(350 pts) CS 4463 Steganography – Project

For your semester project you have 3 options:

1) Develop a steganographic program to hide and extract data
2) Develop a steganalysis tool to detect hidden data
3) Develop a 30-45 minute presentation on a specific steganographic technique (may be smaller team)

For this project, we will generally use teams of 3 or 4. In some cases, I may approve a team of two or one.

Ideally, each team will consist of a team leader, a senior coder, a secondary coder and/or analyst. This is a group project and I expect all members of the group to participate. A group that excludes a weaker member will lose points – find a way to use their skills! The team leader is responsible to coordinate actions, be the deciding vote on an approach, and write up the midpoint and final reports. The senior coder is responsible to work with the other coder/analyst to complete the coding task. Of course, each team is free to divide the work as needed. IF there are team members not responding, let me know ASAP and we'll reach a solution.

Since it is 350 points, 35% of your grade, and you have effectively 4 weeks to complete it, I strongly suggest you start early. **In the nearly twenty years I've taught this class, I have never seen a successful project from a group that waited until the last weekend to begin**.

Option 1:
Develop a steganographic program to hide data within a 24-bit bitmap image, 8-bit grayscale bitmap image, 8/4/1-bit paletted image, jpeg, or an 8/16-bit audio wave file. Any of these formats will present roughly the same challenge. I will give you C code to use for reading/writing bitmaps, jpegs, or waves. Other formats are open for discussion and must be approved by me.

You MUST use C/C++ or python code for this project. You may code it on either Linux or Windows (or MacOS for Python projects). I will give a list of potential projects in class – your team will need to choose by milestone #1.

This assignment will NOT involve ANY graphical user interfaces, permutations, or encryption. I am interested in the steganographic aspect, not how pretty it is. It should be a command line program that takes the following types of inputs:

```
stego.exe –hide –m <message file> -c <coverfile> [-o <stego file>]
```

- hides the message file inside the cover file and outputs the stego file
- specifying a name for the stego file is optional
- -m random should also be an option where the message is just random bits

```
stego.exe –extract –s <stego file> [-o <message file>]
```

- extracts embedded information from the stego file and saves it to a message file
- specifying the name for the message file is optional (you can embed it if you choose)

You may add some additional options or change names if you desire. You should definitely do basic error-checking: the program should not crash if there is error in the input.

Your team may choose from the list of projects on the last page of this document or suggest something else. I suggest you do a little independent research on the various steganographic techniques. I have over 2200 papers so if you find a paper but can't get access I can probably help you find it.

I expect the source code to be well commented – I will be grading the comments as well as the overall functionality. Each function should have header comments describing the overall purpose of the function. Each section of code should have comments explaining what and especially WHY you are doing what you are doing.

```
X = 0;      // setting X to zero                  *** WORHTLESS COMMENT ***
            // initializing the pixel counter      *** GOOD COMMENT ***
```

I strongly suggest that the software be developed in an incremental fashion. There are numerous parts and these should be developed and tested separately and then integrated. At ANY point in time, you should have a version of software that should compile and work up to that point. Make a BACKUP of each working version.

For instance, the following code compiles and works:

```
void main( ) {
   return;
}
```

Now, add the command line parsing and make sure that works. Add the file read functions and make sure they work. At each point, backup a working copy or use GIT or other version control software.

Option 2:
Develop a steganalysis tool to detect a specific type of hidden data. Many of the same comments above apply to a steganalysis tool. Remember, you will need to demonstrate the effectiveness of this tool, so you will also need a stego program to hide the data. I do have some working programs that you can use, or you can go against another group's project or an online program that you were able to download.

For Options #1 and #2, I am expecting a complete and working project for full credit.

In the event a project is not complete, you will be graded based upon the portions that you did complete. Do NOT turn in any code that does not compile error free and execute without crashing. In NO event shall any group make any electronic copies of another group's work. You are welcome to collaborate with other groups, get help in coding and/or debugging, but any group caught making exact or near-exact copies will receive a zero on the entire project and FAIL the course. If your group receives significant help on any part of the project, give the other group credit. Example, you could not get your DCT function to work and "Fred" eliminated some serious bugs and got your code working. In the comments above the function, say something like, "Fred helped us find the bugs here." **In two consecutive recent semesters**, a group waited too long, got desperate, and discovered that the professor knows how to use Google too. **Do NOT copy your project from the Internet**. When we discuss M#1, I usually add little caveats that make the project unique as well, so generally there will not exist an exact project.

Option 3:
Develop a presentation on a specific steganography topic that is suitable for an in-class presentation. You will be expected to deliver the presentation as well. This project involves more research and writing, but little or no coding. HOWEVER, it is not an easy project. We will work closely together on this one because it will need to be something that is not already part of the class. In order to get an A or a B, it will have to be good enough for me to use in future classes.

For ALL options, you will need at least 3 peer-reviewed journal articles to use as a reference and we will make sure early that this is doable.

Non-Technical Issues:
Any non-technical issues (i.e. lazy group member or over ambitious group member) should be handled separately and directly with me as soon as it becomes a problem! After the final exam is TOO LATE to address these types of issues – I can't do anything about it at that point. I am not going to punish "Fred" or "Sally" without giving them a chance to present their perspective. If you are having group issues, first, try to resolve it, but if that fails, get me involved early and we'll resolve it one way or another.

