelajaran :
Di akhir modul, mahasiswa akan dapat **:**

- Membedakan jenis-jenis tipe data dasar dalam pemrograman;
- Menggunakan jenis-jenis tipe data dasar dalam pemrograman;
- Memahami penggunaan varibel dan konstanta dalam pemrograman;
- Mendeklarasikan variabel menggunakan jenis-jenis tipe data dasar.

Materi :
Alat Tulis, Modul dan Laptop

Referensi :

- Davis, S. R. (2014). C++ For Dummies (7th ed.). John Wiley & Sons, Inc.
- Deitel, P., & Deitel, H. (2014). C++ How To Program (9th ed.). United State of America: Pearson.
- Munir, R. (2005). Algoritma dan Pemrograman dalam Bahasa Pascal danC. Bandung: Penerbit Informatika.
- Sjukani, M. (2014). Algoritma dan Struktur Data 1 dengan C, C++ dan Java (Edisi 9 ed.). Jakarta: MItra Wacana Media.

A. TINJAUAN PENDAHULUAN

Pada bagian ini ini kita akan bahas tentang Variabel, Konstanta dan Tipe Data pada algoritma, ingat! pada algoritma! akan sedikit berbeda dengan yang ada pada bahasa pemrograman tapi fungsinya tetap sama.

Tipe data merupakan suatu kesatuan konsep yang paling mendasar didalam pemrograman komputer, karena tipe-tipe data dasar dapat membentuk berbagai macam ekspresi yang akan digunakan dalam program.
Tipe Data Dasar adalah himpunan nilai yang dapat dimiliki oleh sebuah data. Tipe data menentukan apakah sebuah nilai dapat dimiliki sebuah data atau tidak, serta operasi apa yang dapat dilakukan pada data tersebut. Contoh tipe data dalam dunia nyata adalah *bilangan bulat.*
Dalam sebuah program, setiap variabel dan konstanta memiiki tipe data yang harus dideklarasikan di awal program. Pendeklarasian tipe data bertujuan untuk menentukan besarnya tempat dalam memori yang akan digunakan untuk menyimpan data tersebut saat program dijalankan.
Tipe data dasar adalah tipe data yang dapat langsung digunakan. Secara umum terdapat 2 tipe data dasar, yaitu **Numerik** dan **Kategorik.** Tipe data Numerik terdiri atas angka yang dapat dioperasikan dengan perhitungan, sedangkan Kategorik berupa angka maupun huruf namun tidak dapat mengalami operasi perhitungan.

Alokasi waktu untuk mempelajari modul ini sekitar 2 x 50 menit.

Coba kalian cari informasi tentang Variabel, Konstanta dan Type Data di buku, internet, atau sumber pembelajaran lainnya.

Mari cari tahu apa yang Anda ketahui tentang Type Data dan Operator dalam Algoritme!
1. Apa itu type data dan jenis type data dalam algoritma?
2. Jenis Operator dalam Algoritma?

Tulis jawaban Anda di bawah ini dan bagikan jawaban dan komentar Anda di bawah ini!

1. _____

2. _____

B. MATERI PEMBELAJARAN

Berikut adalah rincian materi pembelajaran Anda. Bacalah dengan cermat setiap pertanyaan dan catat pertanyaan apa pun yang mungkin Anda miliki di tempat yang disediakan.

Sebuah variabel akan mengalokasikan tempat penyimpanan yang mempunyai nama dalam memori komputer. Setiap variabel akan memiliki tipe tertentu yang akan menentukan ukuran dan letak memori, rentang nilai yang dapat disimpan dan operasi yang dapat diterapkan ke variabel tersebut.

Variabel

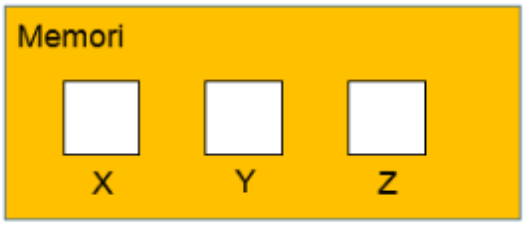
DefinisiSalah satu hal yang paling dasar dalam pemrograman adalah variabel, jika diilustrasikan sebuah variabel seperti sebuah kotak kecil, dimana kita dapat menyimpan barang-barang dalam kotak tersebut untuk digunakan nanti. Konsep variabel diambil dari matematika seperti

x = 1 → menyimpan nilai 1 dalam variabel x

Dalam pemrograman variabel adalah lokasi atau area atau tempat di dalam memori yang dapat menyimpan data sementara dalam suatu program, dan data tersebut dapat diubah, disimpan atau ditampilkan kapanpun dibutuhkan.Setiap variabel harus diberi nama, dan nama variabel harus berbeda antara satu variabel dengan variabel yang lain. Masing-masing variabel memiliki alamat sendiri didalam memori komputer, kita cukup meyebutkan nama variabel dimana data di simpan, maka komputer akan dapat menemukan alamat variabel tempat data tersebut tersimpan pada memori.

