BAB II
TEORI PENUNJANG

2.1 Kriptografi

2.1.1 Sejarah Kriptografi

Kriptografi adalah ilmu yang mempelajari bagaimana suatu pesan atau dokumen kita aman, tidak bisa dibaca oleh pihak yang tidak berhak. Dalam perkembangannya, kriptografi juga digunakan untuk identifikasi pengirim pesan dengan tanda tangan digital dan keaslian pesan dengan sidik jari digital *(fingerprint)*. Kriptografi mempunyai sejarah yang sangat panjang. Sejarah kriptografi dimulai pertama sekali dengan menggunakan metode pertukaran posisi untuk mengenkripsi suatu pesan. Dalam sejarah perkembangannya, Julius Caesar dalam mengirimkan pesan yang dibawa oleh hulubalangnya, sengaja mengacak pesan tersebut sebelum diberikan kepada kurir. Hal ini dilakukan untuk menjaga kerahasiaan pesan baik bagi kurir maupun bagi musuh jika kurir tertangkap di tengah jalan oleh musuh. Ada orang yang mengatakan bahwa apa yang dilakukan oleh Julius Caesar dianggap sebagai awal dari kriptografi.

Dalam sebuah buku yang berjudul The Codebreaker yang dikarang oleh David Kahn pada tahun 1963, disebutkan bahwa kriptografi digunakan pertama sekali oleh bangsa Mesir 4000 tahun yang lalu sampai saat sekarang ini. Sejak munculnya buku tersebut maka kriptografi pun mulai diperbincangkan secara luas. Peminat dari buku tersebut ialah peminat yang berhubungan dengan kemiliteran, layanan diplomatik dan pemerintahan. Kriptografi digunakan sebagai suatu alat untuk melindungi rahasia dan strategi – strategi negara.

Gambar 2.1 Kriptografi zaman dahulu

Sampai pada akhir Perang Dunia I, kriptografi merupakan disiplin ilmu matematika yang spesial. Penelitian dalam bidang ini tidak pernah sampai kepada umum sehingga tidaklah mengherankan kalau banyak orang tidak mengetahui keberadaan ataupun manfaat darinya. Kemudian pada Perang Dunia II, pihak militer pun mulai menyadari akan manfaat dari penggunaan kriptografi maupun kriptanalisis. Kriptografi memungkinkan untuk berkomunikasi dalam saluran yang aman (misalnya komunikasi melalui radio gelombang panjang) dengan cara membuatnya menjadi tidak dapat dimengerti oleh musuh. Kriptografi mencapai kemajuan yang pesat pada akhir Perang Dunia II. Akan tetapi kriptografi masih merupakan sesuatu yang sangat rahasia karena kriptografi telah menjadi bagian yang penting dalam komunikasi militer.

Perkembangan komputer dan sistem komunikasi pada tahun 1960-an mengakibatkan munculnya kebutuhan pihak swasta akan alat untuk melindungi informasi dalam bentuk digital dan untuk menyediakan layanan keamanan informasi. Kriptografi digital dimulai pada tahun 1970 atas usaha Feistel dari IBM dan memuncak pada tahun 1977 dengan diadopsinya sistem kriptografi DES (Data Encryption Standard) oleh U.S. Federal Information Processing Standard untuk mengenkripsi informasi rahasia. DES merupakan

mekanisme kriptografi yang paling terkenal dalam sejarah dan tetap menjadi standar pengamanan data elektronik komersial pada kebanyakan institusi keuangan di seluruh dunia.

Perkembangan yang paling pesat dan berpengaruh dalam sejarah kriptografi ialah pada tahun 1976 dimana Whitfield Diffie dan Martin Hellman mempublikasikan sebuah tesis berjudul *New Direction in Cryptography*. Dalam tesis ini diperkenalkan konsep kunci publik kriptografi yang paling revolusioner dan juga menyediakan metode baru dalam pertukaran kunci, yaitu keamanan yang didasarkan atas logaritma diskrit. Walaupun penulis tesis tersebut tidak mempunyai praktek yang nyata akan bentuk skema enkripsi kunci publik pada saat itu akan tetapi ide tersebut memicu minat dan aktivitas yang besar dalam komunitas kriptografi. Pada tahun 1978, Rivest, Shamir, dan Adleman menemukan enkripsi kunci publik yang pertama dan sekarang ini dikenal dengan nama RSA (Rivest, Shamir, and Adleman). Skema RSA didasarkan pada permasalahan matematika sulit yang terdiri dari pemfaktoran terhadap bilangan yang besar nilainya. Karena adanya permasalahan matematika tersebut maka muncul usaha – usaha untuk mencari cara yang paling efisien dalam pemfaktoran bilangan. Skema kunci publik lainnya yang kuat dan praktis ditemukan oleh ElGamal. Skema ini juga berdasarkan masalah logaritma diskrit.

Salah satu sumbangan yang paling penting dari kriptografi kunci publik ialah tanda tangan digital. Pada tahun 1991, standar internasional yang pertama untuk tanda tangan digital yang dipergunakan adalah berdasarkan pada skema kunci publik RSA. Pada tahun 1994 pemerintah Amerika Serikat mengadopsi standar tanda tangan digital yang didasarkan pada mekanisme skema kunci publik ElGamal.

Pencarian terhadap skema kunci publik yang baru dengan pengembangan dari mekanisme kriptografi yang sudah ada dan pembuktian keamanan berlangsung dengan cepat. Berbagai standar dan infrastruktur yang berhubungan dengan kriptografi sedang dibangun. Produk-produk keamanan

sedang dikembangkan untuk memenuhi kebutuhan akan keamanan informasi pada masyarakat. [HTE].

2.1.2 Definisi Kriptografi

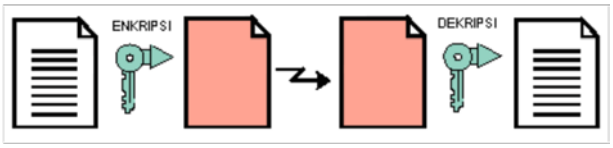
Kriptografi *(cryptography)* berasal dari kata *'kryptos'* yang artinya tersembunyi dan *'grafia'* yang artinya sesuatu yang tertulis (bahasa Yunani) sehingga kriptografi dapat juga disebut sebagai sesuatu yang tertulis secara rahasia (tersembunyi). Kriptografi merupakan seni dan ilmu menyembunyikan informasi dari penerima yang tidak berhak. Enkripsi adalah transformasi data kedalam bentuk yang tidak dapat terbaca tanpa sebuah kunci tertentu. Tujuannya adalah untuk meyakinkan privasi dengan menyembunyikan informasi dari orang-orang yang tidak ditujukan, bahkan mereka mereka yang memiliki akses ke data terenkripsi. Dekripsi merupakan kebalikan dari enkripsi, yaitu transformasi data terenkripsi kembali ke bentuknya semula. Plainimage merupakan data/pesan sebelum dilakukan proses enkripsi atau data sesudah dilakukan proses dekripsi. Chypertext merupakan data setelah dilakukan proses enkripsi.

Kriptografi adalah ilmu yang mempelajari teknik – teknik matematika yang berhubungan dengan aspek-aspek pada keamanan informasi misalnya kerahasiaan, integritas data, otentikasi pengirim / penerima data, dan otentikasi data. Dengan pengembangan bidang kriptografi, pembagian antara apa yang termasuk kriptografi dan apa yang tidak telah menjadi kabur. Dewasa ini, kriptografi dapat dianggap sebagai perpaduan antara studi teknik dan aplikasi yang tergantung kepada keberadaan masalah – masalah sulit.

