Disusun oleh Rhainy Aulia, 2023

# Algoritma

**Asal kata Algoritma** (algorism - algorithm) berasal dari nama Abu Ja’far Muhammad ibn Musa Al-Khuwarizmi

Ilmuan Persia yang menulis kitab “Al Jabar W’Al- Muqabala” (Rules of Restoration and Reduction – Aturan Pemugaran dan Pengurangan)

* Tahun 825 M • Masuk Indonesia tahun 1980-an
* Berasal dari Iran

**Definisi** Alur pemikiran dalam menyelesaikan suatu pekerjaan yang dituangkan secara tertulis.

*\* Suatu algoritma dapat berbeda dengan algoritma orang lain*

**Algoritma adalah** urutan logis pengambilan

keputusan untuk pemecahan masalah *Balai Pustaka (1998)*

Algoritma adalah Jantung Ilmu Komputer / Informatika

#### Menjalankan Algoritma harus :

 Mengerti setiap langkahnya

## Contoh Algoritma

Algoritma menghitung luas Segitiga:

 Masukkan alas (a)

 Masukkan tinggi (t)

 Luas ← a \* t \* 0.5

 tampilkan Luas

# Pseudo-Code

**Adalah** tulisan yang lebih menyerupai kode dibanding algoritma. Cara menuliskan algoritma dengan tingkat tinggi (High-Level).

Kombinasi bahasa yang mudah dimengerti dan notasi matematik yang tidak se-detail source code

#### Aturan Pseudo-Code :

Judul Algoritma, Deklarasi Variabel, Deskripsi **Perbandingan Algoritma & Pseudo-Code # Mencari bilangan dari 2 bilangan input**

 Mengerjakan operasi sesuai langkah

 Ditulis dalam notasi bahasa pemograman

**Program :** perwujudan atau implementasi teknis Algoritma yang ditulis dalam bahasa pemrogaman tertentu sehingga dapat dilaksanakan oleh komputer.

#### Algoritma :

1. Masukkan bilangan pertama
2. Masukkan bilangan kedua
3. Jika bilangan pertama >

#### Pseudo-code :

1. Input a
2. Input b
3. If a > b then kerjakan langkah 4 else kerjakan langkah 5

Bahasa Pemograman

OBJ Software

Compile

bilangan kedua maka kerjakan langkah 4, jika tidak, kerjakan langkah 5.

1. Tampilkan bilangan pertama
2. print a
3. print b

\*OBJ tidak bisa diedit, hanya bisa hex edit

**Kriteria Algoritma** (Donald E. Knuth)

1. Tampilkan bilangan kedua

#### Input Output

: Dapat memiliki nol atau lebih inputan dari luar.

: Harus memiliki minimal satu buah output keluaran.

# Flowchart

**Adalah** algoritma menggunakan simbol.

#### Komponen Flowchart :

**Definiteness**

(pasti)

: Memiliki instruksi yang jelas dan tidak ambigu.

Input

Percabangan

Perulangan

Proses

Output

**Finiteness** (ada :

batas/berakhir) **Effectiveness** : (tepat & efisien)

Harus memiliki titik berhenti (stopping role).

Sebisa mungkin harus dapat dilaksanakan dan efektif. *Contoh A = A \* 1 (Efektif = A\*=1)*

#### Kegunaan Flowchart :

* 1. Mendesign program
  2. Mempresentasikan Program
  3. Mempermudah penentuan alur logika (sebelum pembuatan program)
  4. Menjelaskan alur kepada orang lain (sesudah pembuatan program)

## Jenis Proses Algoritma

Intruksi dikerjakan.....

**Sequence** Process **Selection** Process **Iteration** Process

**Concurent** Process

: Secara Berurutan

: Jika memenuhi kriteria

: Selama memenuhi keadaan (kondisi)

**Tabel Komponen Flowchart**

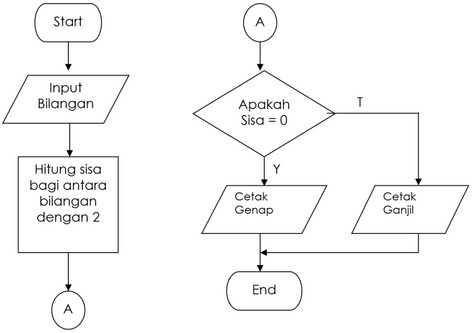
: Bersamaan intruksi lain

## Sifat Umum Algoritma

1. Tidak menggunakan simbol / sintaks pemograman.
2. Tidak bergantung suatu bahasa pemograman. 3. Notasinya dapat digunakan dengan bahasa

pemograman apapun.

