

Steganografia w pliku .BMP

Wygenerowano przez Doxygen 1.8.18



<b>1 Projekt - steganografia w pliku BMP</b>	<b>1</b>
1.1 Założenia	1
1.2 Limitacje	1
1.3 Użycie	1
<b>2 Indeks struktur danych</b>	<b>3</b>
2.1 Struktury danych	3
<b>3 Indeks plików</b>	<b>5</b>
3.1 Lista plików	5
<b>4 Dokumentacja struktur danych</b>	<b>7</b>
4.1 Dokumentacja struktury _BITFILEDS	7
4.1.1 Opis szczegółowy	7
4.1.2 Dokumentacja pól	7
4.1.2.1 alphaMask	7
4.1.2.2 blueMask	7
4.1.2.3 greenMask	8
4.1.2.4 redMask	8
4.1.2.5 xMask	8
4.2 Dokumentacja struktury _BMPHEADER	8
4.2.1 Opis szczegółowy	8
4.2.2 Dokumentacja pól	8
4.2.2.1 HEADER	9
4.2.2.2 OFFSET	9
4.2.2.3 RESERVED1	9
4.2.2.4 RESERVED2	9
4.2.2.5 SIZE	9
4.3 Dokumentacja struktury _DIBHEADER	9
4.3.1 Opis szczegółowy	10
4.3.2 Dokumentacja pól	10
4.3.2.1 BITSPERPIXEL	10
4.3.2.2 COLORPLANES	10
4.3.2.3 COLORS	10
4.3.2.4 COMPRESSION	10
4.3.2.5 DIBSIZE	10
4.3.2.6 HEIGHT	10
4.3.2.7 HORIZONTALRES	11
4.3.2.8 IMPORTANTCOLORS	11
4.3.2.9 SIZE	11
4.3.2.10 VERTICALRES	11
4.3.2.11 WIDTH	11
4.4 Dokumentacja struktury _SGHEADER	11

---

4.4.1 Opis szczegółowy . . . . .	12
4.4.2 Dokumentacja pól . . . . .	12
4.4.2.1 FILESIZE . . . . .	12
4.4.2.2 HEADER . . . . .	12
<b>5 Dokumentacja plików</b>	<b>13</b>
5.1 Dokumentacja pliku src/functions.c . . . . .	13
5.1.1 Opis szczegółowy . . . . .	14
5.1.2 Dokumentacja funkcji . . . . .	14
5.1.2.1 check_for_stegano() . . . . .	14
5.1.2.2 GCD() . . . . .	14
5.1.2.3 LCM() . . . . .	15
5.1.2.4 pixelArray_read_16bit() . . . . .	15
5.1.2.5 pixelArray_read_24bit() . . . . .	16
5.1.2.6 pixelArray_read_32bit() . . . . .	16
5.1.2.7 prepare_data_to_write() . . . . .	16
5.1.2.8 read_data_from_steganofile_16bit() . . . . .	17
5.1.2.9 read_data_from_steganofile_24_32bit() . . . . .	17
5.1.2.10 read_file_to_memory() . . . . .	18
5.1.2.11 read_headers() . . . . .	18
5.1.2.12 write_data_16bit() . . . . .	19
5.1.2.13 write_data_24bit() . . . . .	19
5.1.2.14 write_data_32bit() . . . . .	20
5.1.2.15 write_data_array_to_file() . . . . .	20
5.1.2.16 write_data_from_image() . . . . .	21
5.1.2.17 write_data_to_image() . . . . .	21
5.1.2.18 write_headers() . . . . .	22
<b>Indeks</b>	<b>23</b>

# Rozdział 1

## Projekt - steganografia w pliku BMP

### 1.1 Założenia

Projekt steganografia w pliku BMP polegał na napisaniu programu, który umieszczał w pliku BMP dowolny plik podany przez użytkownika. Program potrafi "ukrywać" jedynie pliki wielkości do 65kB w plikach BMP o głębi kolorów wyższej niż 8 (16, 24, 32).

### 1.2 Limitacje

Program korzysta wyłącznie z obrazów o głębi kolorów wyższej niż 8, ponieważ dla głębi kolorów 8 oraz niżej, obrazy BMP korzystają z tablicy kolorów, gdzie każdy kolor jest jawnie wpisany i każda zmiana nawet jednego bitu bardzo znacząco wpływa na zmianę koloru danego pixela.

