

**LAPORAN PRAKTIKUM MODUL KE-11
KASUS STEGANOGRAPHY DALAM OPERASI FILE BINARY
DAN OPERASI BITSISE**



**Dosen Pengampu :
I Ketut Purnamawan, S.Kom., M.Kom.**

**Disusun Oleh :
I Gede Gelgel Abdiutama ; 2115101014**

**MATA KULIAH PRAKTIKUM ALGORITMA DAN PEMROGRAMAN
UNIVERSITAS PENDIDIKAN GANESHA
SINGARAJA
TA. 2022**

A. PERMASALAHAN

Buatlah program untuk menyembunyikan pesan teks di dalam file gambar, dan program untuk membaca teks tersembunyi tersebut dari file gambar yang berisi pesan!

Perubahan satu bit atau dua bit paling kanan dari setiap komponen warna (RGB) tidak akan terlalu mengubah warna gambar. Mata manusia tidak cukup mampu untuk membedakan perubahannya. Jadi, satu bit atau dua bit paling kanan dari setiap komponen warna bisa diubah semaunya tanpa menimbulkan perubahan yang bisa dilihat oleh mata manusia. Hal ini bisa dimanfaatkan untuk menyembunyikan teks di dalam file gambar.

Satu karakter terdiri dari 1 byte (8 bit) data. Bit-bit dari karakter ini bisa ditaruh di bit paling kanan dari setiap komponen warna gambar. Jika hanya digunakan 1 bit terakhir saja dari komponen warna, maka untuk menyimpan satu karakter diperlukan 8 komponen warna. Jika digunakan 2 bit terakhir dari komponen warna, maka diperlukan 4 komponen warna untuk menyimpan satu karakter. Gunakan file teks sebagai sumber teks, dan file bmp seperti pada modul sebelumnya sebagai file gambar pembawa pesan.

B. KAJIAN TEORI

1. Bahasa Pemrograman C

Bahasa pemrograman C dibuat pada tahun 1972 oleh Dennis Ritchie untuk Sistem Operasi Unix di Bell Telephone Laboratories. Meskipun C dibuat untuk memprogram sistem dan jaringan komputer, bahasa ini juga sering digunakan dalam mengembangkan software aplikasi. C juga banyak dipakai oleh berbagai jenis platform sistem operasi dan arsitektur komputer, bahkan terdapat beberapa compiler yang sangat populer telah tersedia. C secara luar biasa memengaruhi bahasa populer lainnya, terutama C++ yang merupakan ekstensi dari C. Bahasa C terdiri dari beberapa versi seperti C K&R, ANSI C & ISO C, dan C99.

Contoh program bahasa C:

```
#include <stdio>

int main()
{
    printf("Hello World");
    return 0;
}
```

2. Bahasa Pemrograman C++

Bahasa pemrograman C++ adalah bahasa pemrograman komputer yang di buat oleh Bjarne Stroustrup, yang merupakan perkembangan dari bahasa C dikembangkan di Bell Labs (Dennis Ritchie) pada awal tahun 1970-an, bahasa itu diturunkan dari bahasa B yang ditulis oleh Ken Thompson pada tahun 1970 yang diturunkan dari bahasa sebelumnya yaitu BCL. Pada awalnya, bahasa tersebut dirancang sebagai bahasa pemrograman yang dijalankan pada sistem Unix. Pada perkembangannya, versi ANSI (American National Standards Institute) pada bahasa pemrograman C menjadi versi dominan, meskipun versi tersebut sekarang jarang dipakai dalam pengembangan sistem dan jaringan maupun untuk embedded system. Bjarne Stroustrup pada Bell Labs pertama kali mengembangkan C++ pada awal 1980-an. Untuk mendukung fitur-fitur pada C++, dibangun efisiensi dan support system untuk pemrograman tingkat rendah (low level coding). Pada C++ ditambahkan konsep-konsep baru seperti class dengan sifat-sifatnya seperti inheritance dan overloading. Salah satu perbedaan yang paling mendasar dengan bahasa C adalah dukungan terhadap konsep pemrograman berorientasi objek (object-oriented programming).

Contoh program bahasa C++:

```
#include <iostream>

int main()
{
    std::cout << "Hello World";
    return 0;
}
```

3. Tipe Data

Data types atau tipe data adalah sebuah pengklasifikasian data berdasarkan jenis data tersebut. Tipe data dibutuhkan agar kompiler dapat mengetahui bagaimana sebuah data akan digunakan. Untuk mengembangkan program dalam bahasa pemrograman C atau C++ terdapat berbagai jenis tipe data yang dapat dipilih dan digunakan sesuai dengan kebutuhan dan karakter nilai yang ingin disimpan di dalam variable. Jenis tipe data yang sering digunakan yaitu Boolean, Character, String, Integer, Floating Point, dan Double Floating Point.

a. Boolean

Tipe data Boolean merupakan tipe yang memiliki dua nilai yaitu benar (true) atau salah (false). Nilai yang digunakan pada tipe ini sangat penting dalam mengambil keputusan suatu kejadian tertentu.

b. Character

Tipe data character merupakan salah satu tipe data yang memungkinkan kita untuk memesan memori berformat text (huruf, angka, dan simbol) dengan karakter tunggal. Dibutuhkan 1 byte atau 8 bit ruang di dalam memori agar dapat menyimpan sebuah karakter.

c. String

Tipe data string terdiri dari kumpulan karakter dengan panjang tertentu, dan seringkali dianggap sebagai tipe data dasar. Hal ini dikarenakan hingga saat ini tipe data string paling sering digunakan oleh para programmer.

d. Integer

Jenis tipe data integer dapat didefinisikan sebagai bilangan bulat. Artinya, suatu program yang menggunakan tipe data integer ini tidak mendukung penggunaan huruf. Selain itu, bilangan yang digunakan juga haruslah bulat (tidak mengandung pecahan decimal).

e. Floating Point

Tipe data floating point atau real number merupakan tipe data angka yang memiliki bagian decimal di akhir angka. Tipe data float cocok digunakan untuk variable yang akan berisi angka pecahan.

f. Double Floating Point

Sama halnya dengan floating point, yang bersifat menyatakan bilangan pecahan. Bedanya adalah penyimpangan angka maksimal lebih besar daripada float, otomatis double juga akan membutuhkan memori yang lebih besar.

