CS101 Homework #3

Steganography program

Due date: Friday, May 23, 2014 (Until 23:59)

No delayed due date

Please read the homework description carefully and make sure that your program meets all the requirements stated. The homework is an individual task. You can discuss the problem with your friend but you should not program together. **You will get an F on the entire course if your homework includes any plagiarism.**

Goal

Steganography is a program for hiding information into a media file (image, mp3, or video). In this homework, a simple steganography program hides information (text data or small image data) into a big image file, and the program also extracts the information from the big image file.

Description

Since people cannot sense a slight change of image data, a steganography program can hide information into a big image file. To divide the information to small values, this simple steganography program converts the information into a binary string. After that, each binary value in the binary string is hidden into each color value in the big image file. When extracting the information from the big image file, the steganography program samples binary values and converts the binary values to the information.

1. Converting information into a binary string

There are two types of the information, text data and small image data. **The text data** is converted into a binary string by using 'letter_list' in the template code file. You should find an index of each letter in the text data, and then you should convert the index into a binary string. The length of each binary string MUST be 6. For example, text data "K" and "Cs" are converted into "001010" and "000010101100" respectively. **In the image data**, each

color value (r, g, b) is converted into a binary string. The length of each binary string MUST be 24 (8 * 3). For example, the color value (150, 220, 50) is converted into "100101101101110000110010".