Bagi kebanyakan orang, kriptografi lebih diutamakan dalam menjaga komunikasi agar tetap rahasia. Seperti yang telah diketahui dan disetujui bahwa perlindungan *(proteksi)* terhadap komunikasi yang sensitif telah menjadi penekanan kriptografi selama ini. Akan tetapi hal tersebut hanyalah sebagian dari penerapan kriptografi dewasa ini.

Terdapat dua proses penting di dalam kriptografi yang berperan dalam

merahasiakan suatu informasi yakni enkripsi *(encryption)* dan dekripsi *(decryption)*. Enkripsi adalah transformasi data *(Plainimage)* ke dalam bentuk yang hampir tidak dapat dibaca *(cipherimage)* tanpa pengetahuan yang cukup. Tujuan dari enkripsi adalah untuk menjamin kerahasiaan dengan menjaga informasi tersembunyi dari siapapun yang bukan pemilik atau yang berkepentingan dengan informasi tersebut, bahkan bagi orang yang memiliki akses terhadap data yang telah dienkripsi. Sedangkan dekripsi adalah kebalikan dari enkripsi, yakni transformasi dari data yang telah dienkripsi *(cipherimage)* kembali ke bentuk semula *(Plainimage)*. Proses enkripsi dan dekripsi pada umumnya membutuhkan penggunaan sejumlah informasi yang rahasia, yang sering disebut kunci *(key)*.



Gambar 2.2 Konsep dasar dari enkripsi dan dekripsi

Cryptographer adalah orang yang mempraktekkan ilmu kriptografi, sedangkan *cryptoanalysts* adalah orang yang mempraktekkan kriptanalisis, seni dan ilmu dalam memecahkan *cipherimage*.

Aturan fundamental kriptografi yaitu seseorang harus mengasumsikan bahwa seorang kriptoanalis menguasai algoritma umum enkripsi yang digunakan. Jumlah usaha yang diperlukan untuk menemukan, menguji, dan memasang algoritma baru yang selalu berkompromi atau berfikir untuk berkompromi dengan algoritma lama, akan menyebabkan algoritma baru itu menjadi tidak berguna untuk menjaga kerahasiaan. Sistem kriptografi atau Algoritma. Kriptografi adalah sebuah algoritma kriptografi ditambah semua kemungkinan *Plainimage*, *cipherimage* dan kunci. [HTE].

2.1.3 Jenis Algoritma Kriptografi Berdasarkan Kunci

Ada dua jenis agoritma kriptografi berdasarkan kuncinya :

1. Algoritma Simetri (Konvensional)

   Algoritma kriptografi simetris atau disebut juga algoritma kriptografi konvensional. Algoritma ini menggunakan kunci yang sama untuk proses enkripsi dan proses dekripsi. Algoritma kriptografi simetris dibagi menjadi 2 kategori yaitu algoritma aliran (Stream Ciphers) dan algoritma blok (Block Ciphers). Pada algoritma aliran, proses penyandiannya berorientasi pada satu bit atau satu byte data. Sedangkan pada algoritma blok, proses penyandiannya berorientasi pada sekumpulan bit atau byte data (per blok). Contoh algoritma kunci simetris yang terkenal adalah DES (Data Encryption Standard).

2. Algoritma Asimetris

   Algoritma kriptografi asimetris adalah algoritma yang menggunakan kunci yang berbeda untuk proses enkripsi dan dekripsinya. Algoritma ini disebut juga algoritma kunci umum (publik key algorithm) karena kunci untuk enkripsi dibuat umum (publik key) atau dapat diketahui oleh setiap orang, tapi kunci untuk dekripsi hanya diketahui oleh orang yang berwenang mengetahui data yang disandikan atau sering disebut kunci pribadi (private key). Contoh algoritma terkenal yang menggunakan kunci asimetris adalah RSA dan ECC. Syarat – syarat yang harus dipenuhi oleh suatu algoritma publik-key yaitu (Stalling, 1995):

   1. Mudah secara komputasi bagi suatu pihak B untuk mengkonstruksi sepasang kunci asimetris (kunci publik KU, kunci pribadi KR).
   2. Mudah secara komputasi bagi pengirim A, dengan memiliki kunci publik B dan pesan yang ingin dienkripsi, M, untuk menghasilkan *ciphertext* (C).
   3. Mudah secara komputasi bagi penerima B untuk mendekripsi *ciphertext* yang dihasilkan dengan menggunakan kunci pribadinya untuk mengembalikan pesan aslinya.

4. Tidak bisa secara komputasi bagi pihak ketiga untuk memperoleh kunci pribadi KRb hanya dengan mengetahui kunci publik KUb.
5. Tidak bisa secara komputasi bagi pihak ketiga untuk mengembalikan data asli M hanya dengan mengetahui kunci publik KUb dan *ciphertext*

2.1.4 Enkripsi dan Dekripsi

Enkripsi (encryption) atau enciphering (standar nama menurut ISO 7498-2) merupakan proses menyandikan *plaintext* menjadi *ciphertext*. Sedangkan dekripsi (decryption) atau deciphering (standar nama menurut ISO7498-2) merupakan proses mengembalikan*ciphertext* menjadi *plaintext* semula. Enkripsi dan dekripsi bisa diterapkan pada pesan yang dikirimkan melalui media transmisiatau pesan yang disimpan di mediasimpan (storage media).Untuk enkripsi yang diterapkan pada pesanyang dikirimkan disebut dengan encryption ofdata in motion. Contohnya adalah pengiriman nomor PIN dari mesin ATM ke komputer server di kantor pusat sebuah bank. Nomor PIN yang ada pada kartu tetap, tapi pesan dari mesin ATM ke server bank disandikan. Sedangkan enkripsi yang diterapkan pada pesan yang disimpan disebut encryption of data at rest. Contohnya enkripsi pada file teks yang tersimpan pada harddisk. Cipher Algoritma kriptografi disebut juga cipher, yaitu aturan untuk enciphering dan deciphering, atau fungsi matematika yang digunakan untuk enkripsi dan dekripsi. Konsep matematis yang mendasari algortima kriptografi adalah relasi antara dua buah himpunan, yaitu himpunan yang berisi elemen–elemen *plaintext* dan himpunan yang berisi *ciphertext*. Enkripsi dan dekripsi merupakan fungsi yang memetakan elemen– elemen antara kedua himpunan tersebut. Misalnya P menyatakan *plaintext* dan C menyatakan *ciphertext*, maka fungsi enkripsi

E memetakan P ke C :

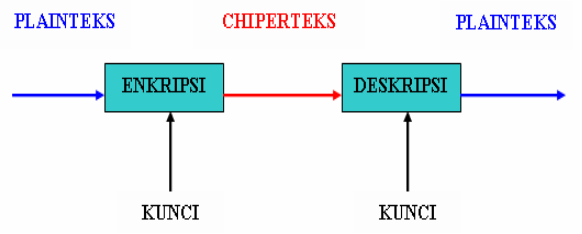
$E(P) = C$

Dan fungsi dekripsi D memetakan C ke P : $D(C) = P$

Karena proses enkripsi kemudian dekripsi mengembalikan pesan ke pesan awal, maka kesamaan berikut harus benar :

D(E(P)) = P

Keamanan algoritma kriptografi sering diukur dari banyaknya kerja (work) yang dibutuhkan untuk memecahkan *ciphertext* menjadi *plaintext*-nya tanpa mengetahui kunci yang digunakan. Kerja ini dapat diekivalenkan dengan waktu, memori, uang, dan lain-lain. Semakin banyak kerja yang dibutuhkan, juga semakin lama waktu yang dibutuhkan, maka semakin baik algoritma tersebut, berarti semakin aman digunakan untuk menyandikan pesan. [HTE].