## Contoh Flowchart

Menentukan bilangan apakah ganjil / genap

# Variable

**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 Variable memiliki alamat masing-masing di dalam memori komputer.

**a**

**b**

**Memori**

**c**

Penggambaran variable a, b dan c di dalam memori

# Program

## Langkah Pembuatan :

1. Mendefinisikan masalah
2. Memilih algoritma
3. Menemukan Solusi
4. Menulis program.
5. Menguji program (Debugging & Profiling)
6. Menulis dokumentasi
7. Merawat program (Menambahkan fasilitas)

## Penamaan Variabel

* 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.
  5. Case sensitive

# Konstanta

**Adalah** nilai numerik yang sifatnya tetap.

*Misalnya pi adalah konstanta karena nilainya akan selalu 22/7 atau 3.14...*

**Variabel ** Simbol dari suatu besaran yang

merepresentasikan suatu lokasi dalam memori komputer.

**Keyword** deklarasi konstanta adalah **const.**

**const tipe\_data nama\_variable = nilai\_variable;**

### const double pi = 3.14159;

**Variable = eskpresi;**

**= eskpresi;**

Dengan menggunakan keyword **const** nilai sebuah variable

Yang akan diolah terlebih dahulu adalah ekspresi, baru hasilnya dimasukkan ke dalam variable.

**a = 30;**

**b = 10;**

**a = b;**

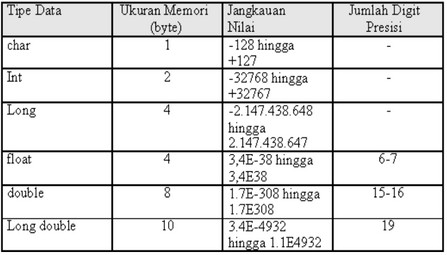
**a ** **10**

tidak bisa diubah setelah dideklarasikan.

Konstanta juga memperpendek **jangkauan tipe data**

sehingga menghemat penggunaan memori.

Simbol “=” adalah operator penugasan

**Tipe Data**

# Tipe Data

**\*1 Byte = 8 Bit (American Standard Code for Information Interchange, ASCII)**

2 byte

4 byte

4 byte

## Struktur Data

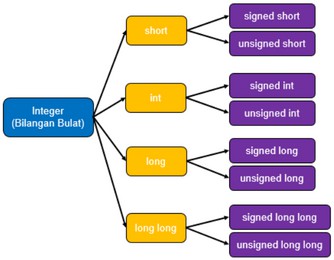
#### Tipe Data Sederhana

Cara menyimpan / merepresentasi- kan data di dalam komputer agar bisa dipakai secara efisisn.

1. Tunggal : int, real, bool, char
2. Majemuk : string

8 byte

Tipe data **Integer** digunakan untuk menyimpan numerik bilangan bulat tanpa koma



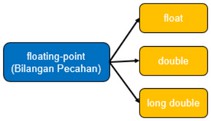
**short**

**int**

**long**

**long long**

4 byte



**float double**

**long double**

8 byte

#### Struktur Data

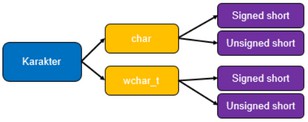
1. Tunggal : array, record
2. Majemuk :
   * Linier : Stack, Queue, List, Multilist
   * Non-Linier : Pohon Biner, Graph

12 byte

Tipe data **Floating Point** digunakan untuk menyimpan

Jika digunakan secara tepat, lebih efisien

numerik bilangan berkoma



**char**

**wchar\_t**

Tipe data **Karakter** digunakan untuk menyimpan kode karakter. Kode karakter merupakan suatu bilangan bulat yang terkait karakter yang diwakilkan. (ASCII; A=65)

#### char

* 1 byte
* alfabet, angka, tanda baca & char control seperti \n (newline) dan \t (tab)
* diapit dengan tanda kutip tunggal ; 'Á'

#### wchar\_t

* setidaknya 2 byte
* menyimpan karakter unicode modern, karakter besar seperti bahasa Cina

*Char hanya bisa menampilkan lebih dari 1 char jika menggunakan array ([ ])*



**bool**

Tipe data **Boolean** digunakan untuk menghasilkan nilai benar (true) atau salah (false) menggunakan gerbang logika AND atau OR.