Contoh :

Misal terdapat 3 buah variabel yang di beri nama **X**, **Y** dan **Z**, jika dilustraskan didalam memori komputer digambarkan sebagai berikut :



Terlihat dari ilustrasi gambar diatas, variabel X, Ydan Zakan menempati lokasi atau area tertentu didalam memori komputer. Dimana variabel tersebut digunakan untuk menampung nilai yang akan digunakan dalam program.

Pemberian Nama Variabel

Pemberian nama variabel ditentukan oleh pembuat program sendiri, namun dalam pemberian nama variabel terdapat syarat-syarat seperti berikut :

1. Nama variabel tidak boleh sama dengan nama keyword dan function.
2. Nama variabel maksimum 32 karakter.
3. Nama variabel harus diawali dengan huruf atau garis bawah (underscore_), karakter berikutnya boleh angka, huruf atau garis bawah.
4. Nama variabel tidak boleh ada spasi.

Contoh Pemberian Nama Variabel :
Berikut adalah beberapa contoh pemberian nama variabel yang benar, dan contoh pemberian nama variabel yang salah:

Penamaan yang Benar	Penamaan yang Salah	Keterangan
X	1X	Awalnya bukan huruf atau garis bawah
X1		
Luas	Luas-1	Mengandung tanda minus (-)
LUAS	Keliling Lingkaran	Mengandung Spasi
KelilingLingkaran	Benar/Salah	Mengandung Special Karakter
Keliling_Lingkaran	Keliling -Lingkaran	Mengandung tanda minus (-)
KL		
_panjang	float	Sama dengan Keyword
FLOAT	switch	Sama dengan Keyword

NOTE Dalam bahasa C++/C penamaan variabel berbeda antara huruf besar dan huruf kecil (*case sensitif*), variabel nilai berbeda dengan NILAI berbeda denganNilai, FLOAT berbeda dengan float yang merupakan keyword.

Konstanta

Konstanta merupakan nilai numerik/angka atau karakter yang tetap. Seperti nilai PI yaitu 22/7 atau 3.14159 merupakan nilai konstanta, nilai yang tidak dapat diubah atau nilainya tetap. Untuk mendeklarasikan konstanta menggunakan keyword **const**.