Gambar 2.3 Gambaran umum enkripsi dan dekripsi text

2.1.5 Tujuan Kriptografi

Menurut Stalling, ada beberapa tuntutan yang terkait dengan isu keamanan data yaitu :

a) Confidentiality

Menjamin bahwa data-data tersebut hanya bisa diakses oleh pihak-pihak tertentu saja.

b) Authentication

Baik pada saat mengirim atau menerima informasi, kedua belah pihak perlu mengetahui bahwa pengirim dari pesan tersebut adalah orang yang sebenarnya seperti yang diklaim.

c) Integrity

Tuntutan ini berhubungan dengan jaminan setiap pesan yang dikirim pasti sampai pada penerimanya tanpa ada bagian dari pesan tersebut yang diganti, diduplikasi, dirusak, diubah urutannya, dan ditambahkan.

d) Nonrepudiation

Nonrepudiation mencegah pengirim maupun penerima mengingkari bahwa mereka telah mengirimkan atau menerima suatu pesan / informasi. Jika sebuah pesan dikirim, penerima dapat membuktikan bahwa pesan tersebut memang dikirim oleh pengirim yang tertera. Sebaliknya, jika sebuah pesan diterima, pengirim dapat membuktikan bahwa pesannya telah diterima oleh pihak yang ditujunya.

e) Access Control

Membatasi sumber-sumber data hanya kepada orang-orang tertentu.

f) Availability

Jika diperlukan setiap saat semua informasi pada system komputer harus tersedia bagi semua pihak yang berhak atas informasi tersebut.

Dari keenam aspek keamanan data tersebut, empat diantaranya dapat diatasi dengan menggunakan kriptografi yaitu confidentiality, integrity, authentication, dan nonrepudiation.

2.2 Pengolahan Citra

2.2.1 Image / Citra

2.2.1.1 Definisi Citra

Citra adalah suatu representasi, kemiripan, atau imitasi dari suatu obyek atau benda. Sebuah citra mengandung informasi tentang obyek yang direpresentasikan. Citra dapat dikelompokkan menjadi citra tampak dan citra tak tampak. Untuk dapat dilihat mata manusia, citra tak tampak harus dirubah menjadi citra tampak, misalnya dengan menampilkannya di monitor, dicetak di kertas dan sebagainya. Salah satu contoh citra tak tampak adalah citra digital.[1].

Gambar 2.4 Citra Kapal

Citra sebagai keluaran suatu sistem perekaman data dapat bersifat optik berupa foto, bersifat analog berupa sinyal-sinyal video seperti gambar pada monitor televisi, atau bersifat digital yang dapat langsung disimpan pada suatu pita magnetik. Gambar analog dibagi menjadi N baris dan M kolom sehingga menjadi gambar diskrit. Persilangan antara baris dan kolom tertentu disebut dengan piksel. Citra digital merupakan suatu larik dua dimensi atau suatu matriks yang elemen-elemennya menyatakan tingkat keabuan dari elemen gambar. Jadi informasi yang terkandung bersifat diskret. Citra digital tidak selalu merupakan hasil langsung data rekaman suatu sistem. Kadang-kadang hasil rekaman data bersifat kontinu seperti gambar pada monitor televisi, foto sinar-X, dan lain sebagainya. Dengan demikian untuk mendapatkan suatu citra digital diperlukan suatu proses konversi, sehingga citra tersebut selanjutnya dapat diproses dengan komputer**.**

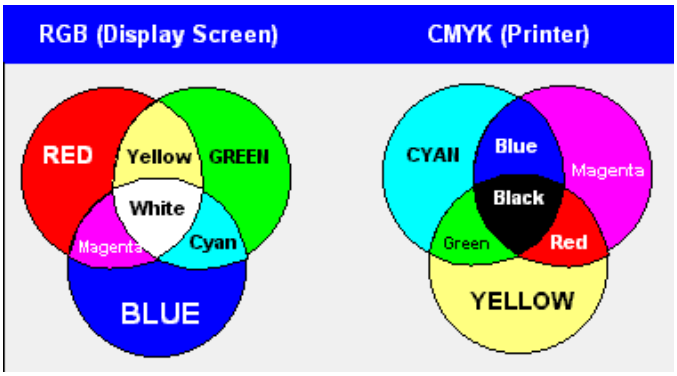
2.2.1.2 Format Citra

Ada beberapa format citra yaitu :

1. Citra berwarna adanya RGB (Red, Green, Blue) adalah berasal dari persepsi warna kita sendiri, karena mata manusia yang peka terhadap warna merah, hijau dan biru.

Kamera dan Scanners mengambil warna dengan sensor yang merekam beragam intensitas merah, hijau dan biru pada setiap pixel dalam bingkai lokasi. Untuk menampilkan layar, merah, hijau dan biru piksel (titik) yang energized intensitas yang sesuai. Saat ketiga piksel dinyalakan maksimum, akan menghasilkan warna putih. Warna dasar layar akan muncul setelah semua pixeltidakaktif. RGB tidak dapat digunakan untuk warna tinta printer, sehingga untuk tinta printer harus menggunakan warna CMYK (Cyan, Magenta, Yellow, blacK).

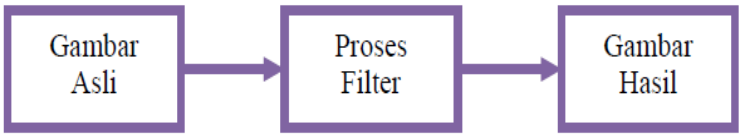
2. Citra Grayscale merupakan suatu istilah untuk menyebutkan satu citra yang memiliki warna abu-abu, hitam dan putih.
3. Citra Biner atau citra yang paling dasar / sederhana adalah citra biner, karena hanya terdiri dari dua warna saja, yaitu hitam dan putih, tidak mengenal adanya derajat keabuan (grayscale). [2].



Gambar 2.5 Format citra

2.2.2 Definisi Pengolahan Citra

Sebuah citra diubah ke bentuk digital agar dapat disimpan dalam memori komputer atau media lain. Proses mengubah citra ke bentuk digital bisa dilakukan dengan beberapa perangkat, misalnya *scanner*, kamera digital, dan *handycam*. Ketika sebuah citra sudah diubah ke dalam bentuk digital (selanjutnya disebut citra digital), bermacam-macam proses pengolahan citra dapat diperlakukan terhadap citra tersebut.*Image processing* atau sering disebut dengan pengolahan citra digital merupakan suatu proses dari gambar asli menjadi gambar lain yang sesuai dengan keinginan kita. Misal suatu gambar yang kita dapatkan terlalu gelap maka dengan *image processing* gambar tersebut bisa kita proses sehingga mendapat gambar yang jelas Secara garis besar dapat diilustrasikan seperti tampak pada gambar 2.6 dibawah ini.



Gambar 2.6 Blok diagram pengolahan citra

Pengolahan citra adalah pemrosesan citra, khususnya dengan menggunakan komputer, menjadi citra yang kualitasnya lebih baik. Sebagai contoh, citra burung nuri pada Gambar 2.7 (a) tampak agak gelap, lalu dengan operasi pengolahan citra kontrasnya diperbaiki sehingga menjadi lebih terang dan tajam (b).

