

Struktur Dasar Program Prosedural

dan

Disain Bagan Sederhana



Subtopik

- Input – Proses – Output dalam program
- Mendisain flowchart pada suatu algoritma
- Deklarasi dan penggunaan variabel, type (dasar dan bentukan), konstanta, ekspresi (aritmatika, relasional, dan logika)
- Input/output
- Sekuens
- Contoh kasus



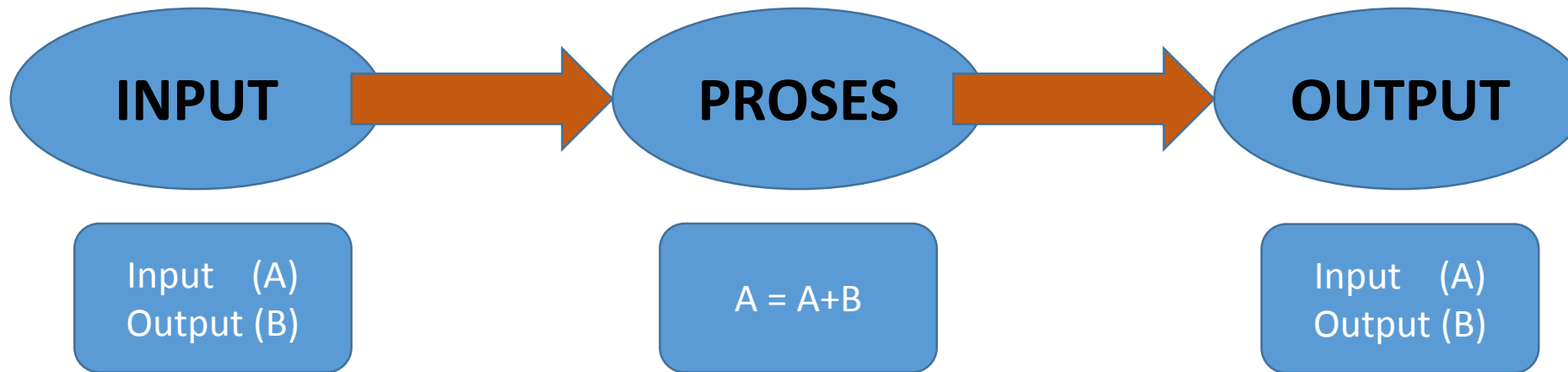
Outcomes

- Mahasiswa mampu mendisain flowchart untuk menggambarkan alur proses algoritma dalam penyesuaian suatu masalah
- Memahami makna dan penggunaan variable, type, konstanta, input/output, dan sekuens.
- Memahami persoalan yang dapat dikonversi menjadi program sederhana dengan memanfaatkan variable, type, konstanta, input/output, dan sekuens.



C++

- C++ merupakan bahasa pemrograman general purpose dan multi paradigma (prosedural, *object oriented*)
- Bahasa pemrograman yang sangat populer dan banyak digunakan
- Dikembangkan oleh Bjarne Stroustrup mulai tahun 1979 di Bell Merupakan pengembangan dari Bahasa C (procedural murni) dengan penambahan konsep, object-orientation
- Dalam kuliah ini, hanya akan menggunakan paradigm procedural
- Merupakan bahasa yang case sensitive perbedaan huruf besar dan kecil berpengaruh



C++

```
cin >> A;  
cin >> B;
```

```
A = A+B;
```

```
cout << A;  
cout << B;
```



```
#include <iostream>
using namespace std;

int main () {
    //KAMUS
    int A;
    int B;

    //ALGORITMA
    cin >> A;
    cin >> B;

    A = A + B;

    cout << A << endl;
    cout << B << endl;
    return 0;
}
```



```
// Program Test  
// Contoh struktur program prosedural dalam C++
```

```
#include <iostream>  
using namespace std;
```

```
int main () {  
    //KAMUS  
    int A;  
    int B;
```

```
    //ALGORITMA  
    A = 10;  
    B = 5;
```

```
    A = A + B;  
    B = B - A;
```

```
    cout << A << endl;  
    cout << B << endl;  
    return 0;  
}
```

Judul Program + spesifikasi, dituliskan dalam komentar

Bagian ini perlu di tambahkan sebagai standard pemrograman C++ di layar Console

KAMUS

ALGORITMA



- `iostream` adalah salah satu header file yang ada di C++. *Header* ini digunakan untuk fungsi input dan output yang ada di C++
- `Using namespace std` adalah perintah yang digunakan untuk mendeklarasikan / memberitahukan kepada *compiler* C++ bahwa kita akan menggunakan semua fungsi/class/file yang terdapat dalam *namespace std*



KAMUS, Tipe Data, Variabel, Konstanta

C++
C++
C++
C++

Kamus

- Kamus dipakai untuk mendeklarasi nama-nama yang digunakan dalam program
- Deklarasi nama yang didefinisikan pemrogram
 - type
 - variabel
 - konstanta
- Deklarasi BUKAN instruksi
- Contoh deklarasi [variabel]:

PASCAL		C++
I	: integer;	int;
JlhUang	: real;	floatjumlahuang
Titik	: Point;	Point Titik;



JENIS TIPE DATA



- ❑ Setiap data memiliki jenis yang berbeda-beda
 - – Data **umur** seseorang berbeda dengan data **nama**
 - Data Umur dibentuk dari kumpulan angka
 - Data nama dibentuk dari serangkaian huruf
 - – Untuk setiap jenis data juga memiliki rentang (range) yang berbeda
 - Data umur rentangnya antara 1 sampai 100 (bila diasumsikan bahwa umur seseorang tidak lebih dari 100).
 - Data nama rentangnya mulai dari 1 sampai 50 (bila di anggap nama tidak ada yang melebihi 50 huruf