4. Percabangan

Percabangan adalah sebuah tahap dimana program akan melakukan pengecekan kondisi. Kondisi ini bisa digunakan untuk menentukan bagian program/statement mana yang akan dijalankan jika kondisi tertentu terpenuhi. Di dalam bahasa C, kita dapat membuat seleksi dengan if else.

a. if

Pernyataan if : “Jika kondisi bernilai benar, maka perintah akan dikerjakan dan jika tidak memenuhi syarat maka akan diabaikan.”

```
if(kondisi){  
    //statement  
}
```

b. if else

Pernyataan if else : “Jika kondisi bernilai benar, maka perintah-1 akan dikerjakan dan jika tidak memenuhi syarat maka akan mengerjakan perintah-2”

```
if(kondisi){  
    //statement  
}else{  
    //statement  
}
```

5. Operasi File

Penggunaan operasi FILE dalam sebuah program sangat dibutuhkan dalam pembuatan program yang sesungguhnya. Membuat operasi file sebagai tempat penampung data-data selama operasi program. Jika menggunakan variabel biasa, data yang biasanya diolah program hanya akan tersimpan sementara dalam memory komputer dan akan hilang ketika program close seperti sifat dasar pada RAM. Berbeda dengan memory, penyimpanan data berbasis file akan tersimpan terus walaupun program telah diclose maupun komputer telah di shutdown. Pada dasarnya, operasi file terbagi 3 jenis. Read (R), Write(W), dan Append(A). Read artinya membaca isi file, write menulis data ke file dan append menambahkan data ke baris terakhir dalam file. Adapun beberapa mode didalam operasi file yang dapat digunakan yaitu sebagai berikut :

MODE	KETERANGAN
r	Untuk membuka file teks dalam mode membaca
w	Untuk membuka atau membuat file teks dalam mode penulisan
a	Untuk membuka file teks dalam mode tambahkan
r+	Untuk membuka file teks dalam mode membaca dan menulis
w+	Untuk membuka file teks dalam mode membaca dan menulis
a+	Untuk membuka file teks dalam mode membaca dan menulis
rb	Untuk membuka file biner dalam mode membaca
wb	Untuk membuka atau membuat file biner dalam mode penulisan
ab	Untuk membuka file biner dalam mode tambahkan
rb+	Untuk membuka file biner dalam mode membaca dan menulis
wb+	Untuk membuka file biner dalam mode membaca dan menulis
ab+	Untuk membuka file biner dalam mode membaca dan menulis

6. Ruang Warna RGB

Ruang warna RGB adalah ruang warna tambahan apa pun yang didasarkan pada model warna RGB. Ruang warna RGB ditentukan oleh koordinat kromatisitas warna primer aditif merah, hijau, dan biru, titik putih yang biasanya merupakan iluminan standar, dan fungsi transfer yang juga dikenal sebagai kurva respons nada (TRC) atau gamma. Menerapkan hukum Grassmann aditivitas cahaya, ruang warna yang ditentukan dapat menghasilkan warna yang diapit oleh segitiga 2D pada diagram kromatisitas yang ditentukan oleh koordinat primer tersebut. TRC dan titik putih selanjutnya menentukan kemungkinan warna, menciptakan volume dalam bentuk 3D yang tidak pernah melebihi batas segitiga. Mata manusia normal mengandung tiga jenis sel fotosensitif yang disebut kerucut, yang peka terhadap panjang gelombang cahaya yang umumnya kita kategorikan sebagai merah, hijau, dan biru. Ruang warna RGB menggunakan primer yang diterangi yang dipilih untuk merangsang setiap jenis kerucut sebebaskan mungkin. Dengan cara ini, pencampuran tiga lampu dalam proporsi yang berbeda dapat merangsang kerucut di mata dan menciptakan persepsi warna.

7. File BMP

Format file BMP, juga dikenal sebagai file gambar bitmap, format file bitmap device independent bitmap(DIB) dan bitmap, adalah format file gambar grafik raster yang digunakan untuk menyimpan gambar digital bitmap, terlepas dari perangkat tampilan (seperti adaptor grafik), terutama pada sistem operasi Microsoft Windows dan OS/2. Format file BMP mampu menyimpan gambar digital dua dimensi baik monokrom maupun berwarna, dalam berbagai kedalaman warna, dan opsional dengan kompresi data, saluran alfa, dan profil warna. Spesifikasi Windows Metafile (WMF) mencakup format file BMP. Adapun header dari file Bitmap ini yaitu sebagai berikut :

Offset hex	Offset dec	Size	Purpose
00	0	2 bytes	The header field used to identify the BMP and DIB file is 0x42 0x4D in hexadecimal , same as BM in ASCII. The following entries are possible: BM Windows 3.1x, 95, NT, ... etc. BA OS/2 struct bitmap array CI OS/2 struct color icon CP OS/2 const color pointer IC OS/2 struct icon PT OS/2 pointer
02	2	4 bytes	The size of the BMP file in bytes
06	6	2 bytes	Reserved; actual value depends on the application that creates the image, if created manually can be 0
08	8	2 bytes	Reserved; actual value depends on the application that creates the image, if created manually can be 0
0A	10	4 bytes	The offset, i.e. starting address, of the byte where the bitmap image data (pixel array) can be found.