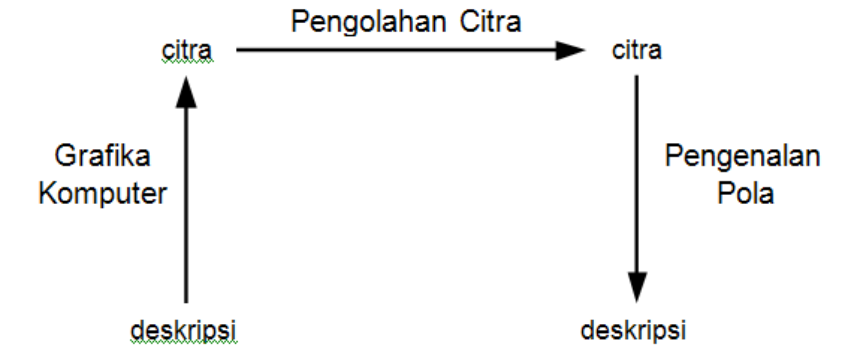
(a) (b)

Gambar 2.7 (a) Citra burung nuri yang agak gelap, (b) Citra burung yang telah diperbaiki kontrasnya sehingga terlihat jelas dan tajam

Di dalam bidang komputer, sebenarnya ada tiga bidang studi yang berkaitan dengan data citra, namun tujuan ketiganya berbeda, yaitu:

1. Grafika Komputer (*computer graphics*).
2. Pengolahan Citra (*image processing*).
3. Pengenalan Pola (*pattern recognition/image interpretation*).

Hubungan antara ketiga bidang (grafika komputer, pengolahan citra, pengenalan pola) ditunjukkan pada Gambar 2.8



Gambar 2.8 Tiga bidang studi yang berkaitan dengan citra

2.2.3 Operasi Pengolahan Citra

Operasi-operasi yang dilakukan di dalam pengolahan citra banyak ragamnya. Namun, secara umum, operasi pengolahan citra dapat diklasifikasikan dalam beberapa jenis sebagai berikut:

1. Perbaikan kualitas citra (*image enhancement*).

   Jenis operasi ini bertujuan untuk memperbaiki kualitas citra dengan cara memanipulasi parameter-parameter citra. Dengan operasi ini, ciri-ciri khusus yang terdapat di dalam citra lebih ditonjolkan.

   Contoh-contoh operasi perbaikan citra:

   a. perbaikan kontras gelap/terang
   b. perbaikan tepian objek (*edge enhancement*)
   c. penajaman (*sharpening*)
   d. pembrian warna semu (*pseudocoloring*)
   e. penapisan derau (*noise filtering*)

Gambar 2.9 adalah contoh operasi penajaman. Operasi ini menerima masukan sebuah citra yang gambarnya hendak dibuat tampak lebih tajam. Bagian citra yang ditajamkan adalah tepi-tepi objek. [3].

(a)

(b)

Gambar 2.9 (a) Citra Lena asli, (b) Citra Lena baru

2. Pemugaran citra (*image restoration*).

Operasi ini bertujuan menghilangkan/meminimumkan cacat pada citra. Tujuan pemugaran citra hampir sama dengan operasi perbaikan citra. Bedanya, pada pemugaran citra penyebab degradasi gambar diketahui.

Contoh-contoh operasi pemugaran citra:

a. penghilangan kesamaran (*deblurring*).
b. penghilangan derau (*noise*)

Gambar 2.10 adalah contoh operasi penghilangan kesamaran. Citra masukan adalah citra yang tampak kabur (*blur*). Kekaburan gambar mungkin disebabkan pengaturan fokus lensa yang tidak tepat atau kamera bergoyang pada pengambilan gambar. Melalui operasi *deblurring*, kualitas citra masukan dapat diperbaiki sehingga tampak lebih baik. [3].

(a) (b)

Gambar 2.10 Kiri: Citra Lena yang kabur (blur), kanan: citra Lena setelah deblurring

3. Pemampatan citra (*image compression*).

Jenis operasi ini dilakukan agar citra dapat direpresentasikan dalam bentuk yang lebih kompak sehingga memerlukan memori yang lebih sedikit. Hal penting yang harus diperhatikan dalam pemampatan adalah citra yang telah dimampatkan harus tetap mempunyai kualitas gambar yang bagus. Contoh metode pemampatan citra adalah metode JPEG. Perhatikan Gambar 2.11. Gambar sebelah kiri adalah citra kapal yang berukuran 258 KB. Hasil pemampatan citra dengan metode JPEG dapat mereduksi ukuran citra semula sehingga menjadi 49 KB saja. [3].

(a) (b)

Gambar 2.11 (a) Citra boat.bmp(258 KB) sebelum dimampatkan, (b) citra boat.jpg(49 KB) sesudah dimampatkan

4. Segmentasi citra (*image segmentation*).

Jenis operasi ini bertujuan untuk memecah suatu citra ke dalam beberapa segmen dengan suatu kriteria tertentu. Jenis operasi ini berkaitan erat dengan pengenalan pola.

5. Pengorakan citra (*image analysis*)

   Jenis operasi ini bertujuan menghitung besaran kuantitif dari citra untuk menghasilkan deskripsinya. Teknik pengorakan citra mengekstraksi ciri-ciri tertentu yang membantu dalam identifikasi objek. Proses segmentasi kadangkala diperlukan untuk melokalisasi objek yang diinginkan dari sekelilingnya.

   Contoh-contoh operasi pengorakan citra:

   a. Pendeteksian tepi objek (*edge detection*)
   b. Ekstraksi batas (*boundary*)
   c. Representasi daerah (*region*)

(a) (b)

Gambar 2.12 adalah contoh operasi pendeteksian tepi pada citra *Camera*. Operasi ini menghasilkan semua tepi (*edge*) di dalam citra.

6. Rekonstruksi citra (*image reconstruction*)

   Jenis operasi ini bertujuan untuk membentuk ulang objek dari beberapa citra hasil proyeksi. Operasi rekonstruksi citra banyak digunakan dalam bidang medis. Misalnya beberapa foto *rontgen* dengan sinar *X* digunakan untuk membentuk ulang gambar organ tubuh.

2.2.4 Aplikasi Pengolahan Citra

Pengolahan citra mempunyai aplikasi yang sangat luas dalam berbagai bidang kehidupan. Di bawah ini disebutkan beberapa aplikasi dalam beberapa bidang

1. Bidang perdagangan
   (a) Pembacaan kode batang (*bar code*) yang tertera pada barang
   (b)Mengenali huruf/angka pada suatu formulir secara otomatis.
2. Bidang militer
   (a) Mengenali sasaran peluru kendali mela lui sensor visual.
   (b) Mengidentifikasi jenis pesawat musuh
3. Bidang kedokteran
   (a) Pengolahan citra sinar X untuk mammografi (deteksi kanker payudara)
   (b) NMR (*Nuclear Magnetic Resonance*)
   (c) Mendeteksi kelainan tubuh dari foto sinar X.
   (d) Rekonstruksi foto janin hasil USG
4. Bidang biologi
   Pengenalan jenis kromosom melalui gambar mikroskopik
5. Komunikasi data
   Pemampatan citra yang ditransmisi.
6. Hiburan
   Pemampatan video (*MPEG*)
7. Robotika
   Visualy-guided autonomous navigation
8. Pemetaan
   Klasif ikasi penggunaan tanah melalui foto udara/LANDSAT
9. Geologi
   Mengenali jenis batu-batuan melalui foto udara/LANDSAT

10. Hukum
    (a) Pengenalan sidik jari
    (b) Pengenalan foto narapidana.

2.3 Permutasi

Permutasi adalah penyusunan kembali suatu kumpulan objek dalam urutan yang berbeda dari urutan yang semula. Sebagai contoh, kata-kata dalam kalimat “saya sedang makan sate” dapat disusun kembali sebagai "*sate sedang makan saya.*" Proses mengembalikan objek-objek tersebut pada urutan yang baku (sesuai ketentuan) disebut sorting.