Jenis Tipe Tipe Data

- Tipe data **primitif** atau tipe **dasar** (dalam C++)
 - Boolean (bool)
 - Integer (int)
 - Real (float)
 - Character (char)
 - String (string)
- Tipe data **turunan** atau **bentukan**
 - Dibentuk dari gabungan tipe dasar
 - Contoh
 - Tipe Data Mahasiswa
 - Dibentuk dari
 - » NIM: string
 - » Nama: string
 - » Umur: integer
 - » Kota: string
 - Tipe Array
 - Dibentuk dari kumpulan integer, misalnya 10 data tentang umur



Contoh Tipe Data

Umur	→ Integer contoh: 25, 44, 35
Kota	→ String, contoh: "Jakarta", "Bandung"
Nama	→ String, contoh: "Budi", "Ali"
Suhu	→ Integer atau float, contoh: 37.5 , 100
Luas	→ Integer atau float, contoh: 400, 43.5
BeratBadan	→ Integer atau float, contoh: 60.5, 75
NIM	→ Integer atau string?, contoh: 15812001



Contoh deklarasi tipe bentukan/komposit/struct

```
// Kamus
typedef struct {
    int x;
    int y;
} Point;
typedef struct {
    string NIM;
    string Nama;
    int Umur;
    string Kota;
} DataMahasiswa;
```



Variabel

- Variabel menyimpan nilai ber-"tipe data" sesuai dengan deklarasi
- Variabel :
 - deklarasi (supaya nama dikenal),
 - inisialisasi nilai (siap dimanipulasi)
- Contoh
 - Deklarasi variabel
 - `int i;`
 - `float A;`
 - Inisialisasi variabel
 - `i = 100;`
 - Artinya variabel i di isi dengan nilai 100
 - `A = 8.25;`
 - Artinya variabel A diisi dengan nilai real 8.25
- Operasi terhadap variabel sangat tergantung dari tipe datanya.



Operasi pada nilai suatu tipe data

- Operasi perhitungan akan memerlukan operator seperti “+”, “-”, “*” **dan** “/” (tambah, kurang, kali dan bagi) untuk melakukan kalkulasi
- Operasi “+” pada tipe data bukan numerik memiliki arti yang berbeda
 - Contoh: “Halo “ + “Apa kabar “ → “Halo Apa kabar”
- Tidak semua operator dapat digunakan untuk tipe data numerik.
 - Contoh: “Halo “ * “Apa kabar “ →



Operasi tipe dasar

□ Int : * / + - % < > <= >= == !=

□ Bool : && || ! !=

□ Float : * / + - < > <= >= == !=

□ Char : == !=



Membuat Nama Variabel yang benar

- ❖ Nama Variabel harus dimulai dengan huruf dan dapat diikuti dengan huruf lagi dan angka
 - tidak boleh ada tanda baca
- ❖ Dalam nama variable tidak boleh dipisahkan oleh spasi
- ❖ Cari nama variable yang bias/mudah dimengerti
 - agar tidak membingungkan
- ❖ C++ adalah bahasa yang **case sensitive**
 - Kesalahan penulisan huruf besar dan kecil menyebabkan error



Contoh yang benar

Volume

Luas

P

Benar atau salah ...?

Contoh yang salah

2Jari

Jumlah,total

7

BNI46
Fast2furious
+
SuperPower
abc123yes
xxxxxxx



Konstanta

- Berbeda dengan Variable, suatu konstanta **tidak boleh diubah** nilainya
- Contoh

const float PI = 3.1415

const int nilai = 1000

- Pemakaian yang salah

PI = 44.5

nilai = 5000

Keduanya salah karena PI dan nilai sudah ditandai sebagai konstanta dengan nilai 3.14159 dan 1000 jadi nilainya tidak boleh diubah



ALGORITMA

- Adalah bagian program dalam bentuk teks algoritmik yang berisi instruksi atau pemanggilan aksi
- Teks algoritmik tsb. dapat berupa:
 - Perintah dasar: Input/Output, assignment
 - Perintah perintah yang berurutan
 - Analisis kasus (jika-maka)
 - Pengulangan



Perintah-perintah dasar

- Pemberian nilai (assignment) sesuai dengan type ke suatu variabel
- Perbandingan (kesamaan, ketidak-samaan)
- Operasi relasional lain (lebih besar, lebih kecil,...)
- Operasi aritmetika (khusus untuk nilai numerik)



Nilai, Input+Output

- Nilai atau harga: suatu besaran bertipe yang telah dikenal
- Pengisian nilai:
 - Pemberian nilai langsung atau disebut sebagai ***assignment***
 - Contoh: `A = 10;`
 - Dibaca dari piranti masukan
 - Contoh: `cin >> A;`



Assignment (=)

- Ruas kiri = Ruas Kanan ;
- Ruas kiri harus variable
- Ruas kanan harus <ekspresi>
- Ekspresi :
 - “rumus perhitungan”
 - Contoh:

Luas = panjang * lebar ;

Ekspresi



Ekspresi

- Ekspresi Aritmatika

$A + B$

$x + 2 * y$

$P - 2 * Q + R/S$

- Ekspresi Relasional (pembandingan)