Program ukrywa pliki nie większe niż 65kB, ponieważ wykorzystuje puste miejsca w nagłówkach plików BMP do przechowywania wielkości ukrytego pliku, niestety do wykorzystania jest tam jedynie 2 bajty, co pozwala na zapisanie maksymalnej wielkości pliku właśnie 65kB.

### 1.3 Użycie

Po sklonowaniu repozytorium należy skompilować program korzystając z narzędzia `make`. Po wywołaniu `make` w głównym katalogu projektu powinien utworzyć się podkatalog `bin`. Tam znajduje się plik wywoływalny programu `main`, w przypadku korzystania z systemu MS Windows plik wykonywalny `main.exe`.

**Do kompilacji projektu wymagane jest posiadanie kompilatora GCC w standardzie przynajmniej C89 oraz narzędzie `make`**

Program można wywołać z flagą `-h`, wtedy wyświetli się poniższa pomoc:

```
Pierwszy argument wywołania programu powinien zawierać następujące flagi: -w lub -r
Flaga -w służy do zapisywania danych w obrazie
Flagi, które są po niej wymagane są następujące:
-i -- po niej należy podać plik z danymi, które program ma ukryć
-p -- po niej należy podać plik obrazu, w którym ma zostać ukryty plik
-o -- po niej należy podać plik wynikowy, w którym będzie ukryty plik
-b -- po niej należy podać na ilu bitach ma zostać zapisana informacja (obsługiwane liczby bitów: 1, 2, 4, 8)
Flaga -r służy do odczytywania danych z obrazu
Flagi, które są po niej wymagane są następujące:
-p -- po niej należy podać plik obrazu, z którego ma zostać odczytany ukryty plik
-o -- po niej należy podać plik wynikowy, do którego na zostać odczytana informacja
-b -- po niej należy podać na ilu bitach została zapisana ukryta informacja
Obsługiwane pliki BMP: 16bit[RGB(1,5,5,5)], 24bit[RGB(8,8,8)], 32bit[ARGB(8,8,8,8), XRGB(8,8,8,8)]
```



## Rozdział 2

# Indeks struktur danych

### 2.1 Struktury danych

Tutaj znajdują się struktury danych wraz z ich krótkimi opisami:

<a href="#">_BITFILEDS</a>	7
<a href="#">_BMPHEADER</a>	8
<a href="#">_DIBHEADER</a>	9
<a href="#">_SGHEADER</a>	11





## Rozdział 3

# Indeks plików

### 3.1 Lista plików

Tutaj znajduje się lista wszystkich udokumentowanych plików z ich krótkimi opisami:

include/ <b>functions.h</b> . . . . .	??
include/ <b>structures.h</b> . . . . .	??
src/ <a href="#">functions.c</a> Implementacje funkcji . . . . .	<a href="#">13</a>



## Rozdział 4

# Dokumentacja struktur danych

### 4.1 Dokumentacja struktury \_BITFILEDS

```
#include <structures.h>
```

#### Pola danych

- uint32\_t redMask
- uint32\_t greenMask
- uint32\_t blueMask
- uint32\_t xMask
- uint32\_t alphaMask

#### 4.1.1 Opis szczegółowy

struktura zawierająca maski BITFIELDS

#### 4.1.2 Dokumentacja pól

##### 4.1.2.1 alphaMask

```
uint32_t _BITFILEDS::alphaMask
```

maska przezroczystości

##### 4.1.2.2 blueMask

```
uint32_t _BITFILEDS::blueMask
```

maska koloru niebieskiego

#### 4.1.2.3 greenMask

```
uint32_t _BITFIELDS::greenMask
```

maska koloru zielonego

#### 4.1.2.4 redMask

```
uint32_t _BITFIELDS::redMask
```

maska koloru czerwonego

#### 4.1.2.5 xMask

```
uint32_t _BITFIELDS::xMask
```

maska zastępcza

Dokumentacja dla tej struktury została wygenerowana z pliku:

