|  |  |
| --- | --- |
| **Politechnika Białostocka**  **Wydział Informatyki** | Data: 31.12.2016 |
| **Przedmiot:** Techniki zapewniania poufności w Internecie.  **Sprawozdanie nr:** 5  **Temat:** STEGANOGRAFIA  **Autor:** Maciej Ziniewicz  **Studia:** stacjonarne II stopnia, semestr 2 | **Prowadzący:**  prof. dr hab. Vyacheslav Yarmolik  Ocena: |

Spis treści

[1. Treść zadania 2](#_Toc464143197)

[2. Część teoretyczna 2](#_Toc464143198)

[3. Rozwiązanie 3](#_Toc464143199)

# Treść zadania

1. Wykonanie aplikacji szyfrującej tekst w obrazie i odczytującej go.

# Część teoretyczna

Steganografia jest nauka zajmującą się ochroną cennej informacji poprzez jej ukrycie w innej nie mającej wartości.

Steganografia cyfrowa daje o wiele większe możliwości ukrycia informacji niż tradycyjna. Bazuje ona, bowiem na dokonywaniu subtelnych zmian w oryginalnym medium w taki sposób, aby ludzkie oko nie było w stanie ich wykryć. Ponadto Internet umożliwia o wiele bardziej swobodny przepływ informacji. Można wręcz umieścić przekaz na publicznym serwerze pozostając anonimowym. Osoba, która jest adresatem wiadomości również może ją niepostrzeżenie odczytać. W przypadku wykrycia informacji przez osoby niepowołane mogą one poznać treść, lecz trudno będzie im ustalić adresata oraz nadawcę. Tak więc niejednokrotnie odczytanie wiadomości może się okazać dla intruza bezużyteczne, ponieważ nie będzie wiedział kogo ona dotyczy.

Nośnikiem ukrytej informacji może być dowolny rodzaj pliku. Najczęściej wykorzystywane są do tego celu pliki multimedialne. Mają one duże rozmiary a ponadto trudno jest wychwycić drobną modyfikację oryginału. Pozwala to na ukrycie dużej paczki danych przy jednoczesnym zachowaniu wysokiego poziomu bezpieczeństwa.

Do wykonania aplikacji została użyta **metoda najmniej znaczących bitów** – LSB (ang. Least Significant Bit) polegająca na podmienieniu jednego lub kilku najmniej znaczących bitów bitami ukrywanymi.

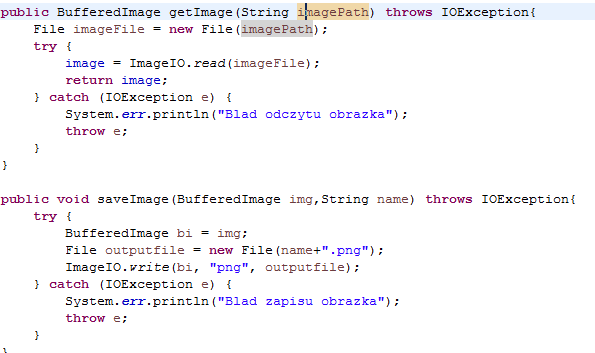
Każdy rodzaj multimediów posiada zapis w którym jego elementy składowe posiadają wartości opisywane liczbowo. Obraz może być zapisany w postaci grafiki rastrowej – czyli kolekcji pikseli, z których każdy ma określone położenie na obrazie oraz barwę. Barwa opisywana jest przy pomocy wartości liczbowych. W przypadku obrazów czarno-białych barwa piksela zapisana jest na jednym bicie. Wartość 0 oznacza kolor czarny a wartość 1 kolor biały. Zmiana wartości zmienia kolor piksela. W przypadku obrazów reprezentowanych w skali szarości, kolor każdego piksela zapisywany jest na 8 bitach co daje 256 różnych możliwości. Wartość 0 oznacza kolor czarny a wartość 255 kolor biały. Zwiększanie wartości liczbowej reprezentującej kolor powoduje jego rozjaśnianie. Niewielka zmiana wartości spowoduje wprowadzenie niewielkich zmian w odcieniu modyfikowanego piksela co z reguły pozostaje niezauważalne dla ludzkiego oka. Jako że chcemy do nośnika dołączyć wartości bitów, najprościej jest podmienić pewne bity przechowujące wartość kolory piksela na bity ukrywanej informacji. Jako że zależy nam na tym by wprowadzone zmiany były jak najmniejsze to podmieniamy wartości najmniej znaczących bitów, gdyż ich zmiana w najmniejszym stopniu wpływa na zmianę koloru danego piksela.