$A < B$

$X == Y$

Total \geq nilai

- Ekspresi Logika

$A \&\& B$

$C || B$



Komentar

- Dalam bahasa pemrograman komentar adalah bagian program yang tidak dieksekusi
 - Bagian ini hanya digunakan untuk memberikan penjelasan suatu langkah, rumus ataupun bisa hanya berupa keterangan
- Dalam C++, komentar dituliskan sebagai:
 - Antara `/*` dan `*/`
`/* ini komentar */`
 - Diawali dengan `//`
`// ini komentar`



Definisi Aksi Sekuensial

- Aksi sekuensial
 - sederetan instruksi primitif dan/atau aksi yang akan dilaksanakan (dieksekusi) oleh komputer berdasarkan urutan penulisannya
 - Setiap aksi akan mengubah status dari program
 - Jadi setiap aksi sekuensial harus ada awal dan akhir.
 - atau dengan kata lain suatu program harus dimulai dan suatu ketika harus berakhir
 - Program yang tidak pernah berhenti adalah program yang salah atau error



Penulisan untuk iinstruksi sekuensial

- ☐ Instruksi ditulis terurut sesuai penulisan perbaris
- ☐ Setiap instruksi selalu diakhiri dengan tanda titik koma
 - Di dalam satu baris dapat terdiri lebih dari satu instruksi



Contoh aksi Sekuensial

<pre>/* contoh aksi sekuensial per baris */ int main() { /* Kamus */ int i; float x; /* Algoritma */ cin >> i; x = 100.75; cout << x << endl; cout << i * 2 << endl; return 0; }</pre>	<pre>/* contoh aksi sekuensial dg titik koma */ int main() { /* Kamus */ int i; float x; /* Algoritma */ cin >> i ; x = 100.75; cout << x << endl; cout << i * 2 << endl; return 0; }</pre>
--	---



Contoh aksi Sekuensial

<pre>/* contoh aksi sekuensial per baris */ int main() { /* Kamus */ int i; float x; /* Algoritma */ cin >> i; x = 100.75; cout << x << endl; cout << i * 2 << endl; return 0; }</pre>	<pre>/* contoh aksi sekuensial dg titik koma */ int main() { /* Kamus */ int i; float x; /* Algoritma */ cin >> i ; x = 100.75; cout << x << endl; cout << i * 2 << endl; return 0; }</pre>
--	---

Perhatikan, keduanya memiliki urutan eksekusi yang sama dan juga hasil yang identik. Perbedaannya hanya pada cara penulisannya.

Mana yang lebih baik penulisannya??



Pengubahan urutan sekuensi yang **tidak merubah** hasil eksekusi

```
/* contoh aksi sekuensial per  
baris */
```

```
int main()
```

```
{
```

```
    /* Kamus */
```

```
    int i;
```

```
    float x;
```

```
    /* Algoritma */
```

```
    cin >> i;
```

```
    x = 100.75;
```

```
    cout << x << endl;
```

```
    cout << i * 2 << endl;
```

```
    return 0;
```

```
}
```

```
/* contoh aksi sekuensial per  
baris */
```

```
int main()
```

```
{
```

```
    /* Kamus */
```

```
    float x;
```

```
    int i;
```

```
    /* Algoritma */
```

```
    x = 100.75;
```

```
    cin >> i;
```

```
    cout << x << endl;
```

```
    cout << i * 2 << endl;
```

```
    return 0;
```

```
}
```




Pengubahan urutan sekuensi yang **merubah** hasil eksekusi

```
/* contoh aksi sekuensial per  
baris */
```

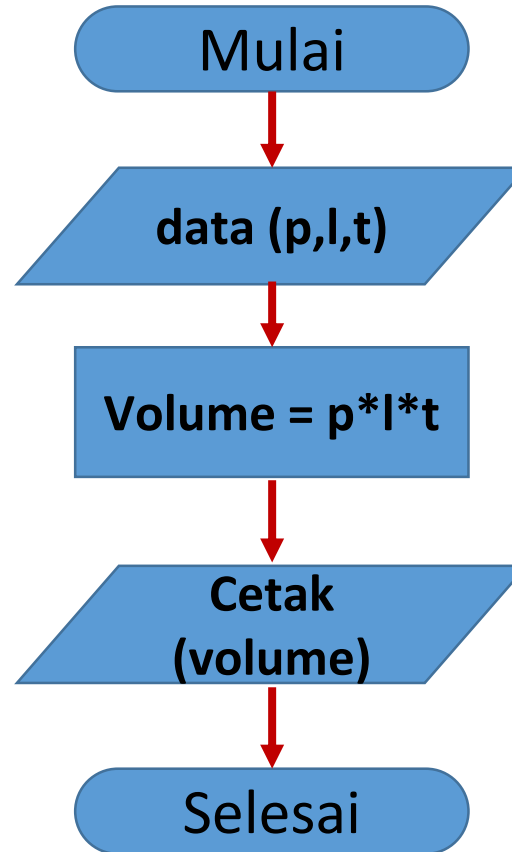
```
int main()  
{  
    /* Kamus */  
    int i;  
    float x;  
  
    /* Algoritma */  
  
    cin >> i;  
    x = 100.75;  
  
    cout << x << endl;  
    cout << i * 2 << endl;  
    return 0;  
}
```

```
/* contoh aksi sekuensial per  
baris */
```

```
int main()  
{  
    /* Kamus */  
    float x;  
    int i;  
  
    /* Algoritma */  
  
    x = 100.75;  
    cin >> i;  
  
    cout << i * 2 << endl;  
    cout << x << endl;  
    return 0;  
}
```