- include/structures.h

## 4.2 Dokumentacja struktury \_BMPHEADER

```
#include <structures.h>
```

### Pola danych

- char \* [HEADER](#)
- uint32\_t [SIZE](#)
- char \* [RESERVED1](#)
- uint16\_t [RESERVED2](#)
- uint32\_t [OFFSET](#)

#### 4.2.1 Opis szczegółowy

struktura zawierająca dane ze standardowego nagłówka pliku BMP

#### 4.2.2 Dokumentacja pól

#### 4.2.2.1 HEADER

```
char* _BMPHEADER::HEADER
```

nagłówek pliku, rozpoznaje typ pliku, jeżeli nie jest "BM", nie jest plikiem BMP standardu Windows NT

#### 4.2.2.2 OFFSET

```
uint32_t _BMPHEADER::OFFSET
```

offset pliku pod którym można znaleźć tablicę pixeli

#### 4.2.2.3 RESERVED1

```
char* _BMPHEADER::RESERVED1
```

pamięć zarezerwowana, używane do zapisania informacji o ukrytym pliku (SG)

#### 4.2.2.4 RESERVED2

```
uint16_t _BMPHEADER::RESERVED2
```

pamięć zarezerwowana, używana do zapisywania wielkości ukrytego pliku

#### 4.2.2.5 SIZE

```
uint32_t _BMPHEADER::SIZE
```

wielkość pliku BMP w bajtach

Dokumentacja dla tej struktury została wygenerowana z pliku:

- include/structures.h

## 4.3 Dokumentacja struktury \_DIBHEADER

```
#include <structures.h>
```

### Pola danych

- uint32\_t [DIBSIZE](#)
- uint32\_t [WIDTH](#)
- uint32\_t [HEIGHT](#)
- uint16\_t [COLORPLANES](#)
- uint16\_t [BITSPERPIXEL](#)
- uint32\_t [COMPRESSION](#)
- uint32\_t [SIZE](#)
- uint32\_t [HORIZONTALRES](#)
- uint32\_t [VERTICALRES](#)
- uint32\_t [COLORS](#)
- uint32\_t [IMPORTANTCOLORS](#)

### 4.3.1 Opis szczegółowy

struktura zawierająca podstawowy nagłówek w standardzie DIBHEADER BITMAPINFOHEADER dla Windows

### 4.3.2 Dokumentacja pól

#### 4.3.2.1 BITSPPERPIXEL

```
uint16_t _DIBHEADER::BITSPPERPIXEL
```

głębokość kolorów, ile bitów przypada na pixel

#### 4.3.2.2 COLORPLANES

```
uint16_t _DIBHEADER::COLORPLANES
```

tablica koloru

#### 4.3.2.3 COLORS

```
uint32_t _DIBHEADER::COLORS
```

liczba kolorów w palecie, 0 dla wartości domyślnej  $2^n$

#### 4.3.2.4 COMPRESSION

```
uint32_t _DIBHEADER::COMPRESSION
```

kompresja, rodzaj algorytmu

#### 4.3.2.5 DIBSIZE

```
uint32_t _DIBHEADER::DIBSIZE
```

wielkość nagłówka w bajtach

#### 4.3.2.6 HEIGHT

```
uint32_t _DIBHEADER::HEIGHT
```

wysokość obrazu w pixelach (ze znakiem)

#### 4.3.2.7 HORIZONTALRES

```
uint32_t _DIBHEADER::HORIZONTALRES
```

rozdzielczość na wysokość

#### 4.3.2.8 IMPORTANTCOLORS

```
uint32_t _DIBHEADER::IMPORTANTCOLORS
```

"ważne kolory", wartość zwykle pomijana, 0 dla wszystkich kolorów jako ważnych

#### 4.3.2.9 SIZE

```
uint32_t _DIBHEADER::SIZE
```

wielkość pliku, dopuszczajna wartość "dummy" wynosząca 0

#### 4.3.2.10 VERTICALRES

```
uint32_t _DIBHEADER::VERTICALRES
```

rozdzielczość na szerokość

#### 4.3.2.11 WIDTH

```
uint32_t _DIBHEADER::WIDTH
```

szerokość obrazu w pixelach (ze znakiem)

Dokumentacja dla tej struktury została wygenerowana z pliku:

- include/structures.h

## 4.4 Dokumentacja struktury \_SGHEADER

```
#include <structures.h>
```

### Pola danych

- char \* [HEADER](#)
- uint16\_t [FILESIZE](#)

#### 4.4.1 Opis szczegółowy

struktura opisująca nagłówek specyficzny dla pliku z ukrytym plikiem, zapisywana zawsze na pierwszych 72 pojedynczych bitach tablicy pixeli

#### 4.4.2 Dokumentacja pól

##### 4.4.2.1 FILESIZE

```
uint16_t _SGHEADER::FILESIZE
```

wielkość pliku (1 bajt)

##### 4.4.2.2 HEADER

```
char* _SGHEADER::HEADER
```

nagłówek steganograficzny, jeżeli jest równy "SG\0", plik zawiera ukryte informacje (wielkość: 2 bajty)

Dokumentacja dla tej struktury została wygenerowana z pliku:

- include/structures.h



## Rozdział 5

# Dokumentacja plików

### 5.1 Dokumentacja pliku src/functions.c

implementacje funkcji

```
#include "../include/functions.h"
#include <math.h>
#include <stdint.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
```

#### Funkcje

- int **GCD** (int a, int b)
- int **LCM** (int a, int b)
- int **read\_headers** (FILE \*filePointer, **BMPHEADER** \*bmpHeader, **DIBHEADER** \*dibHeader)
- uint8\_t \* **pixelArray\_read\_16bit** (FILE \*filePointer, int pixelCount)
- uint8\_t \* **pixelArray\_read\_24bit** (FILE \*filePointer, int pixelCount)
- uint8\_t \* **pixelArray\_read\_32bit** (FILE \*filePointer, int pixelCount)
- uint8\_t \* **read\_file\_to\_memory** (FILE \*filePointer, int \*filesize, int \*arrsize)
- **SGHEADER header\_generate** (uint32\_t filesize)
- uint8\_t \* **prepare\_data\_to\_write** (uint8\_t \*dataTab, int bitsPerPixel, int oldArrSize, int \*newArrSize)
- **BMPHEADER prepare\_new\_header** (**BMPHEADER** bmpHeader, **SGHEADER** sgHeader, **DIBHEADER** dibHeader)
- int **write\_headers** (**BMPHEADER** bmpHeader, **DIBHEADER** dibHeader, FILE \*filePointer)
- int **write\_data\_32bit** (uint8\_t \*dataTab, uint8\_t \*pixelArray, **BMPHEADER** header, **DIBHEADER** dibHeader, FILE \*filePointer, int pixels, int arrSize, int bitsPerPixel)
- int **write\_data\_24bit** (uint8\_t \*dataTab, uint8\_t \*pixelArray, **BMPHEADER** header, **DIBHEADER** dibHeader, FILE \*filePointer, int pixels, int arrSize, int bitsPerPixel)
- int **write\_data\_16bit** (uint8\_t \*dataTab, uint8\_t \*pixelArray, **BMPHEADER** header, **DIBHEADER** dibHeader, FILE \*filePointer, int pixels, int arrSize)
- void **write\_data\_to\_image** (uint8\_t \*pixelArray, uint8\_t \*dataArray, **BMPHEADER** bmpHeader, **DIBHEADER** dibHeader, FILE \*filePointer, uint16\_t dataFileSize, int bitsPerPixelStegano, int bitsPerPixel, int pixels, int arrSize)
- int **check\_for\_stegano** (**BMPHEADER** bmpHeader)
- uint8\_t \* **read\_data\_from\_steganofile\_24\_32bit** (uint8\_t \*pixelArray, **BMPHEADER** bmpheader, int bitsPerPixel)
- uint8\_t \* **read\_data\_from\_steganofile\_16bit** (uint8\_t \*pixelArray, **BMPHEADER** bmpheader, int bitsPerPixel)
- int **write\_data\_array\_to\_file** (uint8\_t \*dataArray, int dataSize, FILE \*filePointer)
- int **write\_data\_from\_image** (**BMPHEADER** bmpHeader, uint8\_t \*pixelArray, FILE \*filePointer, **DIBHEADER** dibHeader, int bitsPerPixel)