Jika terdapat suatu untai abjad *abcd*, maka untai itu dapat dituliskan kembali dengan urutan yang berbeda: *acbd*, *dacb*, dan seterusnya. Selengkapnya ada 24 cara menuliskan keempat huruf tersebut dalam urutan yang berbeda satu sama lain.

abcd	**abdc**	**acbd**	**acdb**	**adbc**	**adcb**
bacd	**badc**	**bcad**	**bcda**	**bdac**	**bdca**
cabd	**cadb**	**cbad**	**cbda**	**cdab**	**cdba**
dabc	**dacb**	**dbac**	**dbca**	**dcab**	**dcba**

Gambar 2.13 Permutasi dari abcd

Setiap untai baru yang tertulis mengandung unsur-unsur yang sama dengan untai semula *abcd*, hanya saja ditulis dengan urutan yang berbeda. Maka setiap untai baru yang memiliki urutan berbeda dari untai semula ini disebut dengan permutasi dari *abcd*. [HTE].

2.4 Secure Image Protection

2.4.1 Sejarah SIP

Pada sekitar tahun 2003 seorang peniliti melakukan sebuah penelitian sebuah algoritma enkripsi gambar kacau berdasarkan pada pemetaan Baker, algoritma ini sebelumnya juga mengadopsi algoritma Fidrich dimana algoritma yang ditemukan oleh Fidrich merupakan algoritma yang mengacu pada enkripsi untaian-untaian matriks yng dibentuk dari uraian tiap-tiap piksel sebuah gambar. Fidrich menginspirasikan pada algoritma yang berdasarkan pada pemetaan Baker ini, karena pada dasarnya algoritma yang berdasarkan pada pemetaan Baker ini sama mengacu pada 2 dimensi yaitu permutasi piksel atau fungsi transformasi dan fungsi grayscale atau skala keabuan. Ada beberapa kelemahan pada algoritma ini yaitu terutama pada tingkat kekuatan kunci. Ternyata ini adalah hal yang memang berkesinambungan karena setelah algoritma yang berdasarkan pada pemetaan Baker muncul algoritma baru yang juga merupakan pengembangan yaitu algoritma pemetaan kacau. Algoritma ini dikembangkan setelah algoritma yang berdasarkan pada pemetaan Baker namun tetap pada masa yang sama yaitu sekitar tahun 2003.

Penelitian demi penelitian dilakukan oleh banyak peneliti sampai akhirnya ditemukan sebuah gagasan untuk tetap meneruskan algoritma ini tetap mengacu pada algoritma sebelum-sebelumnya. Perbedaan antara algoritma ini dengan algoritma-algoritma enkripsi yang lain adalah algoritma ini yang dilakukan proses enkripsi adalah piksel dari gambar bukan bit dari tiap gambar dan juga tidak terbatas dari gambar persegi saja seperti algoritma-algoritma konvensional lainnya yang ada. Algoritma ini mampu mengolah gambar yang berbentuk persegi panjang. [SMI]

2.4.2 Definisi SIP

Secure Image Protection (SIP) secara umum terdiri dari tiga komponen utama:

(1) fungsi transformasi horizontal-vertikal(HVT);

(2) fungsi *shift* (S), dan

(3) fungsi skala abu-abu (GS).

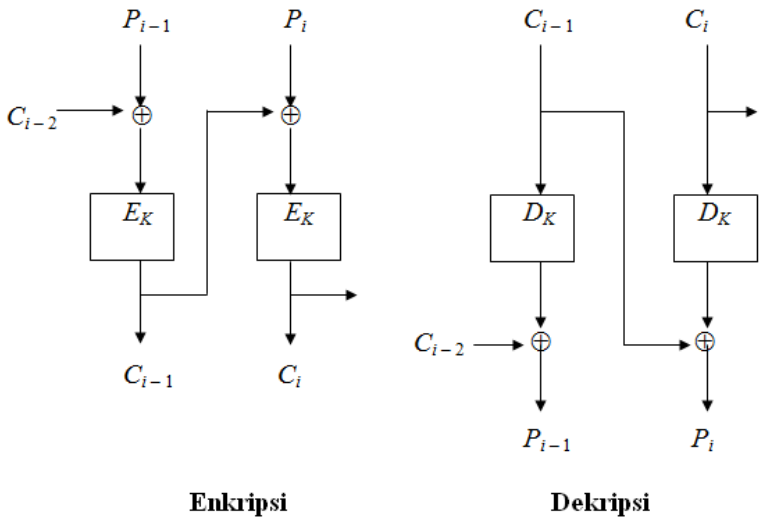
HVT fungsi didasarkan pada pemetaan kacau dua dimensi yang digunakan pada algoritma *chaotic mapping*. Fungsi *Grayscale* meluas ke tiga-dimensi, dimana, yang ketiga dimensi mengacu pada tingkat skala-keabuan pixel. Algoritma yang mendukung dua mode operasi, yaitu *Electronic Code Book* (ECB) dan *Cipher Blok Chaining* (CBC).

Cipher Block Chaining (CBC) mode ini menerapkan mekanisme umpan-balik (*feedback*) pada sebuah blok, yang dalam hal ini hasil enkripsi blok sebelumnya di-umpan-balikkan ke dalam enkripsi blok yang *current*.

Caranya, blok plainteks yang current di-XOR-kan terlebih dahulu dengan blok cipherteks hasil enkripsi sebelumnya, selanjutnya hasil peng-XOR-an ini masuk ke dalam fungsi enkripsi.

Dengan mode CBC, setiap blok cipherteks bergantung tidak hanya pada blok plainteksnya tetapi juga pada seluruh blok plainteks sebelumnya.

Dekripsi dilakukan dengan memasukkan blok cipherteks yang current ke fungsi dekripsi, kemudian meng-XOR-kan hasilnya dengan blok cipherteks sebelumnya. Dalam hal ini, blok cipherteks sebelumnya berfungsi sebagai umpan-maju (feedforward) pada akhir proses dekripsi.
Skema enkripsi dan dekripsi dengan cipher blok digambarkan pada Gambar 2.14



Gambar 2.14 Skema enkripsi dan dekripsi dengan mode CBC

Secara matematis, enkripsi dengan mode CBC dinyatakan sebagai

Ci = EK(Pi Ci – 1)

dan dekripsi sebagai

Pi = DK(Ci) Ci – 1

Teknik Kriptografi Klasik yang Digunakan pada Cipher Blok. Algoritma blok cipher menggabungkan beberapa teknik kriptografi klasik dalam proses enkripsi. Dengan kata lain, cipher blok dapat diacu sebagai super-enkripsi.

Teknik kriptografi klasik yang digunakan adalah:

1. Substitusi.

Teknik ini mengganti satu atau sekumpulan bit pada blok plainteks tanpa mengubah urutannya. Secara matematis, teknik substitusi ini ditulis sebagai

ci = E(pi), i = 1, 2, … (urutan bit)

yang dalam hal ini ci adalah bit cipherteks, pi adalah bit plainteks, dan f adalah fungsi substitusi.

Dalam praktek, E dinyatakan sebagai fungsi matematis atau dapat merupakan tabel susbtitusi (S-box).

2. Transposisi atau permutasi

Teknik ini memindahkan posisi bit pada blok plainteks berdasarkan aturan tertentu. Secara matematis, teknik transposisi ini ditulis sebagai

C = PM

yang dalam hal ini C adalah blok cipherteks, P adalah blok plainteks, dan M adalah fungsi transposisi.

Dalam praktek, M dinyatakan sebagai tabel atau matriks permutasi.

