Please do not reply to this email. Contact your lecturer or demonstrator with any queries about your results.

Assignment 2

More steganography

FIT3042 - System Tools and Programming - 1 - 2017

26916592 - PAVIN WU

Task 1: Basic functionality

Task 1.1: Functional correctness: hide/unhide

Task 1.1.1: Multiple files

hide -m implemented?	☐ No ☑ **Yes**
unhide -m implemented?	☐ No ☑ **Yes**
Recognizes -m flag correctly	☐ Poor ☐ Okay ☑ **Good** ☐ N/A
Handles input PPM filenames correctly	☐ Poor ☑ **Okay** ☐ Good ☐ N/A
Does not attempt to hide message in headers	☐ Poor ☐ Okay ☑ **Good** ☐ N/A
hide works with -m flag	☐ Poor ☐ Okay ☑ **Good** ☐ N/A
unhide works with -m flag	☐ Poor ☐ Okay ☑ **Good** ☐ N/A

Task 1.1.1 Comment: Uses first input image twice to store message, then uses following images in order.

Mark for task 1.1.1: 2.5 out of 3.0

Task 1.1.2: Parallel execution

hide -p implemented?	☐ No ☑ **Yes**
Creates subprocesses using fork()	☐ Poor ☐ Okay ☑ **Good** ☐ N/A
Subprocesses run in parallel, not sequentially	☐ Poor ☐ Okay ☑ **Good** ☐ N/A

Task 1.1.2 Comment: Would be easier to use dup2 in the child just after it has forked, before calling exec.
Parent doesn't seem to wait for children.

Mark for task 1.1.2: 3.0 out of 3.0

Task 1.1.3: Before-and-after images

hide -s implemented?	☐ No ☑ **Yes**
Shows "before" image	☐ Poor ☑ **Okay** ☐ Good ☐ N/A
Shows "after" image	☐ Poor ☑ **Okay** ☐ Good ☐ N/A

Task 1.1.3 Comment: Calls update surface once per pixel, so it takes a long time to draw image.
Puts one window on top of other, but doesn't call update surface later, so surface may not be seen.
No need for forking here, or separate executable.

Mark for task 1.1.3: 1.0 out of 2.0
Mark for task 1.1: 6.5 out of 8.0

Task 1.2: I/O and command line parameters

Handles command line parameters as per spec	☐ Poor ☐ Okay ☑ **Good**
Accepts message from stdin	☐ Poor ☐ Okay ☑ **Good**
Correct use of file functions	☐ Poor ☐ Okay ☑ **Good**
Closes all files opened	☐ No ☐ Some ☑ **All**

Task 1.2 Comment: None
Mark for task 1.2: 1.0 out of 1.0

Task 1.3: Submission guidelines

Tarfile submitted	☐ No ☑ **Yes**
README contains required information	☐ No ☐ Some ☑ **All**
Makefile submitted	☐ No ☑ **Yes**
Makefile builds both hide and unhide properly	☐ Poor ☐ Okay ☑ **Good** ☐ N/A
tarfile extracts properly	☐ Poor ☐ Okay ☑ **Good** ☐ N/A

Task 1.3 Comment: None
Mark for task 1.3: 1.0 out of 1.0
Mark for task 1: 8.5 out of 10.0

Task 2: Code quality

Task 2.1: Error handling

Handles invalid command line parameters	☐ Poor ☐ Okay ☑ **Good**
Checks for error returns from file I/O calls	☐ Poor ☑ **Okay** ☐ Good

Handles truncated image file	☐ Poor ☑ **Okay** ☐ Good
Handles PPM files with invalid headers	☐ Poor ☐ Okay ☑ **Good**
Handles messages that are too long for image	☐ Poor ☐ Okay ☑ **Good**

Task 2.1 Comment: Doesn't always check return values from file IO calls for errors.
Doesn't detect truncated image files, encodes message anyway if it will fit.

Mark for task 2.1: 2.0 out of 3.0

Task 2.2: Memory management

Checking for null pointers before access	☐ Poor ☑ **Okay** ☐ Good
Checking array bounds	☐ Poor ☐ Okay ☑ **Good**
All memory allocated is freed	☐ No ☐ Some ☑ **All** ☐ N/A
Any allocated memory is of an appropriate size	☐ No ☐ Some ☑ **All** ☐ N/A

Task 2.2 Comment: Doesn't check all pointers for null before use.

Mark for task 2.2: 2.5 out of 3.0

Task 2.3: Readability/usability

Inline comments	☐ None ☐ Uninformative/insufficient ☑ **Good**
Function header comments	☐ None ☐ Uninformative/insufficient ☑ **Good**
File header comments	☐ None ☐ Uninformative/insufficient ☑ **Good**
README is readable	☐ Poor ☐ Okay ☑ **Good**
Are identifier names well-chosen?	☐ No ☐ Some ☑ **All**
Functions are an appropriate size	☑ **Too long** ☐ Too short ☐ Just right
Structure of .c files	☐ Poor ☐ Okay ☑ **Good**
Use of header files	☐ Poor ☐ Okay ☑ **Good**

Task 2.3 Comment: Some functions could be shorter.

Mark for task 2.3: 2.5 out of 3.0

Task 2.4: Process handling

Does not fork() excessively	☐ Poor ☐ Okay ☑ **Good** ☐ N/A
Parent process waits for all child processes	☑ **Poor** ☐ Okay ☐ Good ☐ N/A

Use WNOHANG!

wait(&status) waits for any process?

Task 2.4 Comment: Code has wait(), but parent doesn't run it, instead terminates before children do.

Mark for task 2.4: 0.5 out of 1.0
Mark for task 2: 7.5 out of 10.0

Task 3: Penalty

Marks may be deducted for lateness or for breaches of the academic integrity policy. No marks may be deducted here without discussing it beforehand with a lecturer.

Task 3 Comment: None
Mark for task 3: -0.0 out of 20.0

Task 4: Bonus: using a directory

hide -d implemented?	☑ **No** ☐ Yes
Reads files from inputdir	☐ Poor ☐ Okay ☐ Good ☑ **N/A**
Writes files to outputdir	☐ Poor ☐ Okay ☐ Good ☑ **N/A**

Task 4 Comment: None
Mark for task 4: 0.0 out of 2.0

Total mark: 16.0