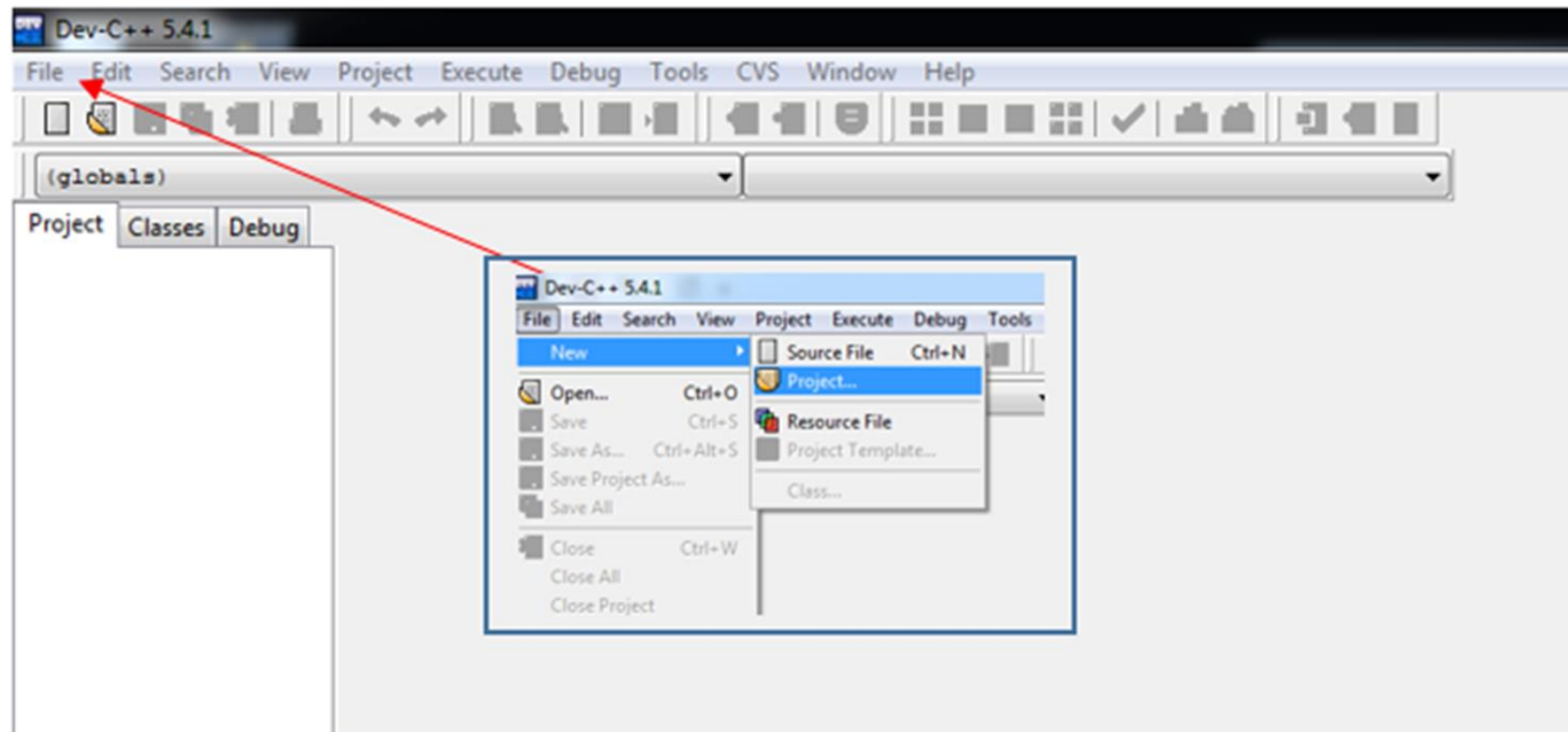


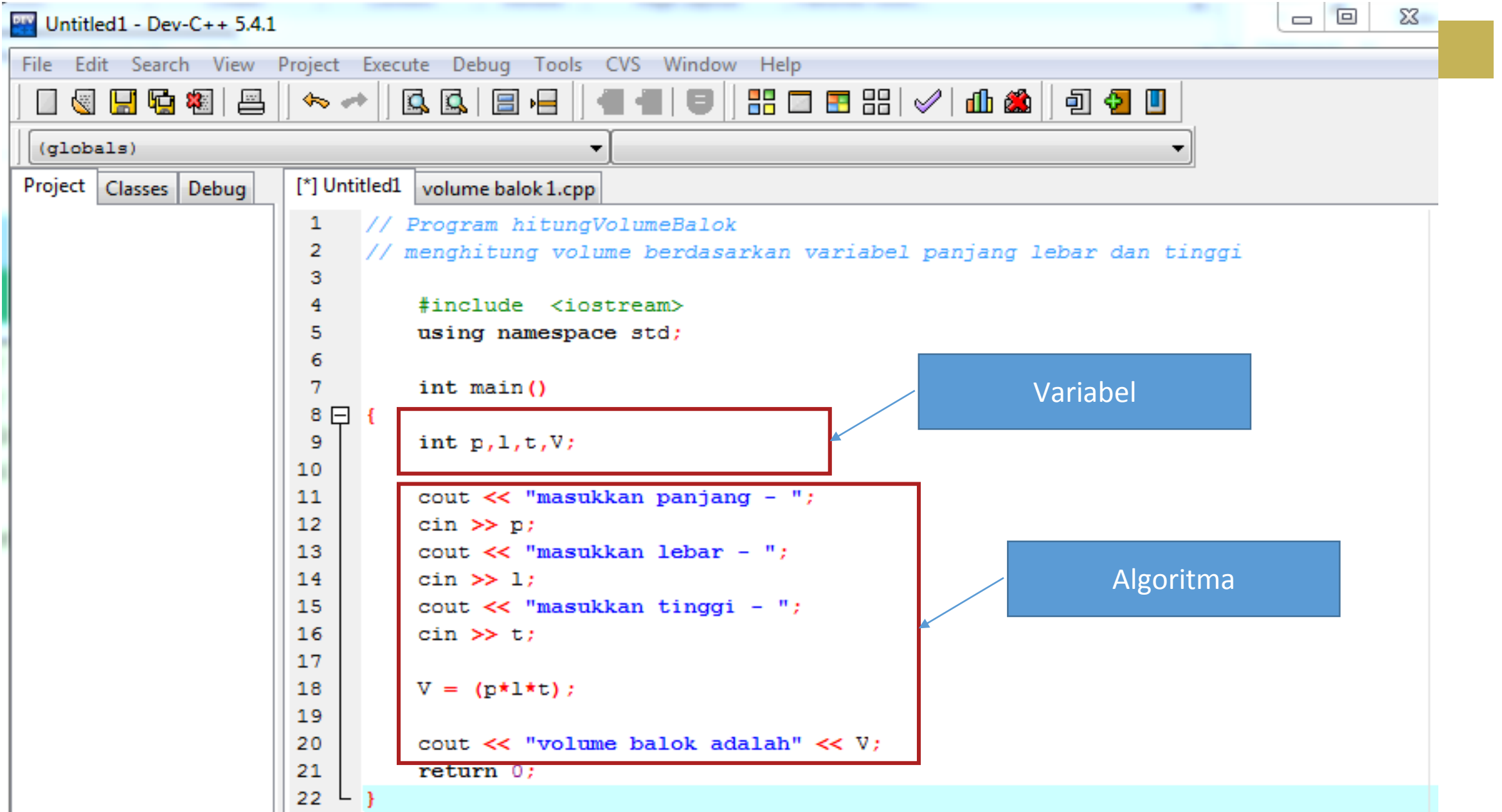
Flowchart Menghitung volume persegi panjang