8. Steganography

Kata steganografi (steganography) berasal dari bahasa Yunani yaitu Steganos yang artinya tersembunyi, atau terselubung. Dan Graphein yang artinya menulis, sehingga kurang lebih artinya adalah menulis tulisan yang tersembunyi atau terselubung. Teknik ini meliputi banyak sekali metode komunikasi untuk menyembunyikan pesan rahasia. Metode ini termasuk tinta yang tidak tampak, microdots, pengaturan kata, tanda tangan digital, jalur tersembunyi dan komunikasi spektrum lebar. Atau dalam bahasa umum, steganografi adalah teknik menyembunyikan data atau pesan rahasia (hiding message) didalam media penampung berupa citra digital sehingga keberadaan (eksistensi) data rahasia tersebut tidak terdeteksi oleh indera penglihatan manusia. Pada komputer, gambar adalah suatu array dari bilangan yang merepresentasikan intensitas terang pada point yang bervariasi (pixel). Pixel ini menghasilkan raster data gambar. Suatu ukuran

gambar yang umum adalah 640 x 480 pixel dan 256 warna (8 bit per pixel), suatu gambar akan berisi kira-kira 300 kilobit data. Gambar digital disimpan juga secara khusus di dalam file 24-bit atau 8-bit. Gambar 24-bit menyediakan lebih banyak ruang untuk menyembunyikan informasi, bagaimanapun itu sudah sangat besar. Semua variasi warna untuk pixel yang diperoleh dari tiga warna dasar yaitu warna merah, hijau dan biru. Setiap warna dasar direpresentasikan dengan 1 byte, gambar 24-bit menggunakan 3 byte per pixel untuk merepresentasikan suatu nilai warna. 3 byte ini dapat direpresentasikan sebagai nilai hexadecimal, decimal dan biner. Dalam banyak halaman web, warna latar belakang direpresentasikan dengan bilangan 6 digit hexadecimal, yang aktualnya tiga ikatan merepresentasikan merah, hijau dan biru. Latar belakang putih akan mempunyai nilai FFFFFFFF: 100% merah (FF), 100% hijau (FF) dan 100% biru (FF). Nilai decimalnya 255, 255, 255 dan binernya adalah 11111111, 11111111, 11111111 yang adalah tiga byte yang menghasilkan putih. Definisi latar belakang putih adalah analog dengan definisi warna dari pixel tunggal dalam suatu gambar. Pixel merepresentasikan kontribusi pada ukuran file. Untuk contoh, misalkan kita mempunyai suatu gambar 24-bit luasnya 1,024 pixel dengan ketinggian 768 pixel, yang merupakan resolusi umum untuk grafik beresolusi tinggi. Suatu gambar mempunyai lebih dari dua juta pixel, masing-masing mempunyai definisi yang akan menghasilkan suatu kelebihan file 2 Mbyte. Karena gambar 24-bit masih relatif tidak umum pada internet, ukuran seperti ini akan menarik perhatian selama transmisi. Kompresi file akan menguntungkan, jika tidak perlu transmisi file seperti itu. Steganografi pada media digital file gambar digunakan untuk mengeksploitasi keterbatasan kekuatan sistem penglihatan manusia dengan cara menurunkan kualitas warna pada file gambar yang belum disisipi pesan rahasia. Sehingga dengan keterbatasan tersebut manusia sulit menemukan gradasi penurunan kualitas warna pada file gambar yang telah disisipi pesan rahasia.

C. PEMBAHASAN

Dalam pembahasan ini, akan menjelaskan proses atau tahapan yang dilakukan pada saat melakukan praktikum membuat program untuk mengolah gambar dengan format file BMP Windows 24 bit. Program dapat membuat gambar yang lebih terang dari gambar aslinya. Berikut bagian-bagian dari program yang telah dibuat :

1. Header

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

using namespace std;
```

Pada program ini menggunakan bahasa pemrograman C++, yang dimana menggunakan standar library C++ dan library conio.h digunakan untuk bisa menggunakan perintah getch(); didalam program untuk membuat program lebih dinamis agar mudah untuk digunakan.

2. Tipe Data & Function

```
void header(FILE *bmp, FILE *copy_bmp, unsigned int* width); //menyalin header bmp
void menu(); //pilihan menu program
bool cek(string fname); //mengecek file
unsigned char mergebit(char bmp, char text2bit); //memasukkan 2bit txt kedalam byte pixel
unsigned char merge4byte(unsigned char txt_byte[]); //menyatukan 2bit akhir dari 4 byte
void encode(string txt_file, string bmp_file); //memasukkan pesan kedalam pixel file bmp
void decode(string pesan_bmp); //membaca pesan yang tersembunyi pada file bmp
int pilih;
```

Dalam program ini dibuat beberapa function sehingga mudah untuk dieksekusi pada main program. Tipe data yang saya gunakan yaitu ada yang menggunakan char, string, unsigned char dan int. Penggunaan tipe data yang unsigned bertujuan untuk mendapatkan nilai yang tidak (-) minus. Karena didalam proses program nanti agar dapat menghasilkan nilai dari 0 – 255 sesuai rentang RGB. Berikut adalah tabel rentang nilai antara signed char dan unsigned char :

Tipe data	Rentang Nilai
signed char	-128 hingga 127
unsigned char	0 hingga 255

3. Main Program

```
int main(){
    awal:
    menu();
    switch(pilih){
        case 1:{
            char str[30];
            cout << "Nama file BMP (.bmp) : ";
            scanf("%s", str);
            string bmp_file(str);

            cout << "Nama file TXT (.txt) : ";
            scanf("%s", str);
            string txt_file(str);