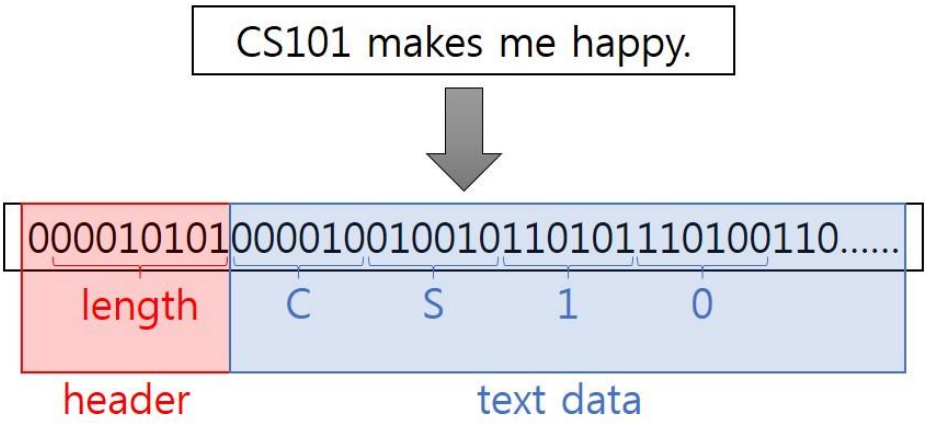

Figure 1. Example of converting text data

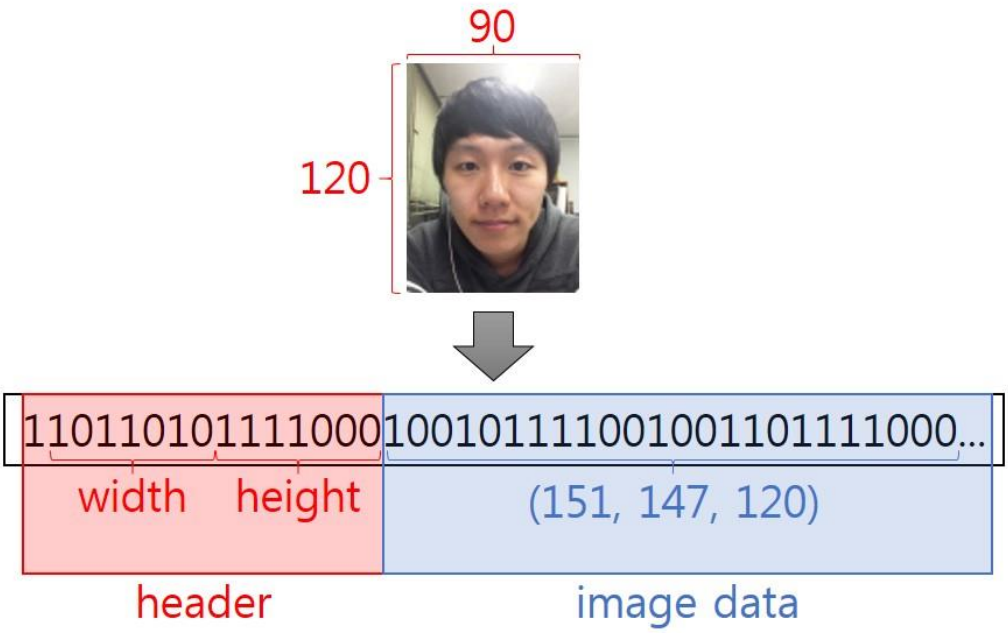

Figure 2. Example of converting image data

This steganography program needs header in front of the binary string to know the information type and size. The first binary value in the header is used for distinguishing between the text data and image data. The first binary value is "0" when the information is text data, and the first binary value is "1" when the information is image data.

The header length of text data is 9 where following 8 binary values are the length of the text data. So, the maximum length of text data is 255. On the other hand, the header

length of image data is 15 where following 14 binary values are the size of image data (each 7 binary values for a width and height of the image data). The maximum size of the image data is (127 * 127). Figure 1 and Figure 2 show examples of converting text data and image data respectively.

2. Hiding a binary string into a big image file

To hide a binary string into a big image file, the binary string is divided into each three binary values. Each three binary values are hidden into each color value in the big image file. First of all, you should make each color value as an even number using subtraction (e.g. (134, 57, 219) → (134, 56, 218)). After that, the three binary values are added to the color value (e.g. (134, 56, 218) → (135, 57, 218) when the three binary values are "110"). Figure 3 shows an example of hiding a binary string into a big image file.