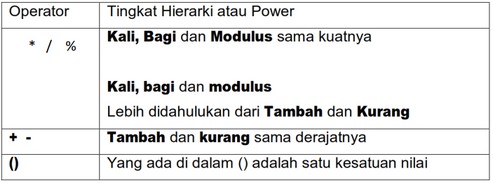
# Arithmetic Expression^

## X = A+B\*(C-D);

A, B, C D adalah **Operand** (Yang dioperasikan)

+, \*, -, (), atau % dan / adalah **Operator**

## Hierarki Operator

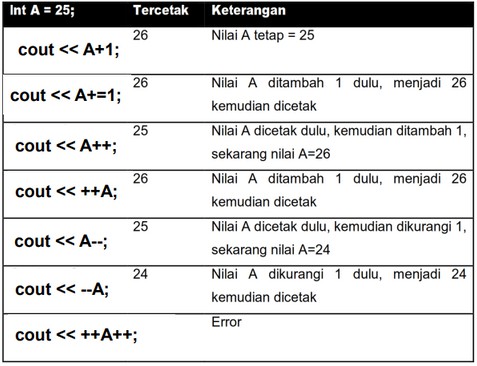


**% (modulus/sisa bagi)** hanya dapat digunakan pada tipe data **integer** (bilangan bulat)

## Increment & Decrement

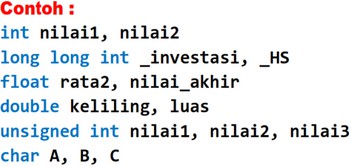
**bool benar = true bool salah = false**

representasi 1

representasi 0

**bool nama\_variabel = nilai\_variabel**

# Deklarasi Variable



Dengan Tipe Data

# Operator Input & Output

Assignment Statement

**#include <iostream>**

## cout <<

Library Input Output Stream

atau Pernyataan Penugasan **Adalah** proses mengisi sebuah variable dengan sebuah nilai.

**Variable = eskpresi; **  **= eskpresi;**

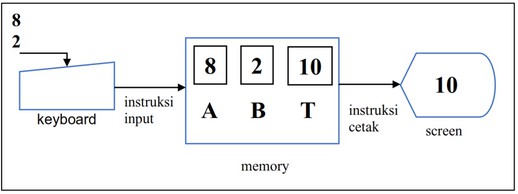
nilai ekspresi harus cocok dan sesuai dengan tipe data si variable

Menampilkan informasi ke standar output (layar) endl << / endline (enter)

## cin >>

Menerima informasi dari standar input (keyboard)

## proses input output



**c = a;**

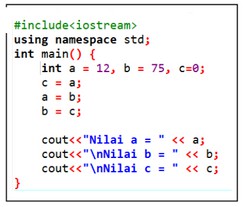
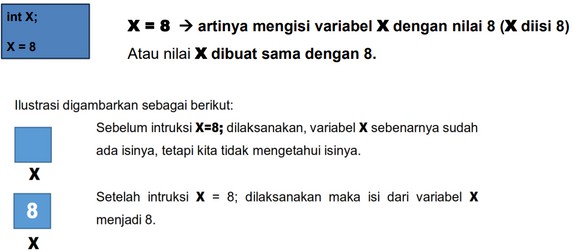
**/co**

# Statement IF^^

**Dapat diartikan sebagai** pemilihan keputusan atau juga percabangan pada sebuah program.

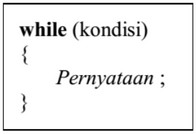
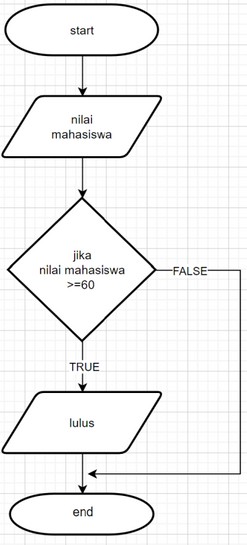
**if (kondisi) {**

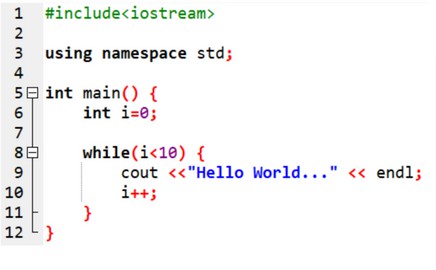
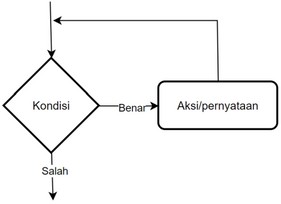
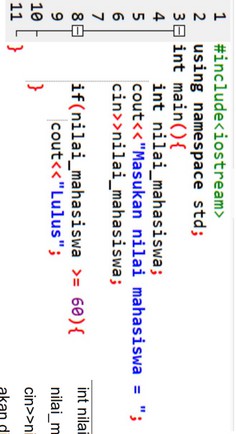
**= a dimasukkan ke c**