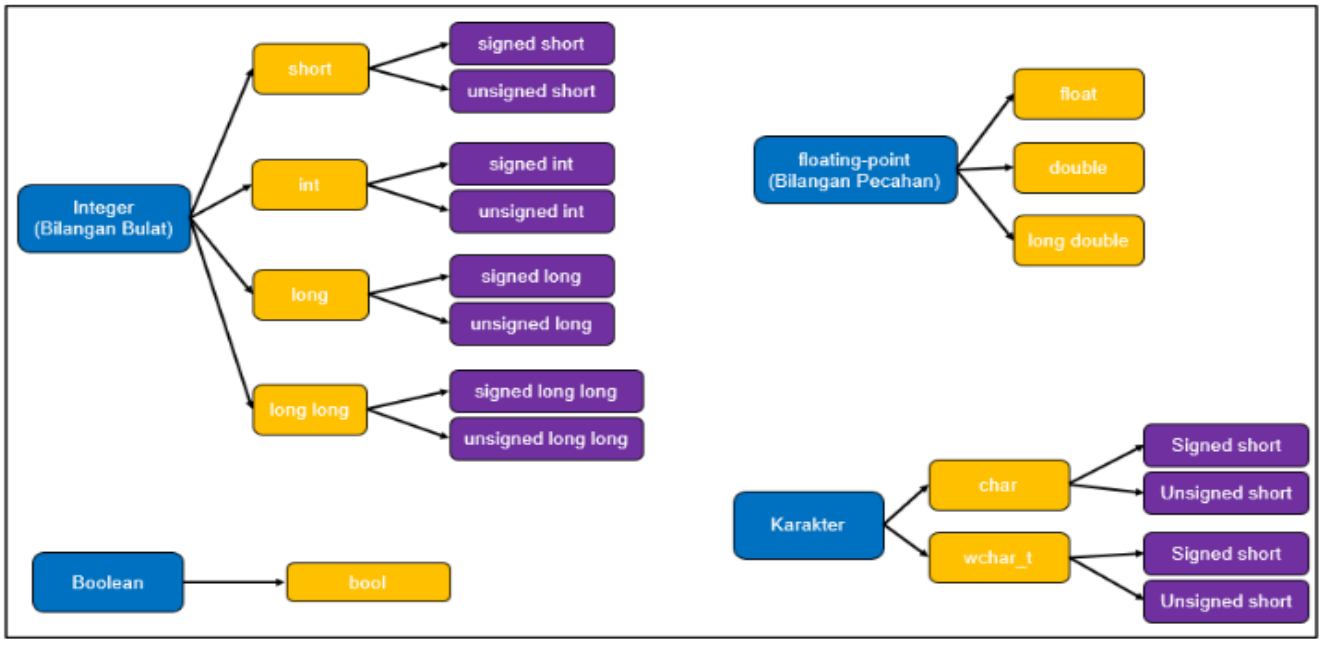
```
const double PI = 3.14159;
```

Dengan menggunakan keyword **const**, nilai dari varibel **PI** tidak bisa diubah setelah dideklarasikan.

TIPE DATA

Mengutip dari Wikipedia :
"A data type or simply type is a classification of data which tells the compiler or interpreter how the programmer intends to use the data".

Terjemahannya :
"Tipe data atau kadang disingkat dengan 'tipe' saja adalah sebuah pengelompokan data untuk memberitahu compiler atau interpreter bagaimana programmer ingin mengolah data tersebut"



Pada umumnya program komputer bekerja dengan manipulasi obyek (data) didalam memori. Obyek yang akan diprogram dapat bermacam-macam jenis atau tipenya, misalnya nilai *numerik*, *karakter*, *string* dan rekaman (*record*). Suatu tipe menyatakan pola penyajian data didalam komputer.

Sebagai contoh, misalkan saya memiliki data berupa angka. Agar bisa dipahami oleh compiler C++, data ini harus disimpan ke dalam variabel yang sudah di siapkan agar bisa menyimpan angka seperti tipe data **integer**, **float** dan **double**. Atau jika data yang harus disimpan dalam bentuk teks atau kata, bisa disimpan ke dalam tipe data **string**.

Suatu tipe diacu dari namanya. Nilai-nilai yang dicakup oleh tipe tersebut dinyatakan dalam ranah (*domain*) nilai. Operasi-operasi (beserta operator) yang dapat dilakukan terhadap tipe tersebut juga didefinisikan.

Secara garis besar, terdapat 2 kelompok tipe data dalam bahasa C++, yakni **tipe data sederhana** (***Primitive data types***), dan tipe data kompleks (***Non-primitive data types***).

Primitive Data Type, terdiri dari tipe data berikut :

1. Tipe data **Integer**

   Tipe data **integer** adalah tipe data yang dipakai untuk menampung angka bulat positif maupun negatif, seperti : 1, 45, dan -1945. Di dalam bahasa C++, terdapat beberapa sub-tipe integer yang dibedakan berdasarkan jangkauan angka yang bisa ditampung. Setidaknya terdapat 4 tipe data integer di dalam bahasa C++, seperti **char, short, int dan long.**

   Loh, bukannya **char** adalah tipe data untuk karakter? Kenapa juga ada di dalam integer? Betul, inilah keunikan tipe data **char** di dalam bahasa C / C++. Secara internal, **char** sebenarnya bertipe **integer**. Untuk membedakan apakah **char** ini berisi karakter atau huruf dilakukan pada saat menampilkannya (akan kita lihat dengan contoh praktek).

   Perbedaan dari keempat jenis tipe data diatas adalah dari segi jangkauan angka yang bisa ditampung. Tabel berikut menyajikan data yang lebih lengkap:

Jenis Tipe Data	Ukuran Memory Penyimpanan	Jangkauan
char	1 bytes	-128 sd. 127
short	2 bytes	-32,768 sd. 32,767
int	2 bytes	-32,768 sd. 32,767
long	4 bytes	-2,147,483,648 sd. 2,147,483,647

   Ukuran Memory Penyimpanan adalah jumlah memory yang dibutuhkan untuk menyimpan angka tersebut. Semakin besar jangkauan, semakin banyak juga ruang memory yang dibutuhkan. Khusus untuk tipe data **int**, bisa terdiri dari 2 byte maupun 4 byte. Ini tergantung sistem komputer dan compiler bahasa C++ yang dipakai. Jangkauan tipe data diatas adalah untuk angka yang bertanda (*signed*). Maksudnya, setiap tipe data bisa menampung angka positif dan negatif. Kita bisa mengorbankan nilai negatif ini untuk memperbesar jangkauan angka positif dengan menambahkan keyword ***unsigned*** sebelum penulisan tipe data.