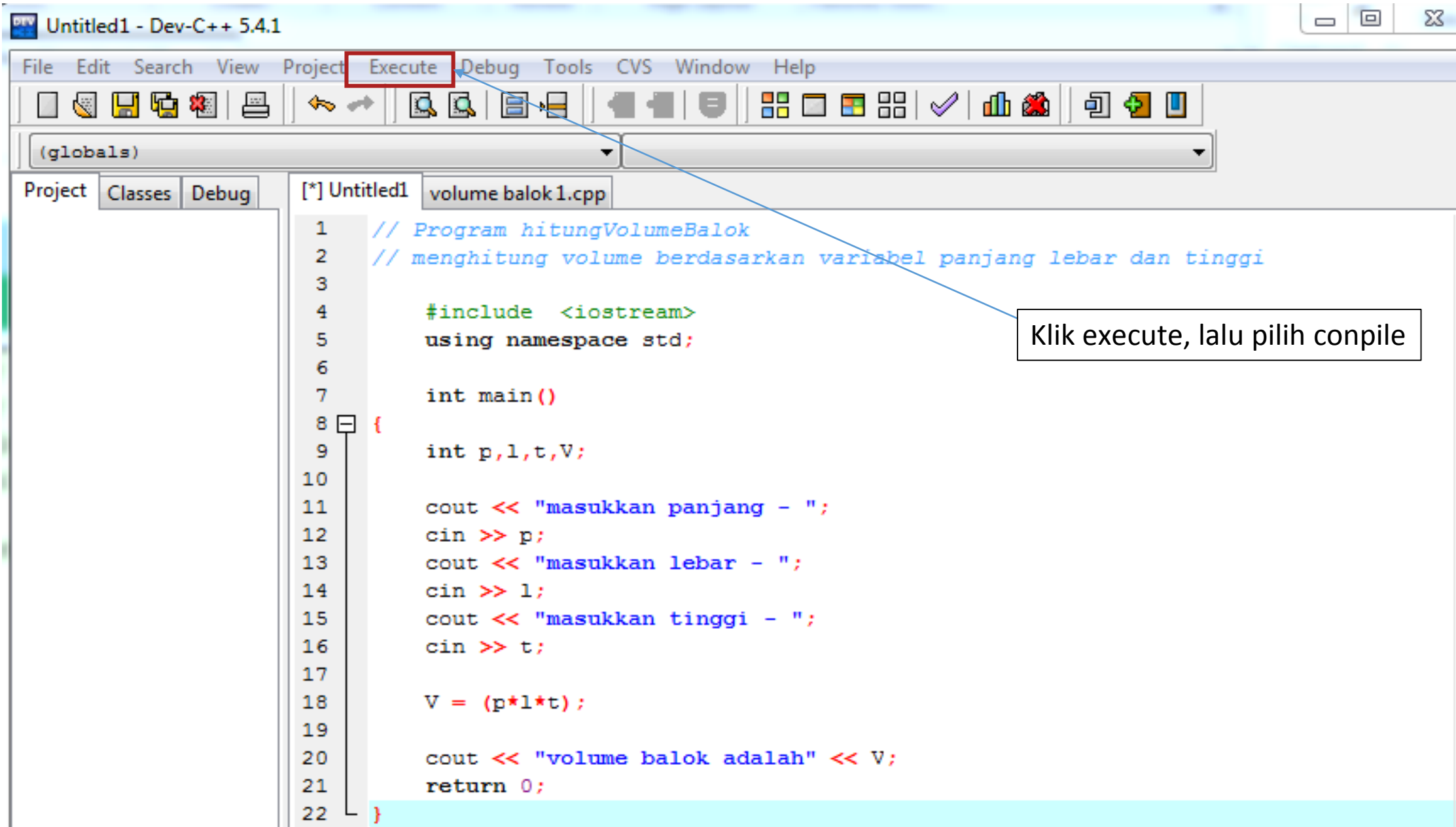


Mengoperasikan C++ dengan DevC++

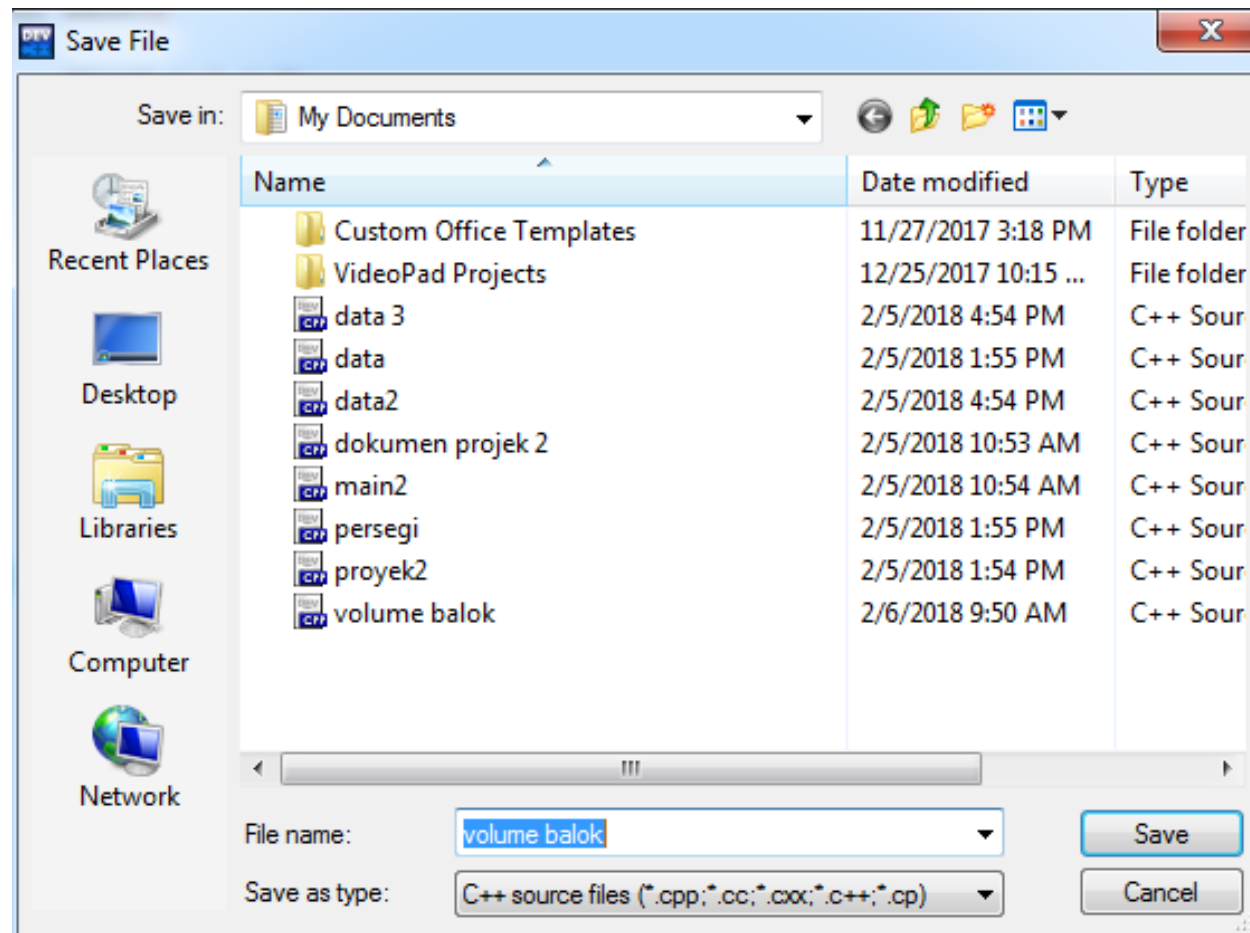
- Buka aplikasi Dev C++ dengan mengklik icon
- Klik file kemudian pilih new > source File