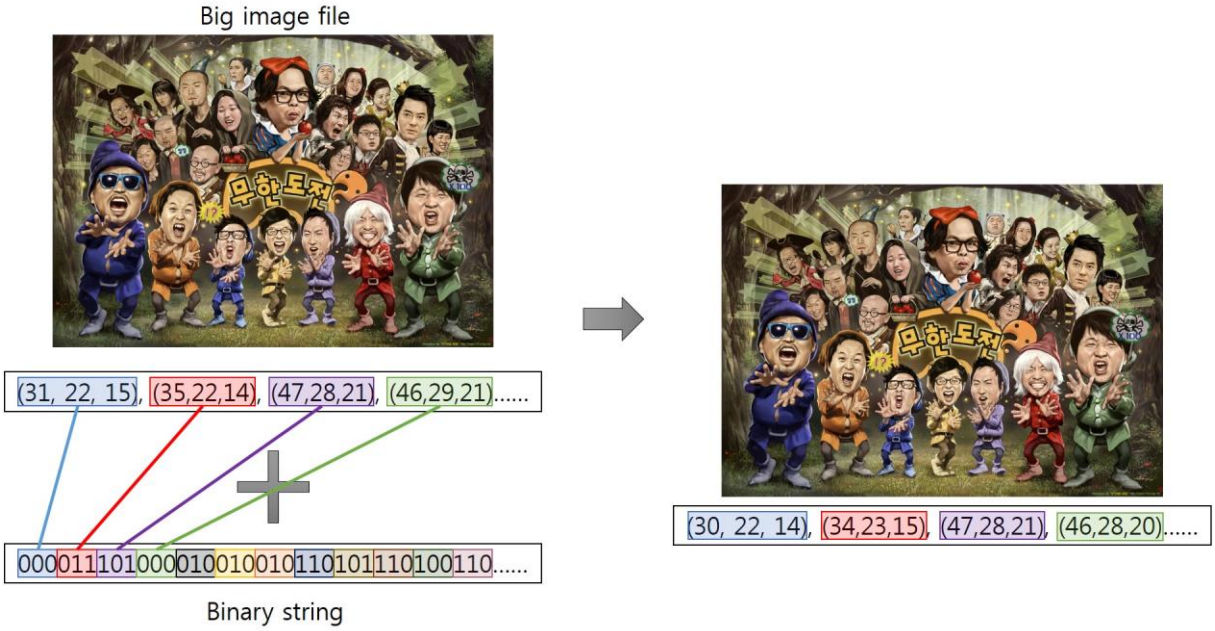

Figure 3. Example of hiding a binary string

3. Extracting information from a big image file

This steganography program samples a binary value from each color value. If the color value is an odd number, the binary value is "1". Otherwise, the binary value is "0" (e.g. sampling "101" from a color value (211, 106, 79)). After that, the program makes a binary string by joining all the sampled binary values. We can know the type and size of the information from header in the binary string. Then, this program can convert the binary

string to text data or image data. Figure 4 shows an example of extracting a binary string into the big image file and converting the binary string to the information.

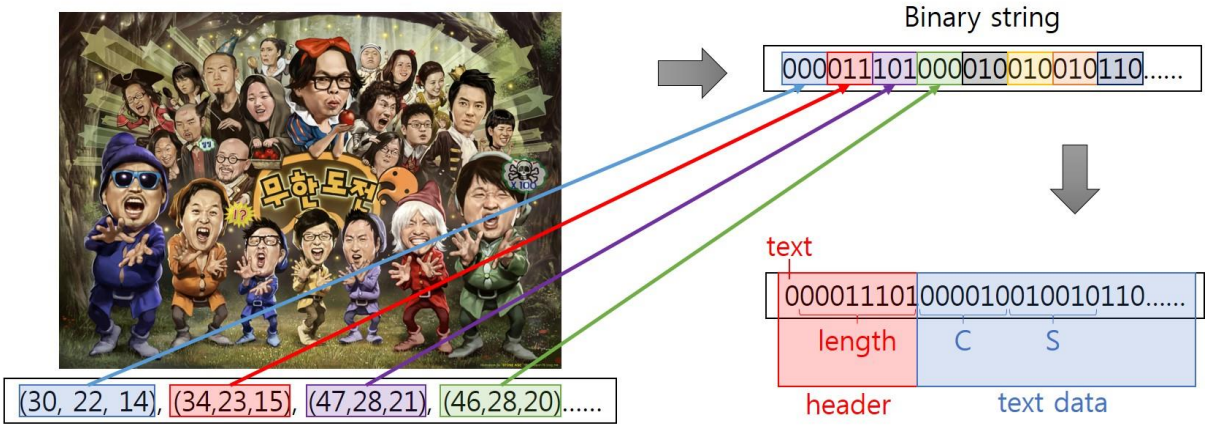

Figure 4. Example of extracting information

Requirement

Figure 5 and Figure 6 show hiding & extracting text data and image data respectively. We assume that we already know image file names, and you can load images from the image files. You should use image files of a 'bmp' type (Other image types cannot be used in this program).

```
================================
0. Exit
1. Hiding data
2. Extracting data
================================
Select a menu : 1
================================
1. Hiding text
2. Hiding image
================================
Select a menu : 1
Enter an image file name for hiding : Yuna_Kim.bmp
Enter text data : CS101 makes me happy.
```

```
================================
0. Exit
1. Hiding data
2. Extracting data
================================
Select a menu : 2
Enter an image file name for extracting : Yuna_Kim.bmp
CS101 makes me happy.
```

Figure 5. Hiding & extracting text data

```
================================
0. Exit
1. Hiding data
2. Extracting data
================================
Select a menu : 1
================================
1. Hiding text
2. Hiding image
================================
Select a menu : 2
Enter an image file name for hiding : Infinity_Challenge.bmp
Enter a hiding image file name: Dohoo.bmp
```

```
================================
0. Exit
1. Hiding data
2. Extracting data
================================
Select a menu : 2
Enter an image file name for extracting : Infinity_Challenge.bmp
```

Figure 6. Hiding & extracting image data

You should define eight functions, 'integer_to_binary()', 'binary_to_integer()', 'text_to_binary()', 'binary_to_text()', 'image_to_binary()', 'binary_to_image()', 'hiding()', and 'extracting()'.