   Jika tipe data integer di-set sebagai **unsigned** (tidak bertanda), maka tipe data tersebut tidak bisa menampung angka negatif, namun jatah untuk angka negatif akan dialihkan kepada angka positif sehingga jangkauannya menjadi 2 kali lipat.

Jenis Tipe Data	Ukuran Memory Penyimpanan	Jangkauan
unsigned char	1 bytes	0 hingga 255
unsigned short	2 bytes	0 hingga 65,535
unsigned int	2 bytes	0 hingga 65,535
unsigned long	4 bytes	0 hingga 4,294,967,295

Terlihat jangkauan angkanya menjadi lebih besar, tapi tidak bisa menampung angka negatif. Tipe data **unsigned** ini cocok untuk data yang tidak pernah negatif, seperti tinggi badan, berat badan, jumlah orang, dsb.

Contoh Kode Program Menggunakan Tipe Data Integer :
Cukup dengan teori, mari kita masuk ke contoh praktek. Dalam kode program berikut saya menampilkan 3 tipe data **integer** dalam bahasa C++ :

```cpp
#include <iostream>

using namespace std;

int main()
{
  short angka1;
  int   angka2;
  long  angka3;

  angka1 = 32767;
  angka2 = 2147483647;
  angka3 = 2147483647;

  cout << "Isi variabel angka1 (short): " << angka1 << endl;
  cout << "Isi variabel angka2 (int)  : " << angka2 << endl;
  cout << "Isi variabel angka3 (long) : " << angka3 << endl;

  return 0;
}
```

Di awal kode program saya membuat 3 variabel : **angka1**, **angka2**, dan **angka3**. Setiap variabel di-set dengan tipe data yang berbeda-beda, dimana variabel **angka1** sebagai **short**, variabel **angka2** sebagai **int**, dan variabel **angka3** sebagai **long**.

Setelah pendefinisian variabel, setiap variabel diisi dengan angka. Angka yang ada sengaja saya set dengan nilai maksimum untuk setiap tipe data. Terakhir setiap variabel ditampilkan menggunakan perintah **cout**. Khusus untuk **char**, mendapat perlakuan khusus karena di proses dengan sedikit berbeda. Berikut contoh kasusnya :

```cpp
#include <iostream>

using namespace std;

int main()
{
  char foo = 65;
  char bar = 'Z';

  cout << "Isi variabel foo: " << foo << endl;
  cout << "Isi variabel bar: " << bar << endl;

  printf("Isi variabel foo: %d \n",foo);
  printf("Isi variabel bar: %d \n",bar);

  return 0;
}
```

Dalam kode program ini saya membuat dua buah variabel bertipe char: **foo** dan **bar**. Untuk variabel **foo** diisi dengan angka 65, sedangkan variabel **bar** diisi dengan karakter 'Z'. Sekilas kode seperti ini terlihat aneh. Kenapa tipe data **char** bisa diisi angka dan karakter sekaligus? Jawabannya karena secara internal, tipe data char disimpan sebagai angka. Akan tetapi begitu di tampilkan dengan perintah **cout** seperti di baris 10 dan 11, tampil berbentuk karakter.

Ketika variabel **foo** diakses akan tampil huruf 'A'. Dari mana datangnya huruf 'A' ini? Huruf A tersebut berasal dari daftar kode karakter ASCII. Di dalam daftar karakter ASCII, huruf A berada di urutan ke 65. Jika kita ingin mengakses angka yang tersimpan di dalam variabel char, bisa menggunakan perintah **printf** milik bahasa C. Inilah yang saya pakai di baris 13 - 14. Hasilnya, variabel **foo** tampil sebagai angka 65, dan variabel **bar** sebagai angka 90. Angka 90 untuk karakter 'Z' ini juga bisa dilihat dari daftar karakter ASCII.