Generalna zasada jest taka: im więcej bitów modyfikujemy tym więcej informacji możemy ukryć (uzyskać większą pojemność steganograficzną). Jednak im więcej bitów zmodyfikujemy tym bardziej będą widoczne wprowadzone zmiany.

# Rozwiązanie

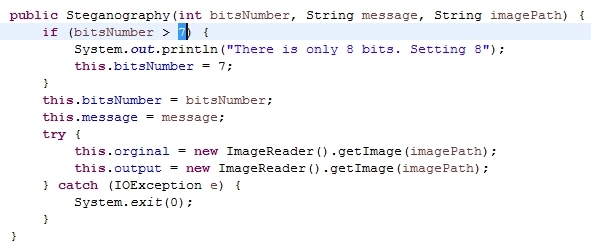
Aplikacja składa się z dwóch klas. Jedna Wczytująca i zapisująca obrazy, oraz druga szyfrująca i generująca nowy obraz. Aplikacja oznacza koniec treści dwona znakami NULL.

Odczyt/Zapis:

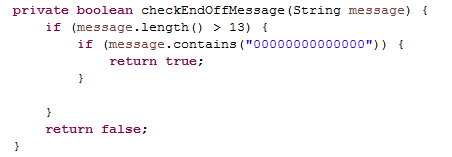


Metoda getImage wczytuje obraz o podanej nazwie, a metoda saveImage zapisuje podany obraz w pliku o podanej nazwie. Pliki wczytywane musza znajdowac sie w katalogu lokalnym dla aplikacji, a pliki generowane pojawią się w tym katalogu.

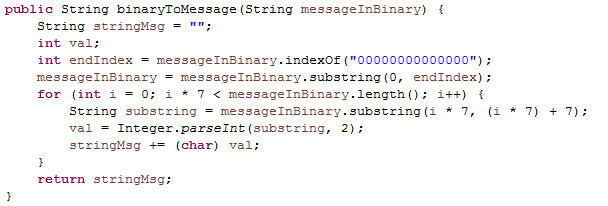
Klasa wykonująca szyfrowanie składa się z dwóch m.in głownych metod cipher i decpiher oraz konstruktora w którym definiujemy ilość bitów na których ma być zapisywana treść, tekst który chcemy zaszyfrować oraz obraz w którym ma byc szyfrowany tekst.



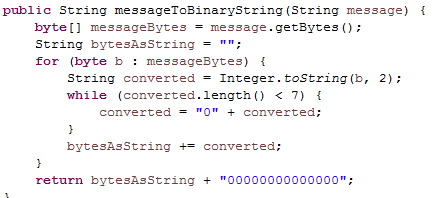
Do wytlumaczenia działania głownych metod cipher i decipher należy wpierwsz przedstawić mniejsze z których powyższe korzystają:



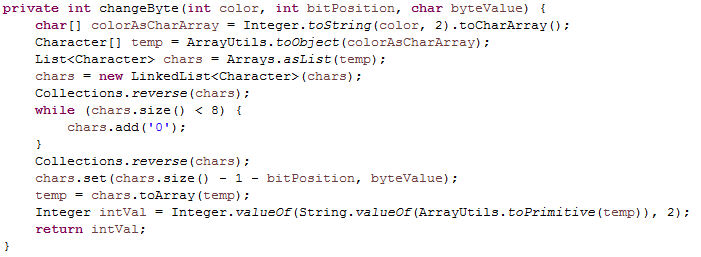
Powyższa metoda sprawdza czy wiadomość została odczytana, sprawdzane jest to wykryciem dwoch znaków NULL w tlumaczonej wiadomości.