1. **integer_to_binary(*i, l*)**

- Input: integer value *i*, length *l*
- Output: binary string
- Convert an integer value *i* into a binary string.
- A length of the binary string should equal the length *l*.

2. **binary_to_integer(*bs*)**

- Input: binary string *bs*
- Output: integer value
- Convert a binary string *bs* into an integer value.

3. **text_to_binary(*ts*)**

- Input: text string *ts*
- Output: binary string
- Convert a text string *ts* into a binary string.
- First of all, make header which include "0" and a length of the text string *ts*. The length should be also converted into binary string by using the 'integer_to_binary()' function.
- Find an index of each letter in the text string *ts* by using 'letter_list', and then convert the index into a binary string by using the 'integer_to_binary()' function.
- The length of a binary string for each letter is 6.
- Join the header and every binary string for each letter of the text string *ts* in sequence.
- If the length of the text string *ts* is longer than the maximum text length (255), return None.

4. **binary_to_text(*bs*)**

- Input: binary string *bs*
- Output: text string
- Convert a binary string *bs* into a text string.
- First, find the length of a text string through header in the binary string *bs*.
- Convert each 6 letters following the header into a integer value by using the 'binary_to_integer()' function, and then find a letter in the 'letter_list' with the integer

value.

- Make the text string by joining all letters.

5. image_to_binary(*img*)

- Input: image *img*
- Output: binary string
- Convert an image *img* into a binary string.
- Make header which include "1" and a size of the image *img*. The size is composed of a width and height which are also converted into binary strings.
- Convert each color value in the image *img* into a binary string by using the 'integer_to_binary()' function.
- The length of a binary string for each color value is 24 where the length of red, green, and blue values are commonly 8.
- Join all binary strings for each color value in sequence.
- If the size of the image *img* is bigger than the maximum image size (127 * 127), return None.

6. binary_to_image(*bs*)

- Input: binary string *bs*
- Output: image
- Convert a binary string *bs* into an image.
- Find a size of the image through header in the binary string *bs*.
- Make a new picture by using the size, and set each color value converted from each 24 letters in the binary string *bs*. You also use the 'binary_to_integer()' function to convert the binary string.

7. hiding(*bs*, *img_fname*)

- Input: binary string *bs*, image file name *img_fname*
- Output: None (No return)
- Hide a binary string into a big image file.
- Load the image file from an 'original_pictures' folder, and then modify color values in the image by using the binary string (please refer to the Figure 3.). After that, save the modified image to a 'steganography_pictures' folder.
- You should check that the binary string *bs* can be hidden into the big image file. (the

binary string *bs* is None, or the big image file is too small to hide the binary string *bs*.) If the binary string cannot be hidden, print an error message and finish this function without hiding.

8. extracting(*img_fname*)

- Input: image file name *img_fname*
- Output: None (No return)
- Show hidden information (text data or image data) in a big image file.
- Load the image file from a 'steganography_pictures' folder, and then extract a binary string in the image.
- If the binary string is text data, convert the binary string into a text string by using 'binary_to_text()' function. After that, print the returned text string from the function.
- Otherwise, convert the binary string into an image by using 'binary_to_text()' function, and show the returned image.

Submission

You need to submit the followings:

- The file 'yourid.py' (e.g.) 20141234.py
- The report 'yourid.doc' or 'yourid.docx' or 'yourid.pdf' or 'yourid.hwp' (e.g.) 20141234.doc, 20141234.docx, 20141234.pdf, 20141234.hwp
- In the report, you need to describe each function and show your results.

You have to archive the source code and the report into 'yourid.zip', and then you submit the archived file through the webpage for homework submission (e.g.) 20141234.zip

In this homework, we will not accept the delayed homework, and the server will be closed after May. 23. Please try to submit your solution before the due date.