3. Ekspansi

Teknik ini memperbanyak jumlah bit pada blok plainteks berdasarkan aturan tertentu, misalnya dari 32 bit menjadi 48 bit. Dalam praktek, aturan eskpansi dinyatakan dengan tabel.

4. Kompresi

Teknik ini kebalikan dari ekspansi, di mana jumlah bit pada blok plainteks diciutkan berdasarkan aturan tertentu. Dalam praktek, aturan kompresi dinyatakan dengan tabel. [HTE].

2.5 Pemrograman Java

Java sebagai salah satu bahasa pemrograman baru menjanjikan banyak kemudahan bagi programer junior maupun senior. Tutorial ini akan membawa Anda mengenal lebih jauh bahasa ini melalui pembahasan konsep model perancangan dan petunjuk sederhana penggunaannya.

2.5.1 Pengertian Java

Java adalah bahasa pemrograman berorientasi objek yang dikembangkan oleh Sun Microsystems sejak tahun 1991. Bahasa ini dikembangkan dengan model yang mirip dengan bahasa C++ dan Smalltalk, namun dirancang agar lebih mudah dipakai dan platform independent, yaitu dapat dijalankan di berbagai jenis sistem operasi dan

arsitektur komputer. Bahasa ini juga dirancang untuk pemrograman di Internet sehingga dirancang agar aman dan portabel.

2.5.2 Platform Independent

Platform independent berarti program yang ditulis dalam bahasa Java dapat dengan mudah dipindahkan antar berbagai jenis sistem operasi dan berbagai jenis arsitektur komputer. Aspek ini sangat penting untuk dapat mencapai tujuan Java sebagai bahasa pemrograman Internet di mana sebuah program akan dijalankan oleh berbagai jenis komputer dengan berbagai jenis sistem operasi. Sifat ini berlaku untuk level source code dan binary code dari program Java. Berbeda dengan bahasa C dan C++, semua tipe data dalam bahasa Java mempunyai ukuran yang konsisten di semua jenis platform. Source code program Java sendiri tidak perlu dirubah sama sekali jika Anda ingin mengkompile ulang di platform lain. Hasil dari mengkompile source code Java bukanlah kode mesin atau instruksi prosesor yang spesifik terhadap mesin tertentu, melainkan berupa bytecode yang berupa file berekstensi .class. Bytecode tersebut dapat langsung Anda eksekusi di tiap platform yang dengan menggunakan Java Virtual Machine (JVM) sebagai interpreter terhadap bytecode tersebut.

JVM sendiri adalah sebuah aplikasi yang berjalan di atas sebuah sistem operasi dan menerjemahkan bytecode program Java dan mengeksekusinya, sehingga secara konsep bisa dianggap sebagai sebuah interpreter. Proses pengeksekusian program Java dapat dilukiskan seperti di Gambar 4. Dengan cara ini, sebuah program Java yang telah dikompilasi akan dapat berjalan di platform mana saja, asalkan ada JVM di sana. [HTE].

Gambar 2.15 Java Virtual Machine

Kompiler dan interpreter untuk program Java berbentuk Java Development Kit (JDK) yang diproduksi oleh Sun Microsystems. JDK ini dapat didownload gratis dari situs java.sun.com. Interpreter untuk program Java sendiri sering juga disebut Java Runtime atau Java Virtual Machine. Interpreter Java, tanpa kompilernya, disebut Java Runtime Environment (JRE) dapat didownload juga di situs yang sama. Untuk mengembangkan program Java dibutuhkan JDK, sementara jika hanya ingin menjalankan bytecode Java cukup dengan JRE saja. Namun untuk mengeksekusi applet (sebuah bytecode Java juga) Anda biasanya tidak perlu lagi mendownload JRE karena browser yang Java-enabled telah memiliki JVM sendiri.

2.5.3 Library

Selain kompiler dan interpreter, bahasa Java sendiri memiliki library yang cukup besar yang dapat mempermudah Anda dalam membuat sebuah aplikasi dengan cepat. Library ini sudah mencakup untuk grafik, desain user interface, kriptografi, jaringan, suara, database, dan lain-lain.

2.5.4 Pemrograman Berorientasi Objek

Java adalah bahasa pemrograman berorientasi objek. Pemrograman berorientasi objek secara gamblang adalah teknik untuk mengorganisir program dan dapat dilakukan dengan hampir semua bahasa pemrograman. Namun Java sendiri telah mengimplementasikan berbagai fasilitas agar seorang programer dapat mengoptimalkan teknik pemrograman berorientasi objek.

Sedikit perbandingan tambahan dengan bahasa C dan C++, Java banyak mewarisi konsep orientasi objek dari C++ namun dengan

menghilangkan aspek-aspek kerumitan dalam bahasa C++ tanpa mengurangi kekuatannya. Hal ini mempermudah programer pemula untuk mempelajari Java namun mengurangi keleluasaan programer berpengalaman dalam mengutak-atik sebuah program. Di balik kemudahan yang ditawarkan Java, luasnya fasilitas library Java sendiri membuat seorang programer membutuhkan waktu yang tidak singkat untuk dapat menguasai penggunaan library-library tersebut.

2.5.5 Memulai Pemrograman Java

Untuk membuat program Java, seperti telah disebutkan sebelumnya, Anda membutuhkan JDK. Proses instalasi JDK tersebut sangat mudah dan tidak membutuhkan pengetahuan tertentu. Namun untuk menggunakannya Anda perlu melakukan beberapa penyesuaian dengan sistem operasi Anda. Umumnya yang perlu Anda lakukan adalah memasukkan path ke direktori JDK Anda ke setting path pada sistem operasi Anda. Misalkan direktori JDK Anda adalah C:\jdk1.4 maka pada Windows 98 Anda cukup menambahkan baris perintah SET PATH=C:\jdk1.4\bin pada file autoexec.bat Anda. Untuk Windows NT/2000/XP Anda cukup menambahkan direktori C:\jdk1.4\bin pada variabel path di System Environment. Caranya: klik kanan ikon *My Computer*, pilih *Properties*. Kemudian pilih tab *Advanced*. Lalu klik tombol *Environment Variables*, cari variabel path, kemudian tambahkan path direktori JDK Anda ke dalam variabel tersebut. Untuk Linux, tambahkan baris perintah SET CLASSPATH=*(direktori jdk Anda)* ke file profile Anda. Untuk mencoba JDK, ketikkan perintah java dan javac pada shell prompt (atau DOS Command Prompt). Jika perintah tersebut sudah dikenali maka program java atau javac akan menampilkan sintaks penggunaan. Untuk kemudahan dan berbagai fasilitas tambahan Anda dapat menggunakan Integrated Development Environment (IDE) untuk bahasa Java seperti **Visual Café** dari Symantec atau **JBuilder** dari Borland.

Urutan langkah-langkah yang harus Anda lakukan untuk membuat sebuah program Java sederhana adalah:

1. Membuat source code program dengan editor teks apapun. Ingat, file tersebut harus berekstensi .java dan case sensitive.
2. Mengkompile source code dengan perintah javac. Misalnya: javac HelloWorld.java. Jika berhasil, hasilnya adalah file bytecode berakhiran .class.
3. Mengeksekusi bytecode dengan perintah java. Parameter dari perintah ini adalah nama file hasil kompilasi *tanpa* ekstensi .class. Contoh: java HelloWorld.