**kode yang di eksekusi jika benar**

**}**

**Contoh :** seorang dosen memberikan tugas kepada mahasiswa jika seorang mahasiswa mendapatkan nilai lebih dari 60 maka mahasiswa akan dinyatakan lulus



**Bentuk Umum :**

**Flowchart :**

**Kode :**

**Output :** Hello World... Hello World... Hello World... dst. 10x

## Operator Penghubung (Relational)



/co **: A=7, B=8; if(A<=B){cout "betul"} : betul**

## Operator Logika



And : KEDUA kondisi harus bersifat True

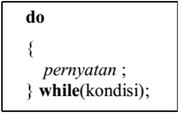
Or : Hanya perlu 1 kondisi yang bersifat True And : Membalikkan Nilai, True -> False

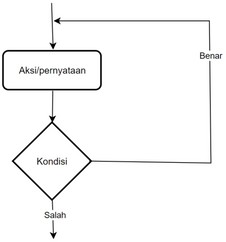
## Jenis IF

**Do While**

Mirip While, hanya saja pernyataan yang terdapat di dalamnya minimal akan sekali dieksekusi.

#### Karakteristik :

1. Menjalankan dulu pernyataan awal sebelum tanpa memeriksa kondisi.
2. Memeriksa kondisi untuk melakukan perulangan
3. Perulangan dilakukan sampai kondisi false



**Bentuk Umum :**

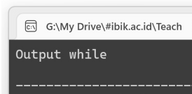
**Flowchart :**

## Perbandingan While & Do While

if bertingkat if else

if, else if

nested if (ada if di dalam if)



# Looping^^^

**Dipakai untuk** melakukan perintah secara berulang-ulang apabila suatu kondisi sudah terpenuhi atupun sebaliknya

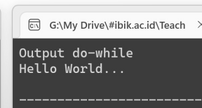
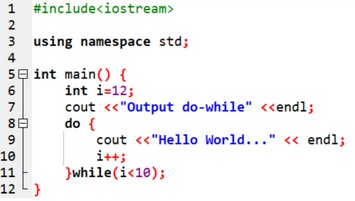
## While

Memungkinkan statement di dalamnya tidak dijalankan sama sekali

#### Karakteristik :

1. Memeriksa kondisi sebelum melakukan perulangan
2. Perulangan dilakukan sampai kondisi false

karena sedari awal tidak memenuhi syarat kondisi, output while tidak muncul.



Menjalankan pernyataan awal sekali lalu berhenti setelah mengetahui pernyataan awal tidak memenuhi syarat.

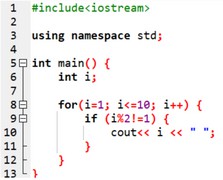
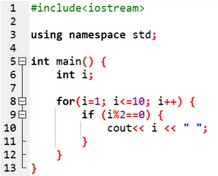
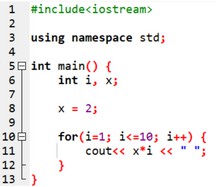
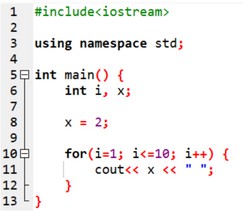
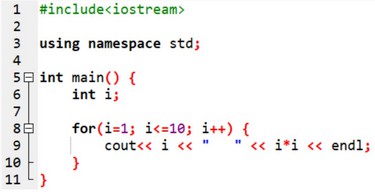
## For

Dipakai apabila kita sudah mengetahui berapa kali looping yang akan digunakan.