DELIVERABLES: (50% penalty for late submissions)

M1: Midpoint Progress Review: (25 pts) **Due Thu Jul 10th, 11:59pm**

- A **cover page** with the title, group number, names of all group members, and the class name and dates
  - ALL group members MUST turn in a copy, preferably the exact same one as everyone else.
- **Introduction** – explain the project, the general approach and goals - I do NOT need a paraphrased paragraph on the definition of steganography
- **Functional block diagram** – the basic outline of your primary hiding/extracting algorithms
  - This diagram needs to be detailed, almost as much as pseudo code
  - "Get message," "hide data" are NOT detailed
    - HOW did you get the message? Which bits are extracted each time?
    - DETAILS. Remember, this isn't code, it's pseudo code.
  - Do a diagram for hiding and another for extracting
    - In many cases they will be similar, but still NOT the same – extracting does not require a cover for instance
    - Sometimes, extracting is completely different
- **Bibliography** - references to **<u>at least 2</u>** relevant professional papers. Use same format as in the professional papers. May also include websites in addition to the professional papers.
- **NAME your document in the following way**:
  - "**2025_06_CS4463_Team_XX_Project_M1.docx**" (Incorrect name? Minus 10 points)
    - XX is your teams number
    - Can be a .docx file or .pdf

This portion is worth 25 points. A progress review hammered out an hour before the due date is not going to get many points. Put some effort into this – then it will be complete for the final report too.

M2: Source Code & Executable and Supplemental Files: (225 pts) **Due Sun Jul 27th, 11:59p.m.**

- I need to run the program with a couple of your test files
  - I will likely have some also
- Code must be turned in as a separate document from the final report
- Make SURE your names are included at the top of the source code
- Program should have documentation including the following:
  - Usage when program is run with no parameters
  - Brief explanation of how to run it including examples
- Turn in a zip file with all source code documents and the executable (or make file for Linux)
  - I have no ability to grade MacOS executable files
- Do enough error checking such that the program does not crash when the user inputs faulty data or the file type is incorrect
- Name your .zip file in the following way:
  - "**2025_06_CS4463_Team_XX_Project_M2.zip**" (Incorrect Name: Minus 10 points)

M3: Final Report: (100 pts) **Due Thu, Aug 7th, 11:59pm**

- **cover page** with the title, group number, names of all group members, and the class name and dates
- **introduction** – explain the project, the general approach and goals, *and the results of analysis*
- **functional block diagram** – the basic outline of your primary hiding/extracting algorithms
- **discussion** of the algorithm and the overall hiding/extracting techniques
- **discussion** of any technical difficulties you had with implementation, any bugs still left, and any features that go beyond the basic approach
- **suggestions** for future work
- **examples** of images with data hidden (if applicable) or a reference to sound files
- **statistical analysis results** – the results of your hiding technique, what was your capacity, at what point did it become humanly perceivable, etc.
- **bibliography** - references to at least 3 relevant professional papers. Use same format as in the professional papers (Hint: cut & paste). May also include websites or texts.

Turn in Summary:

TWO MAJOR POINTS:

#1 – All of your files (Except M#1) must be in a zipped up file named as follows:

"**2025_06_CS4463_Team_XX_Project_M#.zip**" where XX is your Team's number. FAILURE to follow this naming convention will result in a deduction of 10 points for each violation.

# = Milestone number, 1, 2, 3

#2 – Your program MUST run. On Windows for C/C++, it is **imperative** that you set the project properties correctly in Visual Studio as described in the Visual Studio Setup slides, slide #13. If it doesn't run, then you will lose 50 points. You are encouraged to submit a program early and I can check it (by request) to make sure it runs. This test does NOT have to be a completed program – I will simply check that it runs.

1) Softcopy of ALL materials. The final report, all source code, executable program, and any peer-reviewed journal articles you referenced or used. Include some brief instructions on how to run your program.
    a. The program will be graded by running it. I suggest including a few test cover/message files.
    b. Make it as easy to grade as possible – instructions, easy-to-find program, etc.
    c. If I can't run it, you will lose points
    d. I suggest sending me a pre-version that I can verify I can run it ahead of time
    e. For linux, include a makefile so I can quickly/easily compile it
    f. A program that that crashes will cost 50% of your program grade
    g. Do NOT include a full project file for Windows (often MB in size) – just the source code

    h. ALL of your electronic files should be included in single .zip file for each team. Each team member should submit the same .zip file. This is a little different than in the past, but this will prove that each team member had a chance to see whatever is submitted.

Ideas for Projects: (not comprehensive)

1. Run-Length/Edge Hiding in binary image (2-color)
2. ~~Hide in a grayscale or color image using BPCS Techniques~~
3. Hide using audio file statistics
4. Hide using bitmap file statistics
5. Hide in a DNS Packet
6. Hide in the DCT coefficients of a jpg file
7. Steganalysis using jpeg compatibility and a few other tools
8. Dr. Fridrich Palette Hiding Technique
9. Nearest Luminance Palette Hiding Technique
10. Duplicate palette (permuted palettes to make duplication difficult to detect)