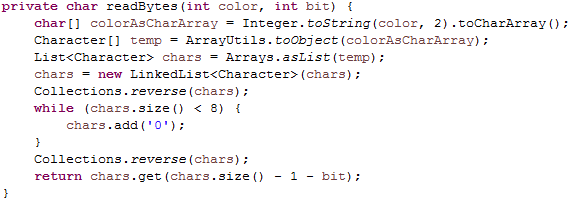
Kolejna metoda binaryTOMessage zamienia wartość binarną wiadomości na tekst, usuwając oznaczenie w postaci dwóch znaków NULL, a następnie dzieląc podany ciąg binarny na wartości 8-bitowe które są zamieniane na wartość liczbową która następnie jest konwertowana do znaku jej odpowiadającego.



Powyższa metoda messageToBinaryString zamienia podany tekst na ciąg binarny, generując tablicę bajtów gdzie każdy bajt odpowiada znakowi w wiadomości a następnie każdy bajt jest prezentowany binarnie.



Następna metoda changeByte podmienia wartość bitu w kolorze na określonej pozycji. Wartość koloru podawana jest jako liczba co zamieniane jest na bity, następnie na wybranej pozycji bit jest podmieniany, po czym nowa wartość jest konwertowana spowrotem do wartości liczbowej.

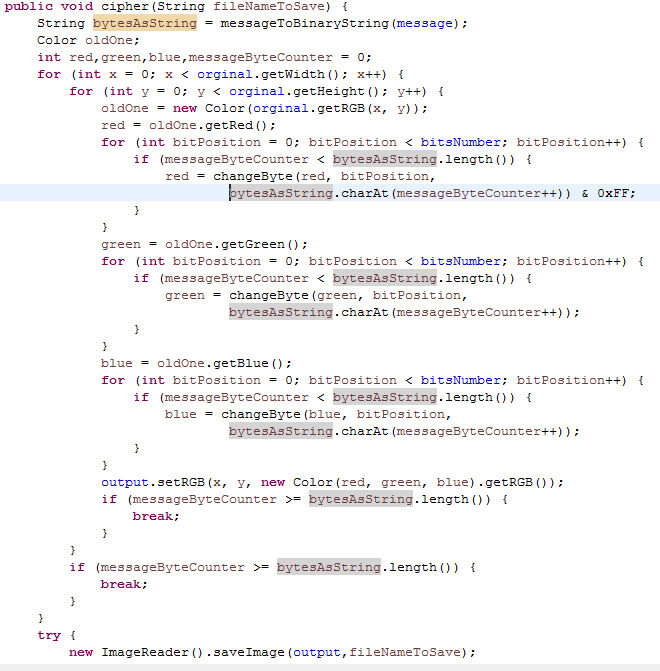


Ostatnia z mniejszych metod to readBytes która zakodowany bit z koloru.

Głowną metodą szyfrującą jest metoda cipher która wywołuje metodę messageToBinaryString, dzięki której otrzymujemy reprezentacje tekstu w formie binarnej, następnie dla każdej pozycji x i y w obrazcie czyli dla każdego pikselu wykonywane są następujące operacje:

* Odczytanie koloru
* Pobranie składowej koloru(czerowny,zielony,niebieski)
* Dla składowej koloru wywoływana jest pętla która podmienia rządane bity w składowej a następnie zapisuje je do zmiennej, podmiana odbywa się za pomocą metody changeByte.
* Na podstawie wygenerowanych nowych składowych kolorów tworzony jest nowy kolor który zostaje podmieniony w obrazie do zapisu.
* Sprawdzenie końca wiadomośći, jeżeli licznik bitów jest równy lub większy od ilości bitów wiadomości przerywana jest pętla iterująca po wszystkich pikselach żeby skrócić czas operacji

Na koniec obraz z zakodowaną wartością jest zapisywany do pliku.



Kolejną metodą jest metoda deszyfrująca. Na początku pobiera ona zaszyfrowany obraz.

Następnie dla każdego piksela wykonuje następujące czynności:

* Pobranie koloru.
* Wykonanie pętli odczytującej zaszyfrowane bity z każdej składowej koloru za pomocą metody readBytes
* Sprawdzenie czy wiadomość została odczytania, jeżeli tak przerywane są petle, jeżelnie nie kontynuowane.

Na koniec wartość binarna odczytana jest zamieniana na tekst za pomocę metody binaryToMessage.