Contoh Kode Program Menggunakan Tipe Data Unsigned Integer :
Di penjelasan sebelumnya, jangkauan angka positif tipe data integer bisa diperbesar dengan menambah keyword **unsigned**. Berikut contoh penggunaannya :

```cpp
#include <iostream>

using namespace std;

int main()
{
  unsigned char  angka1;
  unsigned short angka2;
  unsigned int   angka3;
  unsigned long  angka4;

  angka1 = 255;
  angka2 = 65535;
  angka3 = 4294967295;
  angka4 = 4294967295;

  printf("Isi variabel angka1: %d \n",angka1);

  cout << "Isi variabel angka2: " << angka2 << endl;
  cout << "Isi variabel angka3: " << angka3 << endl;
  cout << "Isi variabel angka4: " << angka4 << endl;

  return 0;
}
```

Hasil kode program:

```
Isi variabel angka1: 255
Isi variabel angka2: 65535
Isi variabel angka3: 4294967295
Isi variabel angka4: 4294967295
```

Sekarang jangkauan setiap tipe data sudah membesar 2 kali lipat dengan mengorbankan angka negatif. Namun, kita tidak bisa lagi menginput angka negatif ke dalam variabel di atas. Tipe data **char** yang sebelumnya hanya bisa sampai angka 127 sekarang bisa menampung nilai 0 - 255. Begitu juga dengan tipe data unsigned **short** yang bisa menampung 0 - 65535. Serta tipe data **int** dan **long** yang sekarang sampai ke 4294967295.

2. Tipe data **Float/Double**
   Tipe data **float** dan **double** dipakai untuk menampung angka pecahan seperti 3.14, 72.12 atau -0.06463. Sama seperti bahasa pemrograman pada umumnya, kita menggunakan tanda titik sebagai pemisah angka bulat dan pecahan, bukan tanda koma seperti yang kita pakai sehari-hari di Indonesia.
   Perbedaan antara **float** dan **double** terletak dari jangkauan angka serta tingkat ketelitian. Berikut tabel perbedaan antara tipe data **float** dan **double** dalam bahasa C++:

Jenis Tipe Data	Ukuran Memory Penyimpanan	Jangkauan
float	4 bytes (32 bit)	$3.4 * 10^{-38}$ sd. $3.4 * 10^{38}$
double	8 bytes (64 bit)	$1.7 * 10^{-308}$ sd. $1.7 * 10^{308}$

Sebenarnya masih ada 1 lagi jenis tipe data untuk angka pecahan, yakni **long double** dengan jangkauan yang lebih besar dari double. Namun tidak semua compiler bahasa C++ mendukung tipe data ini. Penulisan angka pecahan juga bisa menggunakan notasi ilmiah, seperti 3.12e2 atau 4E-3. Tanda e atau E mewakili pangkat 10, sehingga $3.12e2 = 3.12 \times 10^2 = 312$. Sedangkan $4E\text{-}3 = 4 \times 10^{-3} = 0.004$.

Meskipun tipe data **float** dan **double** bisa menyimpan angka yang sangat besar, tapi tipe data ini memiliki kelemahan yang umum di setiap bahasa pemrograman (tidak hanya di bahasa C++ saja). Yakni terdapat batas tingkat ketelitian. Hal ini berhubungan dengan mekanisme penyimpanan di dalam komputer yang berbentuk angka biner.

Contoh Kode Program Tipe Data Float dan Double dalam Bahasa C++
Sebagai contoh kode program pertama, saya akan buat 2 variabel bertipe **float** dan **double**, menginput angka, lalu menampilkannya :

```
1  #include <iostream>
2
3  using namespace std;
4
5  int main()
6  {
7    float  var1;
8    double var2;
9
10   var1 = 3.957;
11   var2 = 0.019;
12
13   cout << "Isi var1 = "  << var1 << endl;
14   cout << "Isi var2 = "  << var2 << endl;
15
16   return 0;
17 }
```

Hasil kode program:

```
Isi var1 = 3.957
Isi var2 = 0.019
```

Di awal kode program, saya mendeklarasikan variabel **var1** bertipe **float**, serta variabel **var2** bertipe **double**. Kemudian di baris 10 – 11, kedua variabel ini diisi dengan angka 3.957 dan 0.019.

Nilai untuk tipe data **float** dan **double** juga bisa ditulis dalam format notasi ilmiah (*scientific notation*)

```
1  #include <iostream>
2
3  using namespace std;
4
5  int main()
6  {
7    double var1 = 4.27e5;
8    double var2 = 4.27e6;
9    double var3 = 7.99E-4;
10
11
12   cout << "Isi var1 = "  << var1 << endl;
13   cout << "Isi var2 = "  << var2 << endl;
14   cout << "Isi var3 = "  << var3 << endl;
15
16   return 0;
17 }
```

Hasil kode program:

```
Isi var1 = 427000
Isi var2 = 4.27e+06
Isi var3 = 0.000799
```

Dari percobaan ini terlihat secara default bahasa C++ menampilkan angka hingga 6 digit. Jika lebih, akan ditampilkan dengan notasi ilmiah. Inilah yang terjadi untuk **var2** yang ditampilkan kembali dengan nilai 4.27e+06.