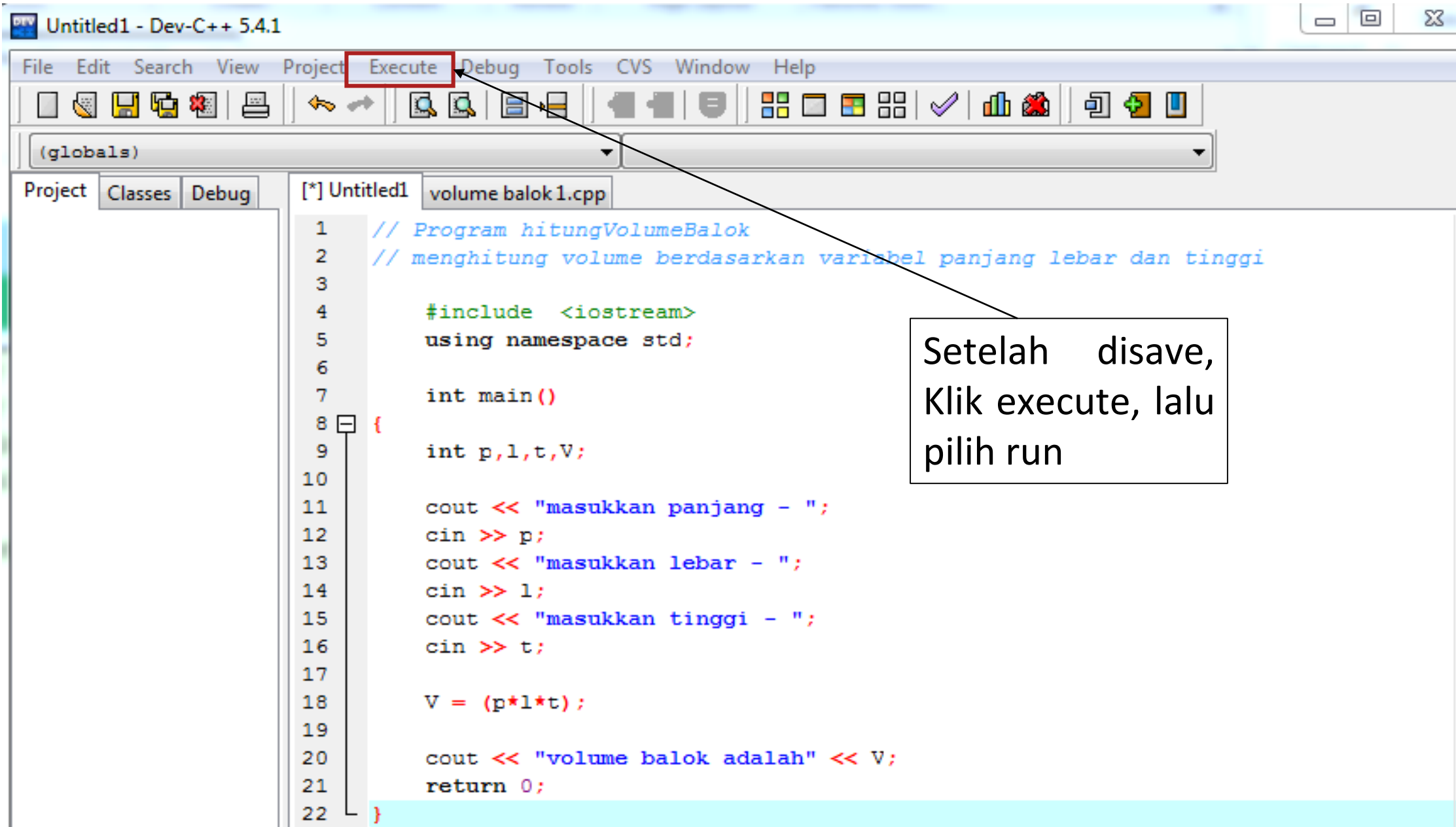




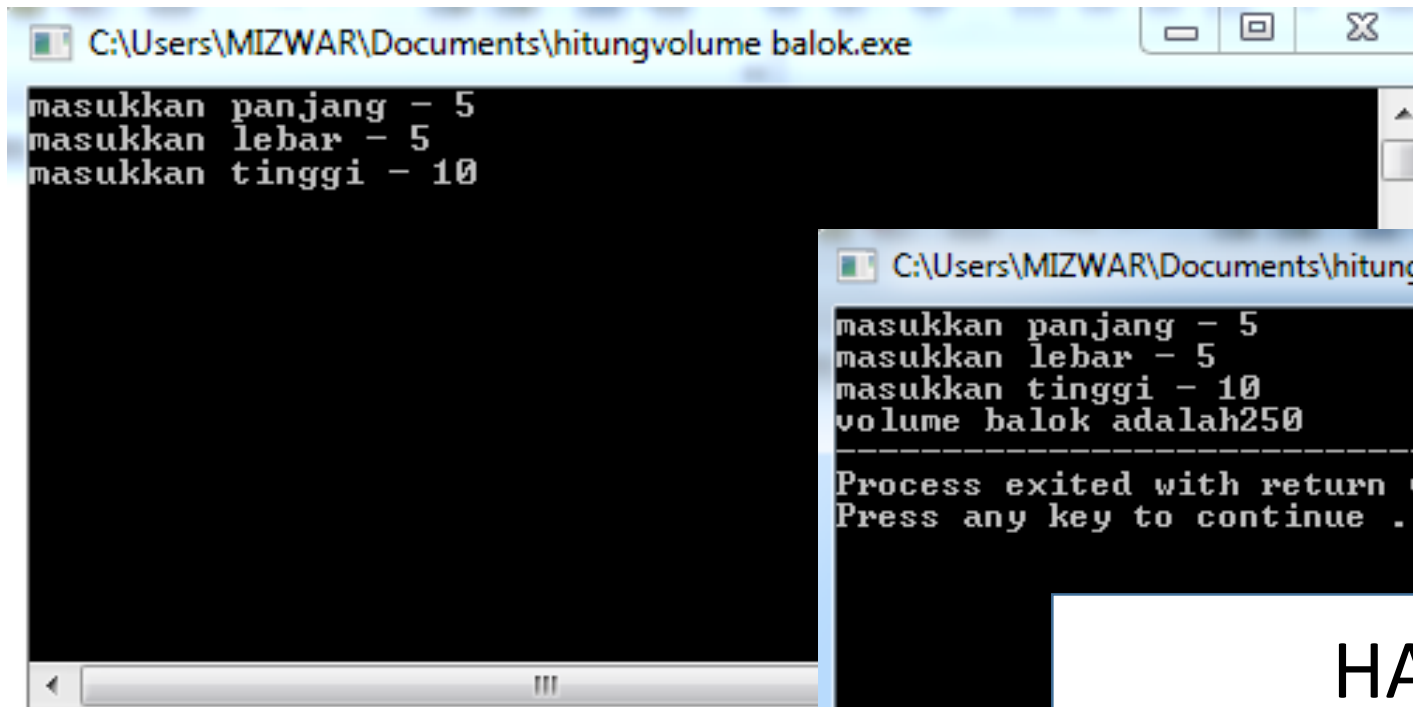


Save projek C++

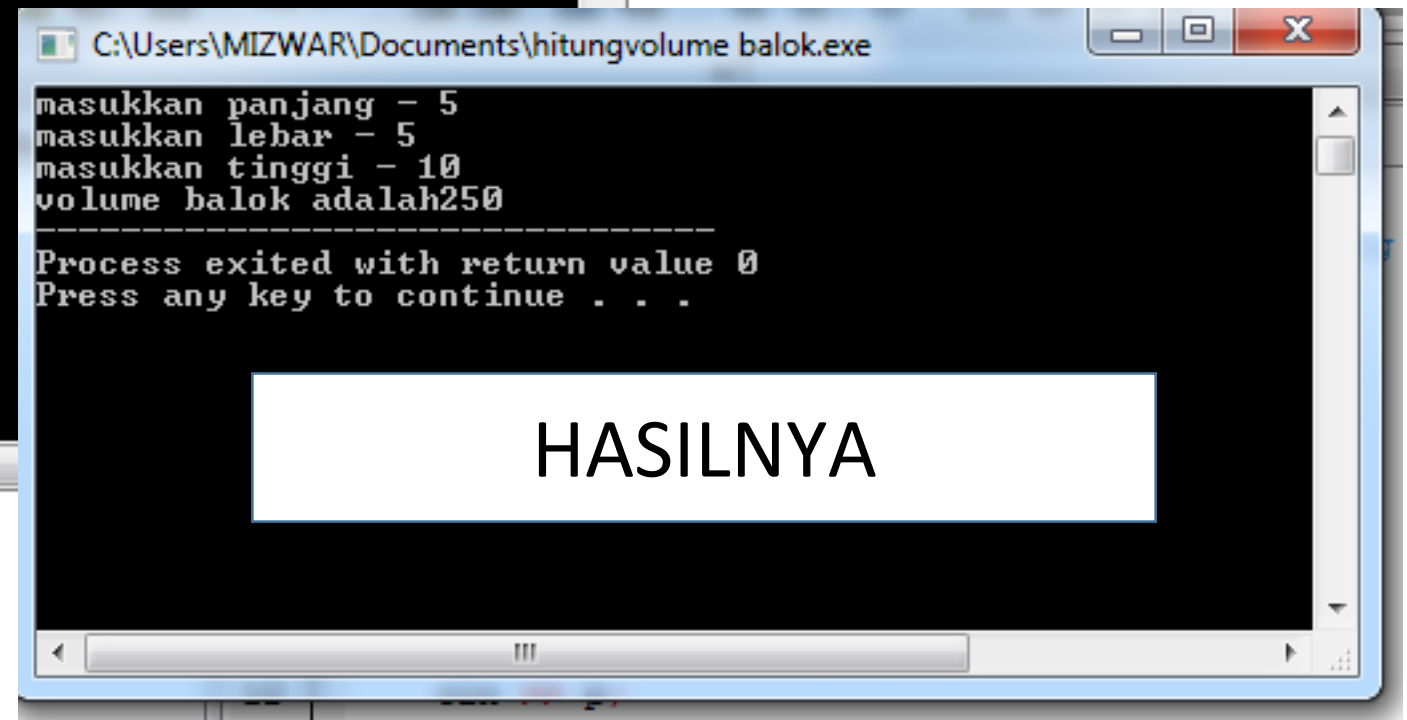




Masukkan data yang ingin kita hitung



```
C:\Users\MIZWAR\Documents\hitungvolume balok.exe  
masukkan panjang - 5  
masukkan lebar - 5  
masukkan tinggi - 10
```



```
C:\Users\MIZWAR\Documents\hitungvolume balok.exe  
masukkan panjang - 5  
masukkan lebar - 5  
masukkan tinggi - 10  
volume balok adalah 250  
-----  
Process exited with return value 0  
Press any key to continue . . .
```

HASILNYA



LATIHAN

- Buat program hitung luas segitiga
- Buat program menghitung rata-rata tinggi badan 5 orang mahasiswa
 - Program akan menerima masukan data tinggi badan untuk 5 orang mahasiswa
 - Kemudian program akan menghitung tinggi rata-rata dari lima mahasiswa tersebut.



TERIMA KASIH