**for (initialization; kondisi; rubah\_kondisi){ Pernyataan1;**

Output :

1 2 3 4 5 6 7 8 9 10



Nilai X = 55

(Bilangan Berurut Kesamping dan Jumlah deret)

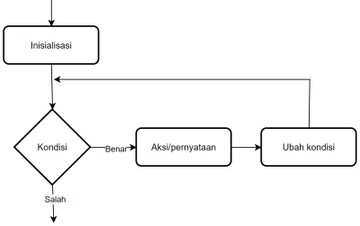
**Pernyataan2;**

**//tempat banyaknya pernyataan}**

#### Bentuk Umum

Output : 1 1

2 4



**Flowchart**

3 9

4 16

5 25

6 36

7 49

8 64

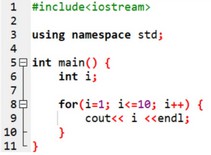
9 81

(Bilangan Berurut dan Hasil perkalian dengan diri sendiri)

10 100

1. Inisialisasi: pernyataan ini biasa dioakai untuk menginisialisasikan untuk pengendalian pengulangan tersebut.
2. Dalam pengulangan, kondisi dipakai untuk menyatakan dan menentukan apakah proses perulangan akan dilanjutkan atau tidak.
3. Ubah\_kondisi : tahap ini biasanya digunakan untuk merubah variable dalam pengatur ataupengendali perulangan.

**Contoh FOR** Output :

1

2

3

* 1. (Bilangan
  2. Berurut dengan

6

7 Enter)

8

9

10

Output :

2 2 2 2 2 2 2 2 2 2

(Perulangan 1 Nilai)

Output :

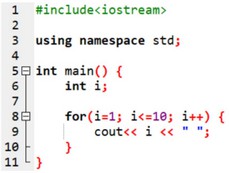
2 4 6 8 10 12 14 16 18 20

(Bilangan Berurut di kalikan 2)

Output :

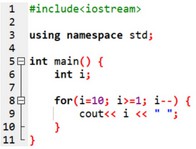
2 4 6 8 10

(Deret Bilangan genap)

Output :

1 2 3 4 5 6 7 8 9 10

(Bilangan Berurut Kesamping)

Output :

10 9 8 7 6 5 4 3 2 1

(Bilangan Mundur Kesamping)

Output :

1 3 5 7 9

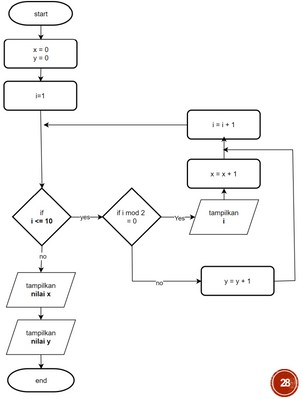
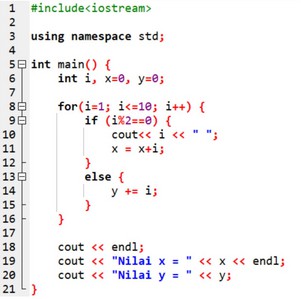
(Deret Bilangan Ganjil)

START

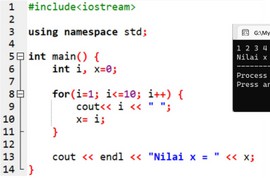
X=0 Y=0

i=1

i = i+1



Output :



1 2 3 4 5 6 7 8 9 10

Nilai X = 10

(Bilangan Berurut dan Bilangan terakhir)

Output :

2 4 6 8 10

Nilai X = 30 Nilai Y = 25

(Deret genap, hasil penjumlahan deret genap dan deret ganjil)

if

i <=10

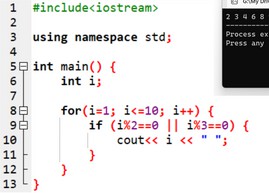
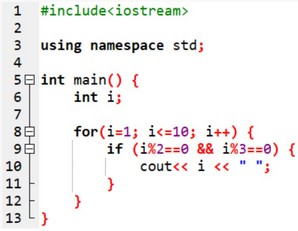
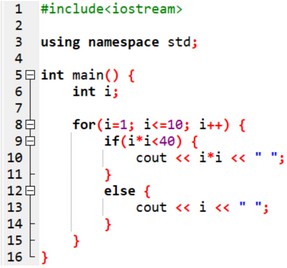
show x show y

if i%2=0

x = x+1 show i

y = y+1

**Contoh** 4x inner loop x 3x outer loop



Output :

1 4 9 16 25 36 7 8 9 10

(Mengeluarkan nilai dikali

**for(I = 1; I <= 3; I++){ for (J = 1; J<=4; J++){**

= 12x looping

dirinya sendiri selama hasil **}**

perkalian kurang dari 40, jika

lebih, maka keluarkan nilai **}**

aslinya)

Looping Process :

1 1234 2 1234 3 1234

Output :

6

Mengeluarkan nilai yang habis dibagi 2 DAN dibagi 3 dalam jangkauan

Output :

