CS 105 – Lab 7 – Steganography

Today, let's write some Pascal programs that practice steganography. This means we are going to hide a secret message inside an otherwise ordinary-looking text file.

Log into the computer as usual. Create a new folder called lab7 to hold today's files.

From the class Web site, please download the large text files `reagan.txt` and `alice.txt`. We will hide a secret message inside each file.

The first program to write is called `steg.pas`. This program will hide a short secret message among the characters of President Reagan's speech. Therefore, `reagan.txt` is the input file. And it will write the resulting output to another file, `reaganmodified.txt`.

Here is the code for `steg.pas`, except that some details have been omitted:

```
(* steg.pas – Let's hide a short message. *)
program steg;

var
  (* delcarations go here *)
begin
  SecretMessage := 'Meet me at Rancho Cielo in Santa Barbara, California';

  (* Tell Pascal that infile refers to reagan.txt and
   * outfile refers to reaganmodified.txt
   *)
  i := 1;
  while not eof(infile) do
    begin
      readln(infile, line);
      if (length(line) >= 35) and (i <= length(SecretMessage)) then
        begin
          line[35] := SecretMessage[i];
          i := i + 1;
        end;
      writeln(outfile, line);
    end;

  close(infile);
  close(outfile);
end.
```

Compile and run the steg program.

1. Look at the output file `reaganmodified.txt`. Can you tell if anything has changed relative to the original `reagan.txt` file? If so, what is different?

2. In `steg.pas`, what is the meaning of "while not eof(infile) do"?

3. Inside the loop, we have an if-statement with two conditions that need to be satisfied. The first condition is `length(line) >= 35`. Why do we need this condition?

4. Why do we need the condition `i <= length(SecretMessage)`?

5. What does the expression `SecretMessage[i]` mean? How is it used in the program?

6. What will the program do if the either if condition is false? In other words, what happens if either the length of the line from the input file is less than 35, or `i` exceeds the length of the secret message?

Next, let's write `steg2.pas`. This program will attempt to discover the hidden message that is contained inside `reaganmodified.txt`. The program will print the secret message to the screen. In other words, `steg2.pas` only needs to open one file – an input file, not an output file.

Here is the code for `steg2.pas`, although some details have been omitted:

```
(* steg2.pas – Extract the hidden message. *)
program steg2;

var
  (* declarations go here *)
begin
  SecretMessage := '';

  (* Tell Pascal that reaganmodified.txt is the infile. *)

  (* Read each line of the modified speech.
   * Grab the 35th character of each line, and append
   * it to the message.
   *)
  while not eof(infile) do
    begin
      readln(infile, line);
      if length(line) >= 35 then
        SecretMessage := SecretMessage + line[35];
    end;

  close(infile);
  writeln('The secret message is:');
  writeln(SecretMessage);
end.
```

Compile and run the steg2 program.

1. Is the secret message being printed out correctly? Why is the program printing more text than the length of this message?

2. Why do we need the if-statement condition "length(line) >= 35"?

3. The first time we execute the assignment statement
   `SecretMessage := SecretMessage + line[35]`
   what are the values of SecretMessage and line[35] on the right side of the := ?

4. What are the values of SecretMessage and line[35] on the right side of the assignment statement the second time it is executed? ✓

Part 2 (using alice.txt)

Steganography is a clever skill! But we have to admit that changing one character on every line of a file isn't very stealty. The modifications are too obvious. We need to come up with a more subtle way to hide the message. That will be the goal of our next program, `steg3.pas`.

Copy steg.pas and steg2.pas into new versions steg3.pas and steg4.pas like this:

```
cp steg.pas steg3.pas
cp steg2.pas steg4.pas
```

The purpose of `steg3.pas` is to hide another secret message inside a larger text file `alice.txt`. And later, `steg4.pas` will attempt to extract this message. Since we have such a large text file in which to hide our message, we can do a better job scattering the individual characters of our secret message. Instead of storing one character per line, we will store a character once every several lines. For example, one character per every 5 lines of text.

Make the following changes to `steg3.pas`:

- Update the comment and program declaration appropriately.
- Change the secret message to: *The Mad Hatter paid too much for his hat*.
- The `infile` and `outfile` should refer to `alice.txt` and `alicemodified.txt`, respectively.

- Since we want to store one character per 5 lines of text, we need some way to count lines from the input file. Therefore,
  - Create a new variable called `LineNumber`.
  - Initialize it to zero before the while loop.
  - Immediately after reading a line of input, increment `LineNumber` by 1.
- Add a third condition to the if-statement: `LineNumber mod 5 = 0`. It would make sense to write this condition before the other two. All three conditions should be joined using "and", because we want them all to be true.

Compile and run the steg3 program. It creates an output file `alicemodified.txt`. Since you personally know the secret message and wrote the program that hid it, you should be able to find where inside the output file the individual characters of the secret message wound up.

1. The first word of the secret message is "The". On which line numbers of the output file do these letters appear?

2. Inside `alicemodified.txt`, which is the first line that could have contained a character from your secret message, except that the line was too short?

Now, we should be able to extract the secret message. To do so, let's modify `steg4.pas` as follows:

- Make the appropriate updates to the introductory comment and program declaration.
- Tell Pascal that `alicemodified.txt` is the input file.
- Since the secret message is contained only on certain lines of the file, we need to keep track of the line number. Therefore,
  - Create a new variable called `LineNumber`.
  - Initialize it to zero before the while loop.
  - Immediately after reading a line of input, increment it by 1.
- Add a second condition to the if-statement saying `LineNumber mod 5 = 0`. It would make sense for this condition to appear first. Join both conditions with the word "and" because we want both of them to be true.

Compile and run `steg4.pas`. Verify that the output is correct. ✓