Mengatur Format Tampilan Float dan Double C++

Dalam banyak situasi, kita ingin membatasi tampilan angka pecahan untuk digit tertentu, misalnya ingin menampilkan 2 tempat desimal atau 3 tempat desimal saja.

Terdapat berbagai solusi untuk membuat tampilan seperti ini, namun kita harus pelajari dulu bagaimana cara kerja bahasa C++ dalam menampilkan angka pecahan.

Secara bawaan (default), bahasa C++ menampilkan angka **float** atau **double** sebanyak 6 digit angka, terlepas apakah angka tersebut berada sebelum atau setelah tanda titik:

```cpp
#include <iostream>
#include <iomanip>

using namespace std;

int main()
{
  double var1 = 3.1415926;
  double var2 = 1111.1234;
  double var3 = -1000000.11;
  double var4 = 0.00123456789;
  double var5 = -99.99999;

  cout << fixed;
  cout << setprecision(2);

  cout << var1 << endl;
  cout << var2 << endl;
  cout << var3 << endl;
  cout << var4 << endl;
  cout << var5 << endl;

  return 0;
}
```

Hasil kode program:

```
3.14
1111.12
-1000000.11
0.00
-100.00
```

Pada kode program diatas saya membuat 5 buah variabel bertipe **double**, lalu menginputnya dengan angka yang berbeda-beda. Ketika ditampilkan, variabel **var1**, **var2** dan **var3** sesuai dengan aturan 6 digit tadi. Jika total lebih dari 6 digit, maka ditampilkan dengan notasi ilmiah seperti isi **var3**. Selain itu angka-angka ini juga dibulatkan agar pas 6 digit, misalnya dari 3.1415926 menjadi 3.14159. Yang agak berbeda ada di **var4** dan **var5**. Untuk **var4**, nilai yang tersimpan adalah 0.000123457. Total digit ini sudah lebih dari 6 digit, namun tetap ditampilkan utuh. Sedangkan isi **var5**, nilainya dibulatkan menjadi -100.

Jika kita ingin menampilkan setiap angka desimal dengan ketelitian 2 angka di belakang tanda titik, bisa menggunakan modifier **cout << fixed** dan **cout << setprecision(2)**. Berikut contoh penggunaannya :

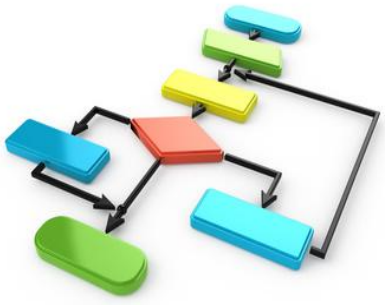
```cpp
#include <iostream>
#include <iomanip>

using namespace std;

int main()
{
  double var1 = 3.1415926;
  cout << fixed;

  cout << setprecision(0) << var1 << endl;
  cout << setprecision(1) << var1 << endl;
  cout << setprecision(2) << var1 << endl;
  cout << setprecision(3) << var1 << endl;
  cout << setprecision(4) << var1 << endl;
  cout << var1 << endl;

  return 0;
}
```

Hasil kode program:

```
3
3.1
3.14
3.142
3.1416
3.1416
```

Dalam kode program ini isi variabel **var1** ditampilkan dengan berbagai tingkat ketelitian. Khusus untuk perintah **cout << var1** di baris 17 akan memakai **setprecision(4)** milik baris sebelumnya, karena itulah pengaturan **setprecision()** paling akhir. Pengaturan ini bisa saja ditimpa jika ada perintah **setprecision()** selanjutnya.

Alternatif cara lain untuk menampilkan angka float dan double adalah dengan perintah **printf** milik bahasa C :



```cpp
#include <iostream>

using namespace std;

int main()
{
  double var1 = 3.1415926;

  printf("Isi variabel var1: %f \n",var1);
  printf("Isi variabel var1: %15f \n",var1);
  printf("Isi variabel var1: %015f \n",var1);
  printf("Isi variabel var1: %.3f \n",var1);
  printf("Isi variabel var1: %010.3f \n",var1);

  return 0;
}
```

Hasil kode program:

```
Isi variabel var1: 3.141593
Isi variabel var1:        3.141593
Isi variabel var1: 00000003.141593
Isi variabel var1: 3.142
Isi variabel var1: 000003.142
```

3. Tipe data **Boolean**
   Tipe data boolean adalah tipe data yang hanya bisa diisi salah satu dari 2 nilai: **true** atau **false**. Tipe data **boolean** banyak dipakai untuk percabangan kode program atau untuk memutuskan apa yang mesti dijalankan ketika sebuah kondisi terjadi.
   Sebagai contoh, kita bisa membuat kode program untuk menentukan apakah sebuah angka genap atau ganjil berdasarkan input dari pengguna. Untuk keperluan ini kita harus periksa terlebih dahulu apakah angka tersebut bisa dibagi 2 (untuk angka genap), atau tidak bisa dibagi 2 (untuk angka ganjil). Tipe data boolean bisa dipakai untuk menampung kondisi seperti ini, yakni benar atau salah (*True* atau *False*).
   Penggunaan tipe data boolean ini akan lebih jelas saat kita masuk ke kondisi percabangan program seperti **if-else** yang akan dibahas pada Modul terpisah.