### 5.1.1 Opis szczegółowy

implemenctacje funkcji

#### Autor

Wojciech Janota

#### Wersja

0.1

#### Data

2020-05-01

#### Copyright

Copyright (c) Wojciech Janota 2020

### 5.1.2 Dokumentacja funkcji

#### 5.1.2.1 check\_for\_stegano()

```
int check_for_stegano (  
    BMPHEADER bmpHeader )
```

funkcja sprawdzająca, czy w nagłówku podanego pliku występuje nagłówek steganograficzny

#### Parametry

<i>bmpHeader</i>	nagłówek BMPHEADER
------------------	--------------------

#### Zwraca

1 jeżeli prawda, 0 jeżeli fałsz

#### 5.1.2.2 GCD()

```
int GCD (  
    int a,  
    int b )
```

pomocnicza funkcja NWD

**Parametry**

<i>a</i>	pierwsza liczba
<i>b</i>	druga liczba

**Zwraca**

NWD

**5.1.2.3 LCM()**

```
int LCM (
    int a,
    int b )
```

pomocnicza funkcja NWW

**Parametry**

<i>a</i>	pierwsza liczba
<i>b</i>	druga liczba

**Zwraca**

NWW

**5.1.2.4 pixelArray\_read\_16bit()**

```
uint8_t* pixelArray_read_16bit (
    FILE * filePointer,
    int pixelCount )
```

funkcja zwracająca tablicę pixeli dla pliku z 16-bitową głębią kolorów

**Parametry**

<i>filePointer</i>	wskaźnik na plik obrazu
<i>pixelCount</i>	licznik pixeli w obrazie

**Zwraca**

tablica pixeli rozbita na poszczególne składowe

#### 5.1.2.5 pixelArray\_read\_24bit()

```
uint8_t* pixelArray_read_24bit (
    FILE * filePointer,
    int pixelCount )
```

funkcja zwracająca tablicę pixeli dla pliku z 24-bitową głębią kolorów

##### Parametry

<i>filePointer</i>	wskaźnik na plik obrazu
<i>pixelCount</i>	licznik pixeli w obrazie

##### Zwraca

tablica pixeli rozbita na poszczególne składowe

#### 5.1.2.6 pixelArray\_read\_32bit()

```
uint8_t* pixelArray_read_32bit (
    FILE * filePointer,
    int pixelCount )
```

funkcja zwracająca tablicę pixeli dla pliku z 32-bitową głębią kolorów

##### Parametry

<i>filePointer</i>	wskaźnik na plik obrazu
<i>pixelCount</i>	licznik pixeli w obrazie

##### Zwraca

tablica pixeli rozbita na poszczególne składowe

#### 5.1.2.7 prepare\_data\_to\_write()

```
uint8_t* prepare_data_to_write (
    uint8_t * dataTab,
    int bitsPerPixel,
    int oldArrSize,
    int * newArrSize )
```

funkcja dzieląca tablicę bajtów odczytanych z pliku wejściowego na paczki bitów na osobnych bajtach

## Parametry

<i>dataTab</i>	tablica z odczytanymi bajtami z pliku wejściowego
<i>bitsPerPixel</i>	na ilu bitach ma być zapisana informacja
<i>oldArrSize</i>	wielkość starej tablicy bajtów
<i>newArrSize</i>	nowa wielkość końcowej tablicy z danymi

## Zwraca

końcowa tablica danych

## 5.1.2.8 read\_data\_from\_steganofile\_16bit()

```
uint8_t* read_data_from_steganofile_16bit (
    uint8_t * pixelArray,
    BMPHEADER bmpheader,
    int bitsPerPixel )
```

funkcja odczytująca ukryte dane z obrazów 16-bitowych

## Parametry

<i>pixelArray</i>	tablica pixeli obrazu wejściowego
<i>bmpheader</i>	nagłówek BMPHEADER obrazu wejściowego
<i>bitsPerPixel</i>	liczba bitów na których została zapisana informacja

## Zwraca

tablica bajtów danych odczytanych z obrazu

## 5.1.2.9 read\_data\_from\_steganofile\_24\_32bit()

```
uint8_t* read_data_from_steganofile_24_32bit (
    uint8_t * pixelArray,
    BMPHEADER bmpheader,
    int bitsPerPixel )
```

funkcja odczytująca ukryte dane z obrazów 24 i 32-bitowych

## Parametry

<i>pixelArray</i>	tablica pixeli obrazu wejściowego
<i>bmpheader</i>	nagłówek BMPHEADER obrazu wejściowego
<i>bitsPerPixel</i>	liczba bitów na których została zapisana informacja