2.5.6 Source Code

Berikut kode untuk HelloWorld.java:

```
public class HelloWorld
{
   public static void main(String[] args)
   {
      System.out.println("Apa Kabar Dunia?");
   }
}
```

Dan ini sebuah contoh lain, yaitu applet sederhana untuk menampilkan teks di applet. Sebutlah file ini bernama HelloWorldApplet.java:

```
import java.awt.Graphics;
public class HelloWorldApplet extends java.applet.Applet
{
   public void paint(Graphics g)
   {
      g.drawString("Apa Kabar Dunia?", 5, 25);
   }
}
```

Secara gamblang dapat diperhatikan bahwa struktur kedua program sangat mirip, dan hanya berbeda dalam konteks eksekusi. Kedua program ini akan dibahas lebih lanjut setelah kita membahas cara mengkompile dan mengeksekusi program tersebut.

Perlu diingat bahwa bahasa Java bersifat case sensitive, sehingga Anda harus memperhatikan penggunaan huruf besar dan kecil. Selain itu penulisan source code program tidak harus memperhatikan bentuk tertentu, sehingga Anda bisa saja menuliskan semua baris source code tersebut dalam satu baris asal Anda tidak lupa membubuhkan tanda titik koma (;), atau menuliskan tiap kata dalam satu baris tersendiri. Namun dianjurkan Anda mengikuti layout seperti pada contoh agar program Anda mudah dibaca dan dimengerti.

2.5.7 Kompilasi

Setelah kedua file disave dengan nama HelloWorld.java dan HelloWorldApplet.java, kita akan mengkompile kedua program tersebut dengan perintah:

prompt> javac HelloWorld.java

prompt> javac HelloWorldApplet.java

Perlu diperhatikan bahwa direktori aktif Anda saat ini adalah direktori tempat Anda meletakkan file-file program tersebut. Anda tetap dapat mengkompile program Anda dari direktori berbeda dengan perintah:

prompt> javac (direktori program)/namafile.java

Setelah perintah ini selesai, Anda akan melihat bahwa telah tercipta dua buah file .class, yaitu bytecode hasil kompilasi source code kita.

2.5.8 Sintaks Program

Sekarang kita akan mencoba membahas elemen-elemen dalam kedua source code tersebut.

Pada awal Listing 2 kita menemukan perintah import. Pada tahap awal ini Anda perlu mengetahui bahwa pernyataan tersebut hanya berfungsi mempermudah penulisan metode atau dalam bahasa pemrograman lain

disebut prosedur atau fungsi. Jadi Anda hanya perlu menulis Graphics sebagai pengganti java.awt.Graphics, karena kita telah mengimpor java.awt.Graphics.

Kemudian di masing-masing listing terdapat pernyataan public class. Pernyataan ini adalah pernyataan pembuka sebuah kelas. Kelas sendiri digunakan untuk menciptakan objek. Ingat bahwa Java berorientasi objek. Kata public di depannya berfungsi agar kelas tersebut dapat diakses oleh semua program lain. Untuk saat ini anggaplah objek sebagai suatu item yang dapat dimanipulasi oleh sebuah program. Dalam Listing 2 terdapat tambahan kata extends. Hal ini berarti kelas yang kita buat akan mewarisi sifat-sifat dari kelas yang kita extends. Dengan kata lain kita menjadikan kelas yang kita extends sebagai himpunan bagian dari kelas kita buat.

Kemudian kita menemukan baris pernyataan public static void main(String[] args) dan public void paint(Graphics g). Keduanya adalah pernyataan pembuka sebuah metode. Metode sendiri adalah kumpulan pernyataan untuk melakukan suatu tugas tertentu dalam kelas. Keduanya sebenarnya mempunyai fungsi yang sama namun dalam konteks yang berbeda. Dalam setiap aplikasi harus ada sebuah metode yang bernama main yang akan dieksekusi pertama kali saat program tersebut dieksekusi. Sementara dalam applet, metode yang pertama kali akan dieksekusi ketika applet diload adalah paint. Kata public di depannya mempunyai fungsi yang sama dengan kata public yang ada di depan baris permulaan kelas. Namun nantinya Anda akan menemukan juga bentuk lain seperti private dan protect yang akan kita bahas nanti.

Pada Listing 1 terdapat kata static pada pernyataan pembuka metode main. Hal ini berarti metode main tidak mengubah atau menggunakan objek yang diciptakan oleh kelas tersebut, sehingga dapat dikatakan berdiri sendiri dan tidak terikat dengan objek. Dalam metode main dalam aplikasi, parameternya adalah selalu String[] args, di mana args hanyalah sebuah nama dari objek array dari String. Array ini nantinya akan berisi

parameter-parameter yang diberikan user sebagai argumen command line. Sementara Anda tidak perlu mengerti mengenai parameter tersebut, cukup diingat bahwa bentuk metode main harus selalu demikian.

Kemudian di dalam kedua metode pada kedua listing tersebut, kita menemukan sebuah pernyataan. Anda tentu dapat saja meletakkan lebih dari satu pernyataan dalam sebuah metode. Setiap pernyataan dalam sebuah metode dipisahkan oleh titik koma dan akan dieksekusi satu persatu. Kedua pernyataan pada listing ternyata memanggil sebuah metode lain yaitu metode println dan paint. Tentunya dapat Anda perhatikan bahwa untuk memanggil sebuah metode diperlukan tiga komponen yaitu:

1. Objek yang ingin kita pakai. Dalam hal ini objek System.out dan Graphics g.
2. Nama metode yang ingin kita pakai. Dalam hal ini println dan paint.
3. Sepasang tanda kurung yang berisi informasi tambahan yang diperlukan oleh metode yang dipanggil, yaitu parameter.

Dalam Listing 1, pernyataan System.out.println("Apa Kabar Dunia?"); berarti carilah objek out dalam kelas System kemudian panggil metode println dari objek out dengan parameter berupa string "Apa Kabar Dunia?". Sedang dalam Listing 2, pernyataan g.drawString("Apa Kabar Dunia?", 5, 25); berarti carilah objek g kemudian panggil metode drawString pada objek g dengan parameter "Apa Kabar Dunia?", 5, 25);.

2.5.9 Eksekusi

Setelah selesai membahas sintaks dasar Java dalam kedua listing, selanjutnya kita akan mencoba mengeksekusi kedua program ini. Untuk program pertama yang berupa aplikasi biasa, kita tinggal mengetikkan perintah java HelloWorld pada prompt dan pesan Apa Kabar Dunia? akan tampil di layar (atau mungkin di tempat lain, bergantung sistem operasi Anda). Sedangkan untuk applet kita mesti membuat sebuah file HTML sebagai pembungkus—atau pemanggilnya. Berikut diberikan contoh file HTML untuk membungkus applet yang kita buat.

```
<HTML>
 <HEAD>
  <TITLE>Coba Applet</TITLE>
 </HEAD>
 <BODY>
  <APPLET CODE="HelloWorldApplet.class" WIDTH=150
HEIGHT=25>
  </APPLET>
 </BODY>
</HTML>
```

Beri nama helloword.html dan simpanlah di direktori yang sama dengan lokasi file-file .java dan .class sebelumnya. Untuk mengeksekusi applet kita cukup membuka file HTML tersebut di browser yang Java-enabled atau mengetikkan perintah appletviewer namafile.html di prompt.

2.5.10 *Compiler*

Banyak macam-macam *compiler* yang digunakan dalam membuat program java, misal : Gel, Netbeans, Eclipse dll. Disini akan dijelaskan mengenai NetBeans IDE.