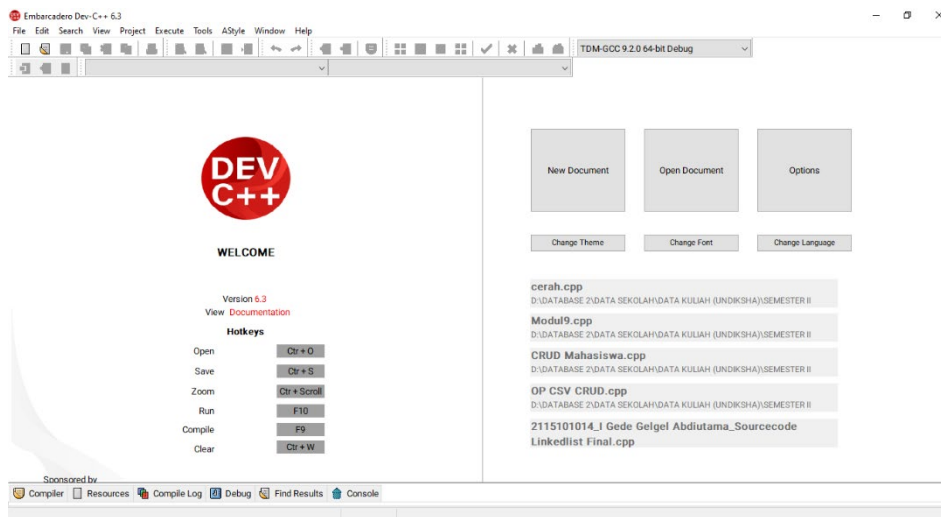
            if (cek(bmp_file)){
                if (cek(txt_file)){
                    encode(txt_file, bmp_file);
                    cout << "\nTekan ENTER untuk kembali ke menu...";
                    getch();
                    system("cls");
                    goto awal;
                }
                else{
                    cout << "Maaf File " << txt_file << " tidak ditemukan";
                    cout << "\n\nTekan ENTER untuk mengulang...";
                    getch();
                    system("cls");
                    goto awal;
                }
            }
            else{
                cout << "Maaf File " << bmp_file << " tidak ditemukan";
                cout << "\n\nTekan ENTER untuk mengulang...";
                getch();
                system("cls");
                goto awal;
            }
            break;
        }
        case 2:{
            char str[30];
            cout << "Nama file BMP (.bmp) : ";
            scanf("%s", str);
            string bmp_file(str);

            if (cek(bmp_file)){
                cout << "Pesan Pada Gambar BMP : " << endl;
                decode(bmp_file);
                cout << "\n\nTekan ENTER untuk kembali ke menu...";
                getch();
                system("cls");
                goto awal;
            }
            else{
                cout << "Maaf File " << bmp_file << " tidak ditemukan";
                cout << "\n\nTekan ENTER untuk mengulang...";
                getch();
                system("cls");
                goto awal;
            }
            break;
        }
        case 3:{
            system("cls");
            int exit;
            system("cls");
            cout << "Apakah Anda Yakin Keluar Dari Program Ini? (Y/N) : ";
            scanf("%s", &exit);
            if(exit=='y' || exit=='Y')
            {
                system("cls");
                cout << "Terimakasih telah menggunakan program ini";
            }
            else
            {
                system("cls");
                goto awal;
            }
            return 0;
        }
        default:{
            cout << "Maaf perintah tidak valid!" << endl;
            cout << "\nTekan ENTER untuk kembali ke menu...";
            getch();
            system("cls");
            goto awal;
            break;
        }
    }
}
```

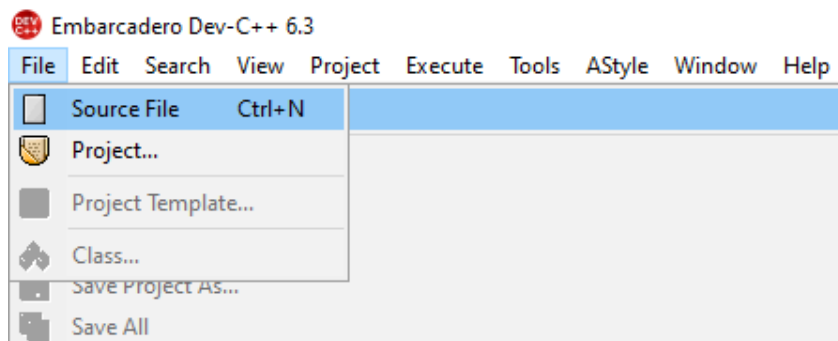
Didalam main program ini adalah isi dari program yang dapat mengeksekusi perintah Encode(Membuat/menyisipkan pesan teks yang berformat txt pada gambar BMP) dan juga dapat melakukan perintah Decode(Membaca pesan teks tersembunyi pada file BMP).

Berikut Langkah-langkah praktikum

1. Pertama, kita buka terlebih dahulu Compiler yang digunakan. Sebagai contoh disini saya menggunakan Dev C++ sebagai compiler nya.



2. Setelah terbuka, kita buat source file dengan cara memilih pada File > New > Source File, atau bisa juga dengan menggunakan shortcut pada keyboard yaitu CTRL + N.

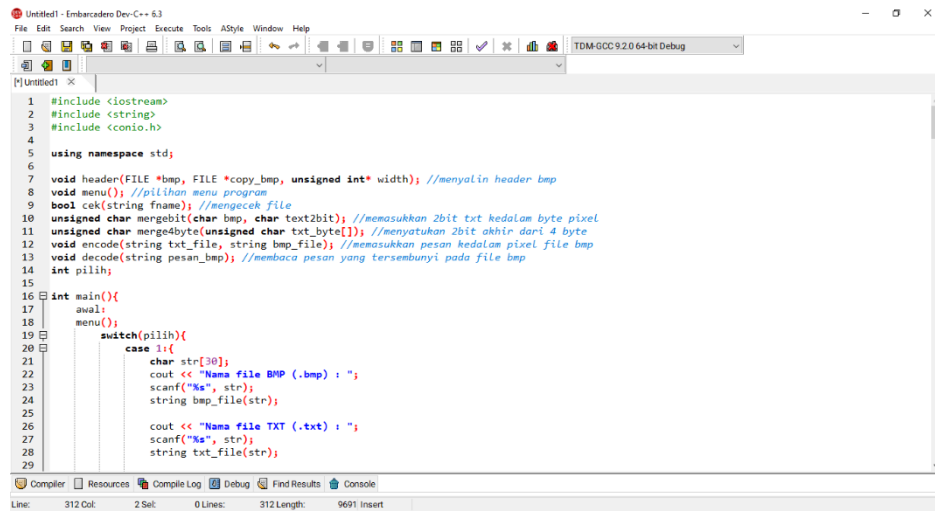


3. Kemudian kita akan eksekusi kode programnya, karena saya menggunakan bahasa pemrograman C++, maka kita membutuhkan sebuah kerangka dasarnya untuk bisa membentuk kode program lainnya, berikut merupakan header dan main program dari bahasa C