2 3 4 6 8 9 10

Mengeluarkan nilai yang habis dibagi 2 ATAU dibagi 3 dalam jangkauan

Output :

0 1 1 2 3 5 8 13 21 34

Fibonacci

Mengeluarkan nilai awal (0) lalu 1. Menukar nilai antar a b c kemudian mengulang kembali, menambahkan bilangan a dan b. dst

#### Perbedaan While, Do While & For While :

Memeriksa syarat terlebih dahulu lalu menjalankan

program selama masih memenuhi syarat

#### Do While :

Menjalankan program terlebih dahulu lalu memeriksa syarat kemudian melanjutkan program selama memenuhi syarat atau langsung berhenti jika tidak

 **For** :

Mengubah kondisi jika syarat terpenuhi

# Nested Loop

#### Nested For

**for1 (initialization1; kondisi1; rubah\_kondisi1){ for2 (initialization2; kondisi2; rubah\_kondisi2)**

**{Pernyataan2;} Pernyataan1;}**

**Merah : Outer Loop **

**Biru : Inner Loop **

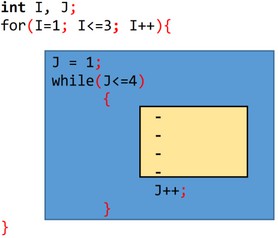
#### Nested While

**inisialisasi1;**

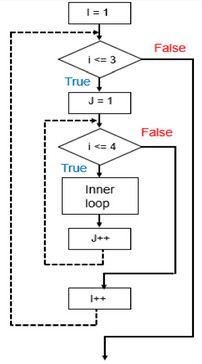
**while1 (kondisi1) { inisialisai2; while2 (kondisi2) { pernyataan2; rubah\_kondisi2}**

**pernyataan1; rubah\_kondisi1}**

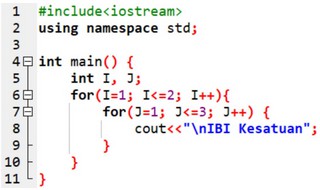
**Nested Loop Gabungan**



**Flowchart Nested Loop**



**Contoh NESTED LOOP**



**Contoh**

**int I, J; I = 1;**

**while (I<=3){ while (J<=4){**

**-**

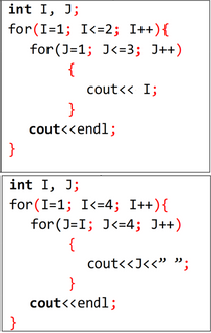
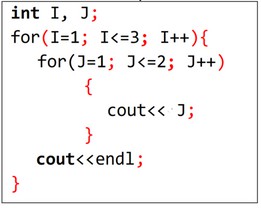
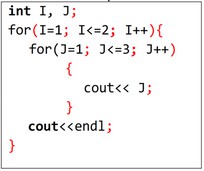
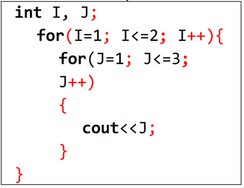
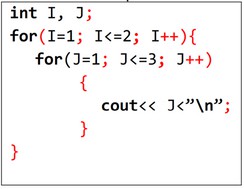
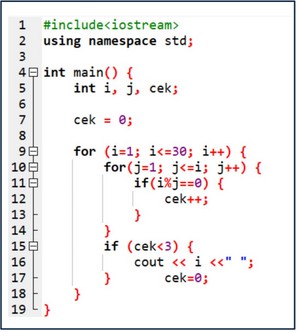
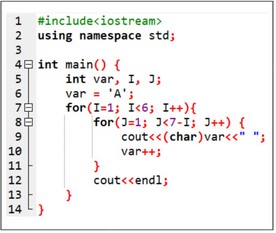
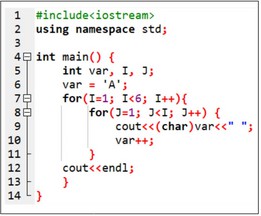
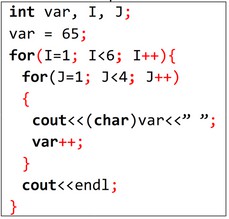
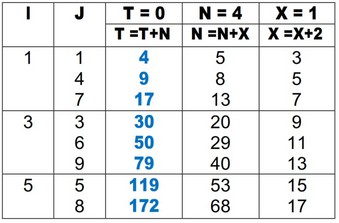
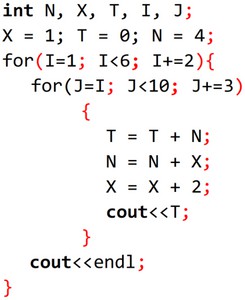
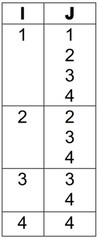
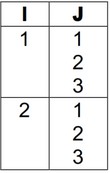
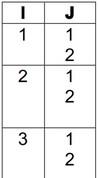
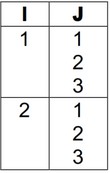
**- J++;}**