**Zwraca**

tablica bajtów danych odczytanych z obrazu

**5.1.2.10 read\_file\_to\_memory()**

```
uint8_t* read_file_to_memory (
    FILE * filePointer,
    int * filesize,
    int * arrSize )
```

funkcja wczytująca plik bajt po bajcie do pamięci

**Parametry**

<i>filePointer</i>	wskaźnik na plik
<i>filesize</i>	wielkość pliku
<i>arrSize</i>	wielkość tablicy z bajtami

**Zwraca**

tablica bajtów z plikiem wejściowym

**5.1.2.11 read\_headers()**

```
int read_headers (
    FILE * filePointer,
    BMPHEADER * bmpHeader,
    DIBHEADER * dibHeader )
```

funkcja wczytująca dane z nagłówków

**Parametry**

<i>filePointer</i>	wskaźnik na plik, z którego odczytane mają zostać nagłówki
<i>bmpHeader</i>	wskaźnik na nagłówek główny
<i>dibHeader</i>	wskaźnik na nagłówek DIB

**Zwraca**

int jeżeli wczytywanie się powiedzie, zwracana jest wartość 1, w przeciwnym wypadku zwracana jest wartość 0

### 5.1.2.12 write\_data\_16bit()

```
int write_data_16bit (
    uint8_t * dataTab,
    uint8_t * pixelArray,
    BMPHEADER header,
    DIBHEADER dibHeader,
    FILE * filePointer,
    int pixels,
    int arrSize )
```

funkcja zapisująca tablicę pixeli z podmienionymi bitami do pliku wynikowego dla obrazów 16-bitowych

#### Parametry

<i>dataTab</i>	tablica podzielonych danych
<i>pixelArray</i>	tablica pixeli
<i>header</i>	nagłówek BMPHEADER
<i>dibHeader</i>	nagłówek DIBHEADER
<i>filePointer</i>	wskaźnik na plik wynikowy
<i>pixels</i>	liczba pixeli w obrazie
<i>arrSize</i>	wielkość tablicy danych

#### Zwraca

1 jeżeli funkcja wykonała się poprawnie, 0 w przeciwnym wypadku

### 5.1.2.13 write\_data\_24bit()

```
int write_data_24bit (
    uint8_t * dataTab,
    uint8_t * pixelArray,
    BMPHEADER header,
    DIBHEADER dibHeader,
    FILE * filePointer,
    int pixels,
    int arrSize,
    int bitsPerPixel )
```

funkcja zapisująca tablicę pixeli z podmienionymi bitami do pliku wynikowego dla obrazów 24-bitowych

#### Parametry

<i>dataTab</i>	tablica podzielonych danych
<i>pixelArray</i>	tablica pixeli
<i>header</i>	nagłówek BMPHEADER
<i>dibHeader</i>	nagłówek DIBHEADER
<i>filePointer</i>	wskaźnik na plik wynikowy
<i>pixels</i>	liczba pixeli w obrazie
<i>arrSize</i>	wielkość tablicy danych

**Zwraca**

1 jeżeli funkcja wykonała się poprawnie, 0 w przeciwnym wypadku

**5.1.2.14 write\_data\_32bit()**

```
int write_data_32bit (
    uint8_t * dataTab,
    uint8_t * pixelArray,
    BMPHEADER header,
    DIBHEADER dibHeader,
    FILE * filePointer,
    int pixels,
    int arrSize,
    int bitsPerPixel )
```

funkcja zapisująca tablicę pixeli z podmienionymi bitami do pliku wynikowego dla obrazów 32-bitowych

**Parametry**

<i>dataTab</i>	tablica podzielonych danych
<i>pixelArray</i>	tablica pixeli
<i>header</i>	nagłówek BMPHEADER
<i>dibHeader</i>	nagłówek DIBHEADER
<i>filePointer</i>	wskaźnik na plik wynikowy
<i>pixels</i>	liczba pixeli w obrazie
<i>arrSize</i>	wielkość tablicy danych