```
#include <iostream>

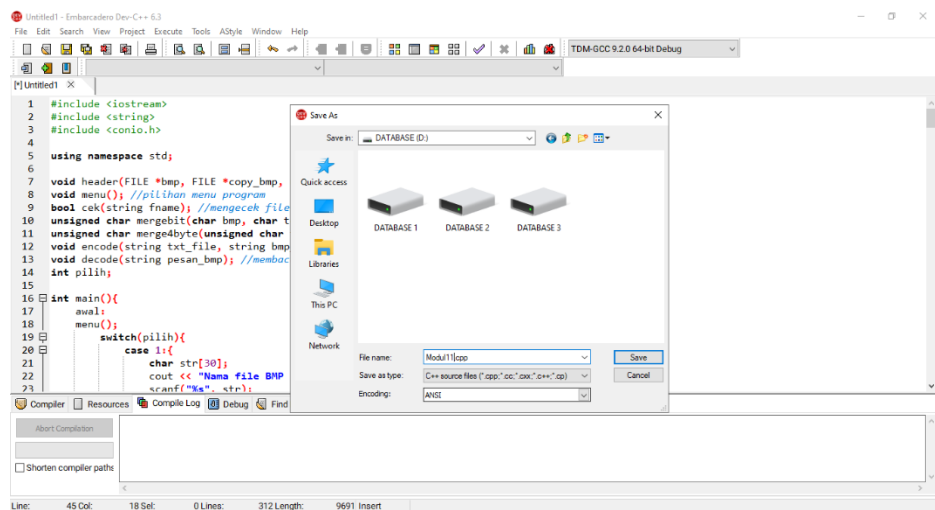
int main()
{
    std::cout << "Hello World";
    return 0;
}
```

4. Pada tahap ini, kita bisa melanjutkan proses pembuatan program dengan melakukan koding sesuai dengan kode program yang akan dibuat. Untuk kode program akan dilampirkan pada halaman Lampiran paling akhir laporan ini.

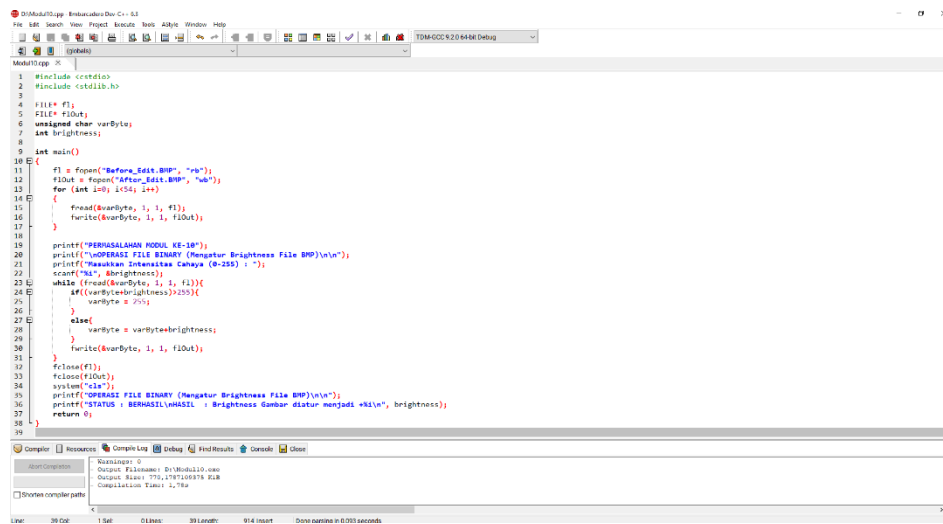