**-**

**- I++;}**

For untuk Outer Loop While untuk Inner Loop

Output : 1



2

3

1

2

3

123 adalah inner loop, setelah menyelesaikan outer loop pertama, (inner loop), dilanjut inner loop selanjutnya sampai outer loop habis.

Enter di Inner Loop

Output : A

B C D E F

G H I J

K L M N O

Penggunaan Var sudah di deklarasi dari 'A'

Output : 1 2 3

1 2 3

Output :

123123

sama dengan yang pertama, tanpa syntax \n

Output : A B C D E F G H I

J K L M N O

Enter di Outer Loop

Output :

1 2

1 2

1 2

Output :

1 1 1

2 2 2

Output :

1 2 3 4

2 3 4

3 4

4

Nilai J di Inner Loop berubah karena mengikuti I di Outer loop yang seiring waktu berubah

Output : 4 9 17

30 50 79

119 172

Output :

1 2 3 5 7 11 13 17 19 23 29

Mengeluarkan Bilangan Prima dengan basis Cek

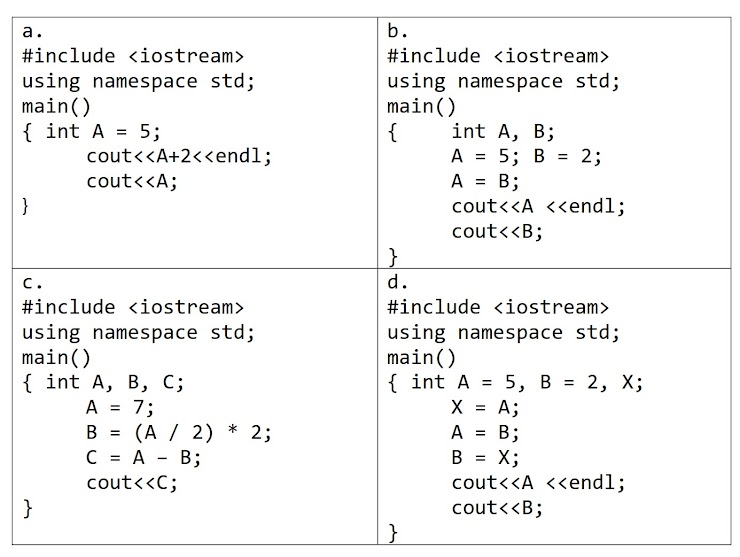
Output : A B C

D E F G H I J K L M N O

Penggunaan Char var merubah angka menjadi huruf berdasarkan tabel ASCII

QUIZ

(1). Apa yang akan tercetak jika program berikut dijalankan!



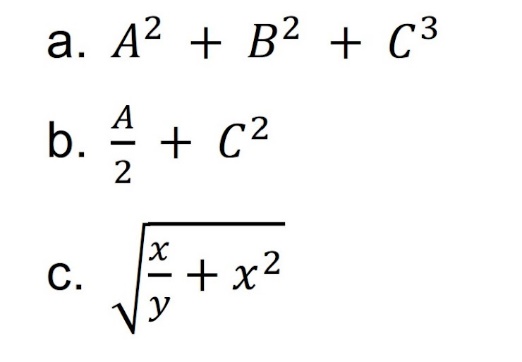
- a Untuk output a yang akan tercetak adalah: 7 5 Ini bisa terjadi karena nilai awal A = 5 dan pada cout pertama nilai A ditambah 2 kemudian di endline atau enter kebawah untuk melanjutkan ke cout yang kedua, cout yang kedua bernilai 5 karena dia hanya memanggil nilai A.

- b Untuk output b yang akan tercetak adalah: 2 2 Hal ini bisa terjadi karena ada assignment A = B yang berarti nilai B di masukkan ke dalam variabel A, karena nilai B = 2 nilai a yang awalnya 5 berubah menjadi 2.