**Zwraca**

1 jeżeli funkcja wykonała się poprawnie, 0 w przeciwnym wypadku

**5.1.2.15 write\_data\_array\_to\_file()**

```
int write_data_array_to_file (
    uint8_t * dataArray,
    int dataSize,
    FILE * filePointer )
```

funkcja zapisująca dane z tablicy bajtów do pliku wynikowego

**Parametry**

<i>dataArray</i>	tablica bajtów odczytanych z obrazu
<i>dataSize</i>	wielkość tablicy bajtów
<i>filePointer</i>	wskaźnik na plik wynikowy



**Zwraca**

1 jeżeli funkcja wykonała się poprawnie, 0 w przeciwnym wypadku

**5.1.2.16 write\_data\_from\_image()**

```
int write_data_from_image (
    BMPHEADER bmpHeader,
    uint8_t * pixelArray,
    FILE * filePointer,
    DIBHEADER dibHeader,
    int bitsPerPixel )
```

funkcja wrapper zapisująca ukryte dane z przekazanej tablicy pixeli do pliku

**Parametry**

<i>bmpHeader</i>	nagłówek BMPHEADER
<i>pixelArray</i>	tablica pixeli obrazu wejściowego
<i>filePointer</i>	wskaźnik na plik wynikowy
<i>dibHeader</i>	nagłówek DIBHEADER
<i>bitsPerPixel</i>	liczba bitów na których zostały zapisane dane

**Zwraca****5.1.2.17 write\_data\_to\_image()**

```
void write_data_to_image (
    uint8_t * pixelArray,
    uint8_t * dataArray,
    BMPHEADER bmpHeader,
    DIBHEADER dibHeader,
    FILE * filePointer,
    uint16_t dataFileSize,
    int bitsPerPixelStegano,
    int bitsPerPixel,
    int pixels,
    int arrSize )
```

funkcja wrapper służąca do wpisywania danych z pliku do obrazu

**Parametry**

<i>pixelArray</i>	tablica pixeli z obrazu wejściowego
<i>dataArray</i>	tablica bajtów pliku wejściowego

## Parametry

<i>bmpHeader</i>	nagłówek BMPHEADER obrazu wejściowego
<i>dibHeader</i>	nagłówek DIBHEADER obrazu wejściowego
<i>filePointer</i>	wskaźnik na plik obrazu wyjściowego
<i>dataFileSize</i>	wielkość pliku wejściowego
<i>bitsPerPixelStegano</i>	na ilu bitach ma zostać zapisana informacja
<i>bitsPerPixel</i>	głębokość kolorów obrazu
<i>pixels</i>	liczba pixeli
<i>arrSize</i>	wielkość tablicy bajtów pliku wejściowego

## 5.1.2.18 write\_headers()

```
int write_headers (
    BMPHEADER bmpHeader,
    DIBHEADER dibHeader,
    FILE * filePointer )
```

funkcja zapisująca nagłówki do pliku

## Parametry

<i>bmpHeader</i>	nagłówek BMPHEADER
<i>dibHeader</i>	nagłówek DIBHEADER
<i>filePointer</i>	wskaźnik na plik wynikowy