   Contoh Kode Program Tipe Data Boolean dalam Bahasa C++
   Untuk membuat tipe data boolean, sebuah variabel harus di deklarasikan dengan keyword **bool**. Berikut contoh kode programnya :

```cpp
#include <iostream>

using namespace std;

int main()
{
  bool var1 = true;
  bool var2 = false;

  cout << "Isi var1 = "  << var1 << endl;
  cout << "Isi var2 = "  << var2 << endl;

  return 0;
}
```

Dalam kode program ini saya mendeklarasikan **var1** dan **var2** untuk menampung tipe data **boolean**, kemudian menginput nilai **true** ke **var1** dan nilai **false** ke **var2**. Pada saat ditampilkan dengan perintah **cout**, nilai boolean **true** akan tampil sebagai angka 1, sedangkan nilai boolean **false** tampil sebagai angka 0.
Penulisan nilai **true** dan **false** dalam bahasa C++ harus dengan huruf kecil semua. Compiler C++ akan menghasilkan error jika nilainya diinput dengan huruf besar atau variasi huruf lain :

```cpp
#include <iostream>

using namespace std;

int main()
{
  bool var1 = True;
  bool var2 = false;

  cout << "Isi var1 = "  << var1 << endl;
  cout << "Isi var2 = "  << var2 << endl;

  return 0;
}
```

Hasil kode program:

```
error: 'True' was not declared in this scope
```

Umumnya, tipe data boolean di dapat dari hasil **operasi perbandingan**, seperti apakah sebuah angka sama dengan angka lain, apakah lebih besar atau lebih kecil, dst. Berikut contoh yang dimaksud :

```cpp
#include <iostream>

using namespace std;

int main()
{
  bool var1, var2, var3;

  var1 = 12 < 10;
  var2 = 30 > 25;
  var3 = 'A' == 'a';

  cout << "Isi var1 = "  << var1 << endl;
  cout << "Isi var2 = "  << var2 << endl;
  cout << "Isi var3 = "  << var3 << endl;

  return 0;
}
```

Operasi-opearsi yang dapat dilakukan pada bilangan boolean dikenal dengan *operasi logika*. Operasi logika menghasilkan nilai dengan domain nilai tipe boolean (yaitu *true* dan *false*). Operator logika yang dapat digunakan untuk operasi logika adalah ***Not***, ***And***, ***Or***, dan ***Xor***.

Operasi dengan bilangan boolean akan menghasilkan nilai dalam bilangan boolean juga. Jika a dan b adalah peubah (*variable*) yang bertipe boolean, maka hasil operasi a dan b dengan operator boolean tersebut di berikan oleh tabel kebenaran berikut

A	Not A
True	False
False	True

A	B	A And B	A Or B	A Xor B
True	True	True	True	False
True	False	False	True	True
False	True	False	True	True
False	False	False	False	False

Cara mengingat operasi boolean sangat mudah. Ingatlah bahwa operasi dengan operator AND hanya akan bernilai benar jika a dan b keduanya bernilai benar. Operasi dengan operator OR hanya akan bernilai salah bila a dan b keduanya bernilai salah. Sedangkan operasi dengan operator XOR akan bernilai benar bila a dan b saling berlawanan nilai kebenarannya.

Contoh operasi boolean, misalkan X, Y dan Z adalah variabel bertipe boolean. Dimana X bernilai true, Y bernilai false dan Z bernilai true.

Operasi Logika	Hasil
(X And Y) Or Z	True
X And (Y Or Z)	True
Not (X And Z)	True
(Y Or Z) And Y	True

4. Tipe data **Char** : Tipe data untuk 1 karakter, seperti ‘a’, ‘Z’ atau ‘%’.
   Tipe data char dalam bahasa C++ digunakan untuk menampung 1 digit karakter, entah itu berupa huruf maupun angka. Variabel yang didefinisikan untuk menampung tipe data char butuh 1 byte memory. Secara teknis, char ini dikodekan dari charset **ASCII**.