NetBeans dimulai pada tahun 1996 sebagai Xelfi (kata bermain di *Delphi*), Java IDE proyek mahasiswa di bawah bimbingan Fakultas Matematika dan Fisika di Universitas Charles di Praha . Pada tahun 1997 membentuk perusahaan Staněk Romawi di sekitar proyek dan menghasilkan versi komersial NetBeans IDE hingga dibeli oleh Sun Microsystems pada tahun 1999. Sun open-source IDE NetBeans pada bulan Juni tahun berikutnya. Sejak itu, komunitas NetBeans terus berkembang. NetBeans IDE 7.0 memperkenalkan dukungan untuk mengembangkan modul IDE dan aplikasi-aplikasi client kaya berdasarkan platform NetBeans, Swing Java GUI builder (sebelumnya dikenal sebagai "Project Matisse"), meningkatkan CVS dukungan, WebLogic 9 dan JBoss 4 dukungan, dan perangkat tambahan banyak

redaksi. NetBeans 6 tersedia dalam repositori resmi distribusi Linux utama.

NetBeans IDE 6.5, dirilis pada bulan November 2008, memperpanjang yang sudah ada Java EE fitur (termasuk dukungan Java Persistence, EJB 3 dan JAX-WS). Selain itu, NetBeans Enterprise Pack mendukung pengembangan aplikasi perusahaan Java EE 5, termasuk SOA alat desain visual, skema XML alat, layanan web orkestrasi (untuk BPEL), dan UML pemodelan. The NetBeans IDE Bundle untuk C / C + + mendukung C / C + + dan FORTRAN pembangunan.

NetBeans IDE 6.8 adalah IDE pertama untuk memberikan dukungan lengkap dari Java EE 6 dan GlassFish Enterprise Server v3 . Pengembang open-source hosting yang mereka proyek-proyek kenai.com tambahan manfaat dari instant messaging dan pelacakan masalah integrasi dan navigasi yang benar dalam IDE, dukungan untuk pengembangan aplikasi web dengan PHP 5.3 dan kerangka Symfony, dan code completion ditingkatkan, layout, petunjuk dan navigasi dalam proyek-proyek JavaFX.

NetBeans IDE 6.9, dirilis pada bulan Juni 2010, menambahkan dukungan untuk OSGi , Spring Framework 3.0, Java EE ketergantungan injeksi (JSR-299), Zend Framework untuk PHP , dan lebih mudah navigasi kode (seperti "ditimpa / Diimplementasikan" penjelasan), format , petunjuk, dan refactoring di beberapa bahasa. NetBeans IDE 7.0 dirilis pada April 2011.

Platform NetBeans adalah dapat digunakan kembali kerangka untuk menyederhanakan pengembangan Java Swing aplikasi desktop. Bundel NetBeans IDE untuk Java SE berisi apa yang diperlukan untuk mulai mengembangkan NetBeans plugin dan aplikasi berbasis NetBeans Platform; tidak ada SDK tambahan diperlukan.

Aplikasi dapat menginstal modul secara dinamis. Setiap aplikasi dapat mencakup modul Update Center untuk mengijinkan pengguna

aplikasi untuk men-download digital-ditandatangani upgrade dan fitur baru secara langsung ke dalam aplikasi yang berjalan. Menginstal ulang upgrade atau rilis baru tidak memaksa pengguna untuk men-download keseluruhan aplikasi lagi.

Platform ini menawarkan layanan dapat digunakan kembali umum untuk aplikasi desktop, memungkinkan pengembang untuk fokus pada logika khusus untuk aplikasi mereka. Di antara fitur dari platform adalah:

- Pengguna antarmuka manajemen (misalnya menu dan toolbar)
- Pengguna pengaturan manajemen
- Manajemen penyimpanan (tabungan dan pemuatan setiap jenis data)
- Jendela manajemen
- Penyihir kerangka kerja (mendukung langkah-demi-langkah dialog)
- NetBeans Visual Perpustakaan
- Tools Pengembangan Terpadu

IDE NetBeans adalah sebuah open source lingkungan pengembangan terintegrasi. NetBeans IDE mendukung pengembangan semua tipe aplikasi Java (Java SE termasuk JavaFX, (Java ME , web , EJB dan ponsel aplikasi) di luar kotak antara fitur-fitur lainnya adalah. Ant berbasis proyek sistem, dukungan Maven, refactorings , kontrol versi (mendukung CVS , Subversion , Mercurial dan ClearCase).

- **Modularitas:** Semua fungsi IDE disediakan oleh modul. Setiap modul menyediakan fungsi yang didefinisikan dengan baik, seperti dukungan untuk bahasa Jawa , mengedit, atau dukungan bagi CVS sistem versioning, dan SVN. NetBeans memuat semua modul yang dibutuhkan untuk pengembangan Java dalam sekali download, memungkinkan pengguna untuk mulai bekerja segera. Modul juga memungkinkan NetBeans untuk diperpanjang. Fitur-fitur baru, seperti dukungan untuk bahasa pemrograman lain, dapat ditambahkan dengan menginstal modul tambahan. Sebagai contoh, Sun Studio , Sun Java Studio Enterprise, dan

Sun Java Studio Creator dari Sun Microsystems semua didasarkan pada NetBeans IDE.

- **Lisensi:** Dari Juli 2006 hingga 2007, NetBeans IDE dilisensikan di bawah Sun Pembangunan Umum dan Lisensi Distribusi (CDDL), lisensi berdasarkan Mozilla Public License (MPL). Pada bulan Oktober 2007, Sun mengumumkan bahwa NetBeans selanjutnya akan ditawarkan di bawah lisensi ganda dari CDDL dan GPL versi 2 lisensi, dengan pengecualian GPL menghubungkan untuk GNU Classpath [7]

NetBeans Profiler adalah alat untuk pemantauan aplikasi Java: Ini membantu pengembang menemukan kebocoran memori dan mengoptimalkan kecepatan. Sebelumnya didownload secara terpisah, itu terintegrasi ke dalam IDE inti sejak versi 6.0.

The Profiler ini berdasarkan sebuah proyek riset Sun Laboratories yang bernama JFluid. Penelitian yang menemukan teknik-teknik khusus yang dapat digunakan untuk menurunkan overhead dari profiling aplikasi Java. Salah satu teknik tersebut adalah bytecode instrumentasi dinamis, yang sangat berguna untuk profiling aplikasi Java yang besar. Menggunakan instrumentasi bytecode dinamis dan algoritma tambahan, Netbeans Profiler mampu mendapatkan informasi runtime aplikasi yang terlalu besar atau kompleks bagi profiler lain. NetBeans juga mendukung Profiling Poin yang memungkinkan Anda profil poin yang tepat dari eksekusi dan mengukur waktu eksekusi.

Pembangun GUI juga memiliki built-in mendukung untuk JSR 296 (swing Application Framework), dan JSR 295 (Beans Binding teknologi). NetBeans JavaScript Editor menyediakan dukungan diperpanjang untuk JavaScript , Ajax, dan CSS . [10] [11]

JavaScript editor fitur terdiri dari sintaks , refactoring , code completion untuk objek asli dan fungsi, generasi JavaScript kerangka kelas, generasi Ajax callback dari template, dan otomatis kompatibilitas browser cek.

Fitur CSS editor terdiri dari code completion untuk nama gaya, navigasi cepat melalui panel navigator, menampilkan deklarasi aturan CSS di List View dan struktur file dalam Tree View, menyortir tampilan outline dengan urutan nama, jenis atau deklarasi (Daftar & Pohon) , menciptakan deklarasi aturan (Pohon saja), refactoring bagian dari nama aturan (Pohon saja). [HTE].

2.5.11 Penelitian Sebelumnya

ENKRIPSI GAMBAR MENGGUNAKAN ALGORITMA SECURE IMAGE PROTECTION. Ini adalah penelitian yang dilakukan oleh Tri Hariyono Reiza Hafidz dan terbatas pada satu jenis gambar saja, oleh karna itu saya mengembangkan dan membuat aplikasi yang baru yang dapat melindungi lebih banyak format gambar.