```
1 #include <iostream>
2 #include <string>
3 #include <conio.h>
4
5 using namespace std;
6
7 void header(FILE *bmp, FILE *copy_bmp, unsigned int* width); //menyalin header bmp
8 void menu(); //pilih menu program
9 bool cek(string fname); //mengecek file
10 unsigned char mergebit(char bmp, char text2bit); //memasukkan 2bit txt kedalam byte pixel
11 unsigned char mergebyte(unsigned char txt_byte[]); //menyatakan 2bit akhir dari 4 byte
12 void encode(string txt_file, string bmp_file); //memasukkan pesan kedalam pixel file bmp
13 void decode(string pesan_bmp); //membaca pesan yang tersembunyi pada file bmp
14 int pilih;
15
16 int main(){
17     awal;
18     menu();
19     switch(pilih){
20     case 1:{
21         char str[30];
22         cout << "Nama file BMP (.bmp) : ";
23         scanf("%s", str);
24         string bmp_file(str);
25
26         cout << "Nama file TXT (.txt) : ";
27         scanf("%s", str);
28         string txt_file(str);
29     }
```

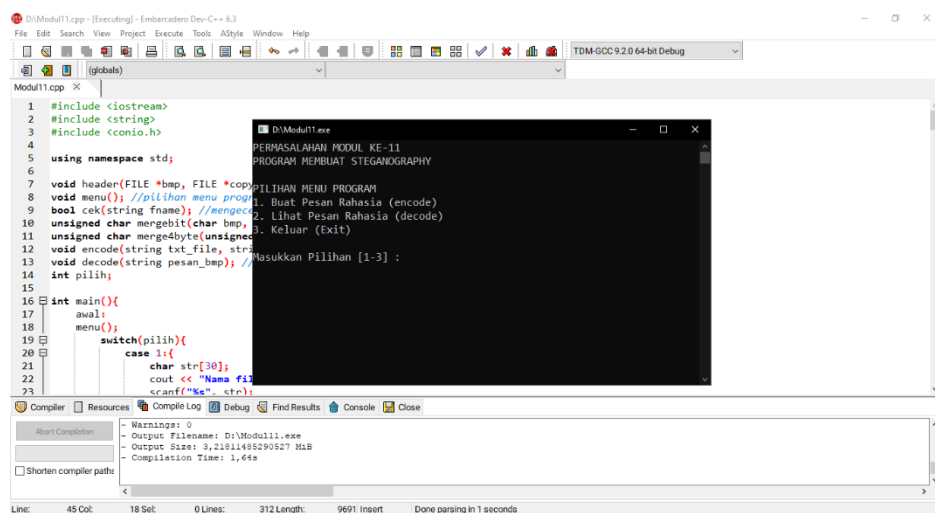
5. Untuk melakukan compile sehingga program dapat dijalankan, kita klik Execute lalu pilih Compile atau bisa juga dengan menggunakan shortcut pada keyboard yaitu F9. Berikutnya akan muncul tampilan yang meminta kita memilih lokasi untuk menyimpan file cpp sebelum di compile. Sebagai contoh disini akan saya simpan dengan nama Modul11.cpp.



6. Maka setelah itu hasil dari program yang telah kita buat akan di compile secara otomatis, perlu diperhatikan jika Errors 0 dan Warnings 0 maka program siap untuk di Run dengan cara klik Execute lalu pilih Run menekan tombol F10.



7. Ketika program sudah di run, maka langkah selanjutnya kita akan mengeksekusi program kita dengan cara melakukan pengetesan atau pengujian Encode (Membuat pesan rahasia pada gambar BMP) lalu mencoba melakukan Decode (Melihat isi pesan rahasia yang ada pada gambar BMP).



D. KESIMPULAN

Pembuatan program Steganography untuk menyembunyikan pesan teks yang berformat txt di dalam file gambar BMP, dan program untuk membaca teks tersembunyi dari file gambar BMP yang berisi pesan ini menggunakan bahasa C++. Perubahan satu bit atau dua bit paling kanan dari setiap komponen warna (RGB) tidak akan terlalu mengubah warna gambar. Satu karakter terdiri dari 1 byte (8 bit) data. Bit-bit dari karakter ini bisa ditaruh di bit paling kanan dari setiap komponen warna gambar. Jika hanya digunakan 1 bit terakhir saja dari komponen warna, maka untuk menyimpan satu karakter diperlukan 8 komponen warna. Jika digunakan 2 bit terakhir dari komponen warna, maka diperlukan 4 komponen warna untuk menyimpan satu karakter. Program ini dibuat sesederhana mungkin agar pengguna dapat menggunakannya dengan mudah dan menghasilkan hasil gambar yang maksimal.

E. LAMPIRAN

Kode Program:

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

using namespace std;

void header(FILE *bmp, FILE *copy_bmp, unsigned int* width); //menyalin header bmp
void menu(); //pilihan menu program
bool cek(string fname); //mengecek file
unsigned char mergebit(char bmp, char text2bit); //memasukkan 2bit txt kedalam byte
pixel
unsigned char merge4byte(unsigned char txt_byte[]); //menyatukan 2bit akhir dari 4
byte
void encode(string txt_file, string bmp_file); //memasukkan pesan kedalam pixel file
bmp
void decode(string pesan_bmp); //membaca pesan yang tersembunyi pada file bmp
int pilih;

int main(){
    awal:
    menu();
    switch(pilih){
        case 1:{
            char str[30];
            cout << "Nama file BMP (.bmp) : ";
            scanf("%s", str);
            string bmp_file(str);

            cout << "Nama file TXT (.txt) : ";
            scanf("%s", str);
            string txt_file(str);

            if (cek(bmp_file)){
                if (cek(txt_file)){
                    encode(txt_file, bmp_file);
                    cout << "\nTekan ENTER untuk kembali ke menu...";
                    getch();
                    system("cls");
                    goto awal;
                }
            }
            else{
                cout << "Maaf File " << txt_file << " tidak ditemukan";
                cout << "\n\nTekan ENTER untuk mengulang...";
                getch();
                system("cls");
                goto awal;
            }
        }
        else{
            cout << "Maaf File " << bmp_file << " tidak ditemukan";
            cout << "\n\nTekan ENTER untuk mengulang...";
            getch();
            system("cls");
            goto awal;
        }
    }
    break;
}
case 2:{
    char str[30];
    cout << "Nama file BMP (.bmp) : ";
    scanf("%s", str);
```

```

        string bmp_file(str);

        if (cek(bmp_file)){
            cout << "Pesan Pada Gambar BMP :" << endl;
            decode(bmp_file);
            cout << "\n\nTekan ENTER untuk kembali ke menu...";
            getch();
            system("cls");
            goto awal;
        }
        else{
            cout << "Maaf File " << bmp_file << " tidak ditemukan";
            cout << "\n\nTekan ENTER untuk mengulang...";
            getch();
            system("cls");
            goto awal;
        }
        break;
    }
    case 3:{
        system("cls");
        int exit;
        system("cls");
        cout << "Apakah Anda Yakin Keluar Dari Program Ini? (Y/N) : ";
        scanf ("%s", &exit);
        if(exit=='y' || exit=='Y')
        {
            system("cls");
            cout << "Terimakasih telah menggunakan program ini";

        }
        else
        {
            system("cls");
            goto awal;
        }
        return 0;
    }
    default:{
        cout << "Maaf perintah tidak valid!" << endl;
        cout << "\nTekan ENTER untuk kembali ke menu...";
        getch();
        system("cls");
        goto awal;
    }
    break;
}
}
}

void header(FILE *bmp, FILE *copy_bmp, unsigned int* width){
    unsigned char data;
    for(int i = 0; i <18 ; i++){
        fread(&data, sizeof(char), 1, bmp);
        fwrite(&data, 1 , sizeof(char) , copy_bmp );
    }
    fread(width, sizeof(int), 1, bmp);
    fwrite(width, 1 , sizeof(int) , copy_bmp );
    for(int i = 0; i <32 ; i++){
        fread(&data, sizeof(char), 1, bmp);
        fwrite(&data, 1 , sizeof(char) , copy_bmp );
    }
}

void menu(){
    cout << "PERMASALAHAN MODUL KE-11\n";
    cout << "PROGRAM MEMBUAT STEGANOGRAPHY\n\n";
    cout << "PILIHAN MENU PROGRAM\n";
}