   Contoh Kode Program Tipe Data Char dalam Bahasa C++
   Contoh penggunaan tipe data **char** sudah pernah kita coba beberapa kali dari tutorial-tutorial sebelumnya. Berikut contoh lain dari tipe data char dalam bahasa C++:

```cpp
#include <iostream>

using namespace std;

int main()
{
  char huruf;
  huruf = 'D';
  cout << "Huruf yang tersimpan adalah: " << huruf << endl;

  return 0;
}
```

Dalam kode ini saya mendefinisikan variabel **huruf** dengan tipe **char**. Variabel huruf kemudian diisi dengan karakter 'D' dan ditampilkan dengan perintah **cout**.

Untuk contoh kedua, saya akan buat konstanta dengan tipe data **char** :

```cpp
#include <iostream>

using namespace std;

int main()
{
  const char huruf = 'a';
  cout << "Huruf yang tersimpan adalah: " << huruf << endl;

  return 0;
}
```

Hasil kode program :
Huruf yang tersimpan adalah : a

5. Tipe data **Void** : Tipe data khusus yang menyatakan tidak ada data.

Non-Primitive Data Type, di antaranya :

1. Tipe Data **String** : Tipe data untuk kumpulan karakter, seperti "Andi", "Duniailkom", atau "Belajar C++".
2. TipeData **Array** : Tipe data untuk kumpulan tipe data lain yang sejenis.
3. Tipe Data **Structur (Struct)** : Tipe data yang terdiri dari kumpulan tipe data dasar. Tipe data tersebut bisa lebih dari 1 jenis.
4. Tipe Data **Enum** : Tipe data bentukan yang dibuat sendiri oleh kita (programmer)
5. Tipe Data **Pointer** : Tipe data untuk mengakses alamat memory secara langsung.

☞ ***Jika ada materi yang belum dipahami atau ada pertanyaan, silahkan bertanya kepada dosen.***

☞ **https://www.youtube.com/watch?v=zbSzPA9xTLo**

Berapa banyak yang Anda pelajari tentang Variabel, Konstanta dan Type Data dalam Algoritma? Ayo berlatih dengan latihan ini.

Tipe data merupakan suatu kesatuan konsep yang paling mendasar didalam pemrograman komputer, karena tipe-tipe data dasar dapat membentuk berbagai macam ekspresi yang akan digunakan dalam program.

1. Sebutkan jenis-jenis tipe data dasar dalam bahasa pemrograman c++ ?
2. Berikan tipe data yang tepat dari nilai-nilai berikut :
   a. 2000
   b. 1.5
   c. .-15000000151
   d. 9999999993.
3. Berapahnilai minimum dan maksimum yang dapat di tampung dalam tipe-tipe data berikut :
   a. short
   b. int
   c. long double
   d. char

Jawab latihan singkat yang telah dilampirkan pada kuliah hari ini sesuai dengan disiplin ilmu kita. Jika anda memiliki pertanyaan atau komentar tentang latihan ini, anda dapat membagikannya di bawah ini.

☞ ***Periksa jawaban Anda menggunakan kunci jawaban.***

C. PENUTUP PEMBELAJARAN

Anda telah mencapai bagian terakhir dari kuliah hari ini. Sekedar ulasan, baca ringkasan singkat tentang Variabel, Konstanta dan Type Data dalam Algoritma di bawah ini.

IDE KUNCI

Pada umumnya program komputer bekerja dengan manipulasi obyek (data) didalam memori. Obyek yang akan diprogram dapat bermacam-macam jenis atau tipenya, misalnya nilai *numerik*, *karakter*, *string* dan rekaman (*record*). Suatu tipe menyatakan pola penyajian data didalam komputer.

Sebagian besar tipe data yang ada di dalam bahasa C++ diturunkan dari bahasa C, oleh karena itu kita akan melihat banyak persamaan dari tipe data ini. Beberapa perbedaan seperti tipe data **boolean** yang sebelumnya tidak ada di dalam bahasa C, serta tipe data **string** yang sekarang sudah menjadi tipe data utama. Di dalam bahasa C, string pada dasarnya adalah array dari tipe data char.

Untuk lebih Variabel, Konstanta dan Type Data, silakan anda baca dan pelajari materi di buku-buku refrensi atau sumber lain.

Berpikir tentang Pembelajaran

Pengalaman Selama Pembelajaran :

Hal-hal yang belum anda pahami?

Kunci Jawaban

Silakan periksa jawaban yang anda kerjakan pada soal latihan MODUL 3 dengan kunci jawaban di bawah ini!

Kunci Jawaban :