## Zwraca

1 jeżeli wykonano poprawnie, 0 jeżeli wystąpił błąd

# Indeks

- [\\_BITFILEDS, 7](#)
  - [alphaMask, 7](#)
  - [blueMask, 7](#)
  - [greenMask, 7](#)
  - [redMask, 8](#)
  - [xMask, 8](#)
- [\\_BMPHEADER, 8](#)
  - [HEADER, 8](#)
  - [OFFSET, 9](#)
  - [RESERVED1, 9](#)
  - [RESERVED2, 9](#)
  - [SIZE, 9](#)
- [\\_DIBHEADER, 9](#)
  - [BITSPERPIXEL, 10](#)
  - [COLORPLANES, 10](#)
  - [COLORS, 10](#)
  - [COMPRESSION, 10](#)
  - [DIBSIZE, 10](#)
  - [HEIGHT, 10](#)
  - [HORIZONTALRES, 10](#)
  - [IMPORTANTCOLORS, 11](#)
  - [SIZE, 11](#)
  - [VERTICALRES, 11](#)
  - [WIDTH, 11](#)
- [\\_SGHEADER, 11](#)
  - [FILESIZE, 12](#)
  - [HEADER, 12](#)
- [alphaMask](#)
  - [\\_BITFILEDS, 7](#)
- [BITSPERPIXEL](#)
  - [\\_DIBHEADER, 10](#)
- [blueMask](#)
  - [\\_BITFILEDS, 7](#)
- [check\\_for\\_stegano](#)
  - [functions.c, 14](#)
- [COLORPLANES](#)
  - [\\_DIBHEADER, 10](#)
- [COLORS](#)
  - [\\_DIBHEADER, 10](#)
- [COMPRESSION](#)
  - [\\_DIBHEADER, 10](#)
- [DIBSIZE](#)
  - [\\_DIBHEADER, 10](#)
- [FILESIZE](#)
  - [\\_SGHEADER, 12](#)
- [functions.c](#)
  - [check\\_for\\_stegano, 14](#)
  - [GCD, 14](#)
  - [LCM, 15](#)
  - [pixelArray\\_read\\_16bit, 15](#)
  - [pixelArray\\_read\\_24bit, 15](#)
  - [pixelArray\\_read\\_32bit, 16](#)
  - [prepare\\_data\\_to\\_write, 16](#)
  - [read\\_data\\_from\\_steganofile\\_16bit, 17](#)
  - [read\\_data\\_from\\_steganofile\\_24\\_32bit, 17](#)
  - [read\\_file\\_to\\_memory, 18](#)
  - [read\\_headers, 18](#)
  - [write\\_data\\_16bit, 18](#)
  - [write\\_data\\_24bit, 19](#)
  - [write\\_data\\_32bit, 20](#)
  - [write\\_data\\_array\\_to\\_file, 20](#)
  - [write\\_data\\_from\\_image, 21](#)
  - [write\\_data\\_to\\_image, 21](#)
  - [write\\_headers, 22](#)
- [GCD](#)
  - [functions.c, 14](#)
- [greenMask](#)
  - [\\_BITFILEDS, 7](#)
- [HEADER](#)
  - [\\_BMPHEADER, 8](#)
  - [\\_SGHEADER, 12](#)
- [HEIGHT](#)
  - [\\_DIBHEADER, 10](#)
- [HORIZONTALRES](#)
  - [\\_DIBHEADER, 10](#)
- [IMPORTANTCOLORS](#)
  - [\\_DIBHEADER, 11](#)
- [LCM](#)
  - [functions.c, 15](#)
- [OFFSET](#)
  - [\\_BMPHEADER, 9](#)
- [pixelArray\\_read\\_16bit](#)
  - [functions.c, 15](#)
- [pixelArray\\_read\\_24bit](#)
  - [functions.c, 15](#)
- [pixelArray\\_read\\_32bit](#)
  - [functions.c, 16](#)
- [prepare\\_data\\_to\\_write](#)
  - [functions.c, 16](#)
- [read\\_data\\_from\\_steganofile\\_16bit](#)

- functions.c, [17](#)
- read\_data\_from\_steganofile\_24\_32bit
  - functions.c, [17](#)
- read\_file\_to\_memory
  - functions.c, [18](#)
- read\_headers
  - functions.c, [18](#)
- redMask
  - \_BITFILEDS, [8](#)
- RESERVED1
  - \_BMPHEADER, [9](#)
- RESERVED2
  - \_BMPHEADER, [9](#)
- SIZE
  - \_BMPHEADER, [9](#)
  - \_DIBHEADER, [11](#)
- src/functions.c, [13](#)
- VERTICALRES
  - \_DIBHEADER, [11](#)
- WIDTH
  - \_DIBHEADER, [11](#)
- write\_data\_16bit
  - functions.c, [18](#)
- write\_data\_24bit
  - functions.c, [19](#)
- write\_data\_32bit
  - functions.c, [20](#)
- write\_data\_array\_to\_file
  - functions.c, [20](#)
- write\_data\_from\_image
  - functions.c, [21](#)
- write\_data\_to\_image
  - functions.c, [21](#)
- write\_headers
  - functions.c, [22](#)
- xMask
  - \_BITFILEDS, [8](#)