```



```

    cout << "1. Buat Pesan Rahasia (Encode)\n";
    cout << "2. Lihat Pesan Rahasia (Decode)\n";
    cout << "3. Keluar (Exit)\n\n";
    cout << "Masukkan Pilihan [1-3] : ";
    cin >> pilih;
}

bool cek(string fname){
    const char *filename = &fname[0];
    FILE *file;
    if (file = fopen(filename, "r")){
        fclose(file);
        return true;
    }
    else{
        return false;
    }
}

unsigned char mergebit(char bmp, char text2bit){
    unsigned char hasil_byte = 0;
    unsigned char uji_byte = 1;

    for(int i = 0; i < 2; i++){
        if ((text2bit & uji_byte) == uji_byte){
            hasil_byte += uji_byte;
        }
        uji_byte <<= 1;
    }
    for(int i = 0; i < 6; i++){
        if ((bmp & uji_byte) == uji_byte){
            hasil_byte += uji_byte;
        }
        uji_byte <<= 1;
    }
    return hasil_byte;
}

unsigned char merge4byte(unsigned char txt_byte[]){
    unsigned char tambah_byte = 1;
    unsigned char hasil_byte = 0;
    for (int i = 0 ; i < 4 ; i++){
        unsigned char uji = 1;
        if ((txt_byte[i] & uji) == uji){
            hasil_byte += tambah_byte;
        }
        uji <<= 1;
        tambah_byte <<= 1;
        if ((txt_byte[i] & uji) == uji){
            hasil_byte += tambah_byte;
        }
        tambah_byte <<= 1;
    }
    return hasil_byte;
}

void encode(string txt_file, string bmp_file){
    FILE *input_bmp, *input_text, *output_bmp;

    const char *file = &bmp_file[0];
    input_bmp = fopen(file, "rb");

    file = &txt_file[0];
    input_text = fopen(file, "rb");

    bmp_file = "Encode_"+bmp_file;

```

```

file = &bmp_file[0];
output_bmp = fopen(file, "wb");

unsigned int width;
unsigned char data;
unsigned char padding;
char data_text;
header(input_bmp, output_bmp, &width);
int padding_size = width % 4;
unsigned char text2bit[width*3];
bool text_left = true;
int last_row;

while (true){
    if (!(feof(input_text))){
        for (int i = 0 ; i < width*3; i += 4){
            fread(&data_text, sizeof(char), 1, input_text);
            if (feof(input_text)){
                break;
            }
            unsigned char testbit = 1;
            for(int j = i ; j < 4 + i; j++ ){
                unsigned char lastbit = 0;
                if ((data_text & testbit) == testbit){
                    lastbit += 1;
                }
                testbit <<= 1;
                if ((data_text & testbit) == testbit){
                    lastbit += 2;
                }
                testbit = testbit << 1;
                text2bit[j] = lastbit;
                last_row = j;
            }
        }
    }
    if (!(feof(input_bmp))){
        for (int i = 0 ; i < 3*width; i++){
            fread(&data, sizeof(char), 1, input_bmp);
            if (feof(input_bmp)){
                break;
            }
            if (text_left){
                unsigned char write;
                write = mergebit(data, text2bit[i]);
                fwrite(&write, 1, sizeof(char), output_bmp );
                if (feof(input_text)&& i ==last_row){
                    text_left = false;
                    unsigned char ETX[4] = {3, 0, 0, 0};
                    for( int j = 0; j < 4; j++){
                        fread(&data, sizeof(char), 1, input_bmp);
                        write = mergebit(data, ETX[j]);

                        fwrite(&write, 1, sizeof(char), output_bmp );
                    }
                    i +=4;
                }
            }
            else{
                fwrite(&data, 1, sizeof(char), output_bmp);
            }
        }
        for(int i = 0; i< padding_size; i++){
            fread(&padding, sizeof(char), 1, input_bmp);
            fwrite(&padding, 1 ,sizeof(char), output_bmp);
        }
    }
}