- c Untuk output c yang akan tercetak adalah: 1 Mengapa 1. alasan di balik outputnya 1 berasal dari awal kode, semua variabel dinyatakan dalam bentuk integer yang dimana integer tidak dapat memiliki bilangan desimal, ini harus di ingat karena akan dibahas setelah ini. Pertama nilai A berubah menjadi 7 karena ada assignment A = 7 yang berarti 7 dimasukkan kedalam variabel A, selanjutnya B mengambil nilai dari operasi hitungan yang melibatkan nilai A, di situ ada A/2 yang dimana pasti akan menjadi bilangan desimal karena nilai A merupakan bilangan ganjil atau bilangan yang tidak bisa di bagi 2, karena B merupakan integer nilai yang seharusnya 3,5 berubah menjadi 3 lalu kemudian di kali 2 yang berarti nilai B = 6. kemudian ada nilai c, nilai c mengambil nilai dari operasi pengurangan A - B yang dapat di ubah menjadi 7 - 6 yang akan memiliki hasil yaitu 1. nah dari situlah output 1 berasal.

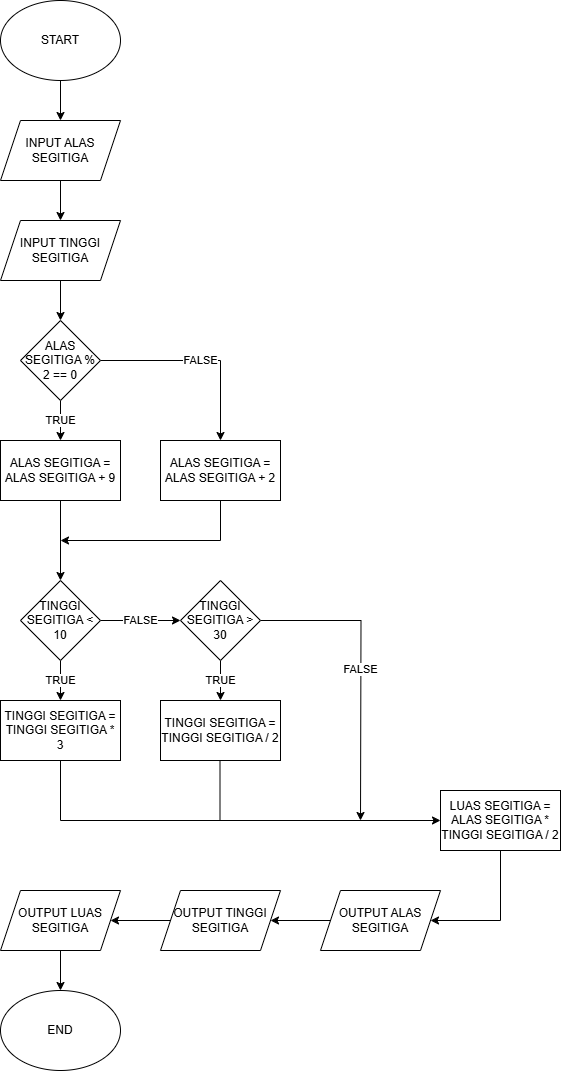
- d Untuk output d yang akan tercetak adalah: 2 5 Di nomor d ini lumayan banyak assignment, pertama nilai X berubah menjadi 5 karena ada assignment X = A yang dimana nilai A = 5, kemudian nilai A menjadi 2 kerena A = B yang dimana B = 2, yang terakhir nilai B menjadi 5 karena ada assignment B = X yang dimana X = 5 dari assignment sebelumnya. terakhir nilai A di output di lanjut dengan endl kemudian nilai B di output juga.

(2). Tulis pernyataan matematika berikut dalam bentuk pemrograman



1. A\*A + B\*B + C\*C\*C B. (A / 2.0) + pow(C, 2) C. sqrt((x / y) + pow(x, 2))

Program Menghitung Luas Segitiga

1. Algoritma Menghitung Luas Segitiga
   * Menerima input alas segitiga
   * Menerima input tinggi segitiga
   * Jika alas segitiga yang diinput genap, alas segitiga ditambah dengan 9
   * Jika alas segitiga yang diinput ganjil, alas segitiga ditambah dengan 2
   * Jika tinggi segitiga yang diinput kurang dari 10, tinggi segitiga dikali dengan 3
   * Jika tinggi segitiga yang diinput lebih dari 30, tinggi segitiga dibagi dengan 2
   * Menghitung luas segitiga, yaitu alas segitiga dikali tinggi segitiga dibagi 2
   * Output alas segitiga, tinggi segitiga, luas segitiga
2. Flowchart