```

```

    }
    else {
        break;
    }
}
fclose(input_bmp);
fclose(input_text);
fclose(output_bmp);
cout << "BERHASIL" << endl;
cout << "Hasil encode : " << bmp_file << endl;
}

void decode(string pesan_bmp){
    FILE *msg, *output_text;

    const char* file = &pesan_bmp[0];
    msg = fopen(file, "rb");

    pesan_bmp = "Decode_Teks.txt";
    file = &pesan_bmp[0];
    output_text = fopen(file, "w");

    unsigned char OneByteText[4];
    unsigned int width;
    bool text_left = true;

    FILE *tmp_header = fopen("tmp", "w");
    header(msg, tmp_header, &width);
    fclose(tmp_header);
    remove("tmp");
    int padding_size = width % 4;
    char padding ;

    while((!feof(msg) )&& (text_left)){
        for (int i = 0; i < width*3; i += 4){
            for (int j = 0; j < 4; j++){
                fread(&OneByteText[j], 1, sizeof(char), msg);
            }
            unsigned char write = merge4byte(OneByteText);
            if (write == 3){
                text_left = false;
                break;
            }
            if (write == 13){
                continue;
            }
            cout<<write;
            fwrite(&write, sizeof(char), 1, output_text);
        }
        for(int i = 0; i < padding_size; i++){
            fread(&padding, 1, sizeof(char), msg);
        }
    }
    fclose(msg);
    fclose(output_text);
}

```

Proses Run pada Command Prompt :

- Tampilan awal Program

```
D:\Modul11.exe
PERMASALAHAN MODUL KE-11
PROGRAM MEMBUAT STEGANOGRAPHY

PILIHAN MENU PROGRAM
1. Buat Pesan Rahasia (Encode)
2. Lihat Pesan Rahasia (Decode)
3. Keluar (Exit)

Masukkan Pilihan [1-3] : █
```

- Tampilan Proses Encode

```
D:\Modul11.exe
PERMASALAHAN MODUL KE-11
PROGRAM MEMBUAT STEGANOGRAPHY

PILIHAN MENU PROGRAM
1. Buat Pesan Rahasia (Encode)
2. Lihat Pesan Rahasia (Decode)
3. Keluar (Exit)

Masukkan Pilihan [1-3] : 1
Nama file BMP (.bmp) : Gambar.bmp
Nama file TXT (.txt) : Pesan.txt
BERHASIL
Hasil encode : Encode_Gambar.bmp

Tekan ENTER untuk kembali ke menu... █
```

- Tampilan Proses Decode

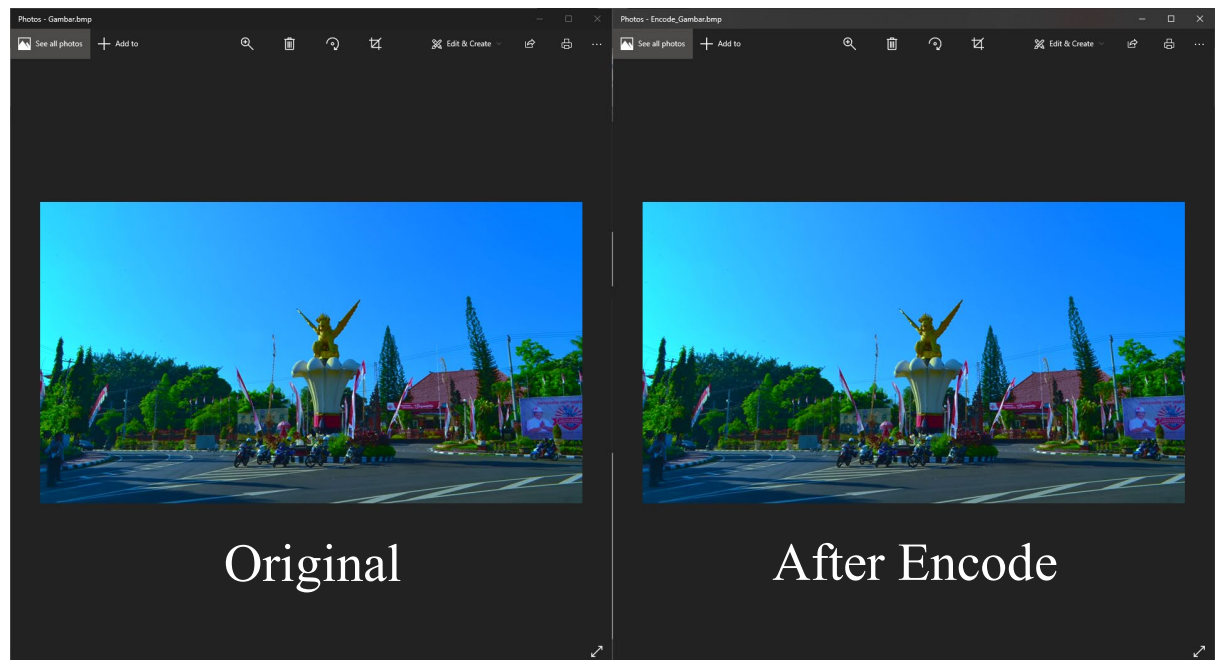
```
D:\Modul11.exe
PERMASALAHAN MODUL KE-11
PROGRAM MEMBUAT STEGANOGRAPHY

PILIHAN MENU PROGRAM
1. Buat Pesan Rahasia (Encode)
2. Lihat Pesan Rahasia (Decode)
3. Keluar (Exit)

Masukkan Pilihan [1-3] : 2
Nama file BMP (.bmp) : Encode_Gambar.bmp
Pesan Pada Gambar BMP :
Hello World,
saya I GEDE GELGEL ABDIUTAMA!
Kode Rahasia 2115101014

Tekan ENTER untuk kembali ke menu... █
```

- File Gambar BMP



- File Pesan Teks

