Open Source Software — CSCI-4961 — Summer 2019
Test 1
June 28, 2019

SOLUTIONS

1. Answer the following short questions (10 pts)

   (a) Eric Raymond wrote, "Before asking a technical question by e-mail, or in a newsgroup, or on a website chat board, do the following:" and then listed 7 actions. List any 4 of them (4/10 pts)

   1 Try to find an answer by searching the archives of the forum or mailing list you plan to post to.
   2 Try to find an answer by searching the Web.
   3 Try to find an answer by reading the manual.
   4 Try to find an answer by reading a FAQ.
   5 Try to find an answer by inspection or experimentation.
   6 Try to find an answer by asking a skilled friend.
   7 If you're a programmer, try to find an answer by reading the source code.

   (b) Name the RPI student who was sued by the RIAA for copyright infringement (1/10 pts)

   Jesse Jordan (Aaron Sherman works as well, although Aaron was not mentioned in our reading.)

   (c) Given the following regex expression `^[a-zA-Z](.*)\.(html|jpg)$`, indicate **Match** if the string would be a match, or **No Match** if it would not. (5/10 pts)

   1 test.html **Match**

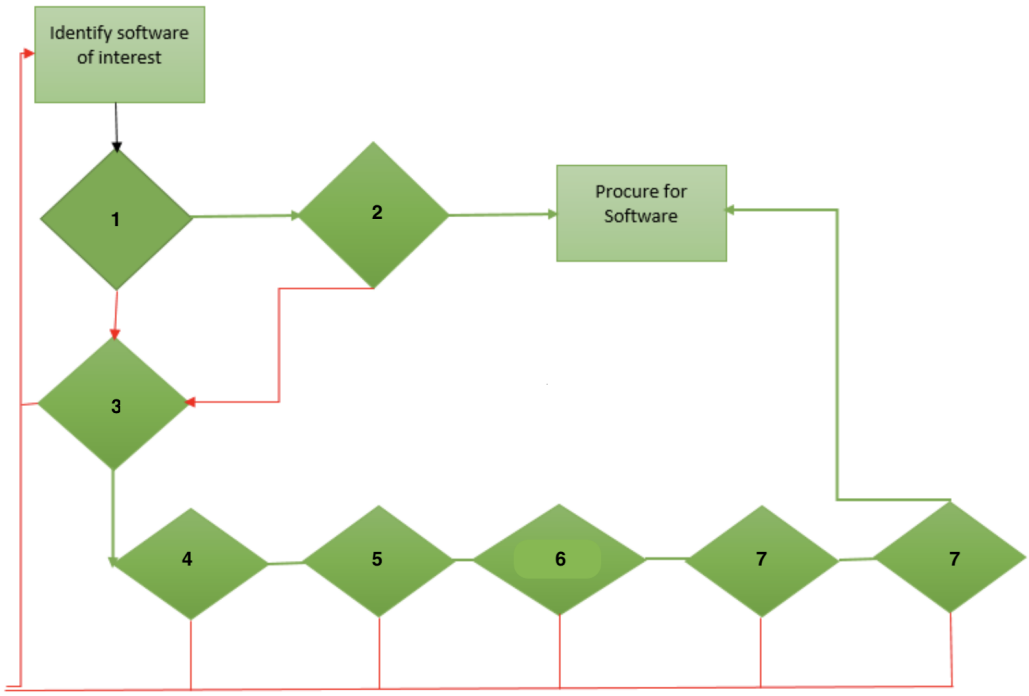
   2 2a38588hfquh.jpg **No Match**

   3 jpg.r **No Match**

   4 a@p239552u7894aghe.jpg **Match**

   5 htmljpg **No Match**

2. The diagram below represents a normal (non-open source) software procurement cycle. Fill in the diagram with the terms needed to complete it. (We know that 7 appears 2 times.)(14 pts)



1 Is software for sale?

2 Does standard license meet needs?

3 Will vendor negotiate?

4 Meet scope?

5 Meet term?

6 Renewal?

7 Condition?

3. The Open Source Initiative defines 10 characteristics of open source software. For each of the following actions, indicate if the action is allowed or prohibited and the characteristic which allows or prohibits it. (15 pts)

   For example, for:

   Require a recipient of a derived work to contact the original author to get a license the answer would be:

   Prohibited, Distribution of License clause

   Each correct answer is worth 3 points with the **Allowed** or **Prohibited** worth 1 point and identifying the appropriate characteristic worth 2 points.

   (a) The author of a source code distribution allows the original code to be freely distributed requiring only attribution for the original authors and inclusion of the copyright notice, but requires that any derived works be distributed under an AGPL license. (Original license is not AGPL)

   Prohibited, Derived Works

   (b) A user (recipient) gets a USB distribution of an open source family of products, all of the code on the USB is related and licensed simlarly. The user extracts one program from the USB and redistributes it as an independent program.

   Allowed, License Must Not Be Specific to a Product

   (c) An author requires that her source be maintained unchanged in any derivative works, but allows recipients to use *git patch* files to implement and distribute modifications to the code.

   Allowed, Integrity of the Author's Source Code

   (d) An author distributes source under a BSD license, but adds a clause that prohibits the software from being used by the military or by nuclear power plants.

   Prohibited, No Discrimination Against Persons or Groups

   (e) A user downloads the source code of several open source project from repositories on github onto a $4.00 USB and then sells the offers the USB for sale on EBAY for $10000 without paying or offering remuneration to the original author.

   Allowed, Free Redistribution

4. Define the following types of intellectual property rights (9 pts):

   (a) Copyright Form of intellectual property, applicable to any expressed representation of a creative work, that grants the creator of an original work exclusive rights to its use and distribution, usually for a limited time.

   (b) Patent A set of exclusive rights granted by a sovereign state to an inventor or assignee for a limited period of time in exchange for detailed public disclosure of an invention.

   (c) Trademark A recognizable sign, design, or expression which identifies products or services of a particular source from those of others.

5. Write LaTeX code to duplicate the document below. You can assume the photo is "images/pelican.jpg". Your text should go on the next page. (12 pts):

1 First Section

We have some text with an an inline equation $x = \sum_0^{100} x^2$. It extends on until it wraps around.

We have a new paragraph. Extend it to wrap as well. Below is a picture of a pelican.

And of course we need a table:

1	middle	left
2	middle	left
3	middle	left

```
\documentclass{article}
\usepackage[pdftex]{graphicx}
\begin{document}
\section{First Section}
We have some text with an an inline equation $x = \sum_0^{100} x^2$. It extends on until it wraps
    around.

We have a new paragraph. Extend it to wrap as well. Below is a picture of a pelican.

%scaling doesn't matter
\includegraphics[scale=0.10]{images/pelican.jpg}

And of course we need a table:

\begin{tabular}{ccc}
1& middle   & left \\
2& middle & left \\
3 & middle & left \\
\end{tabular}

\end{document}
```

6. Consider the following scenario. You are working in a "blessed" repository configuration. On your laptop you have a clone of your fork with two remotes set up. The remote **origin** points to your forked repository and **upstream** points to the blessed repositiory. All three repositories are out of synch. You can assume the blessed repository *(upstream)* has:

   - a new file "d.cxx"

   and your fork *(origin)* has:

   - a new file "e.cxx".

   Your local repository on your laptop has

   - a new file named "a.cxx" and
   - two modified files "b.cxx" and "c.cxx".

   Your goal is to get all of these changes in the master branch in your fork *(origin)* so you can make a pull request to the blessed repository. Give the sequence of git commands required to reach the state where all changes are present in your fork and you are ready for the pull request. (15 pts)

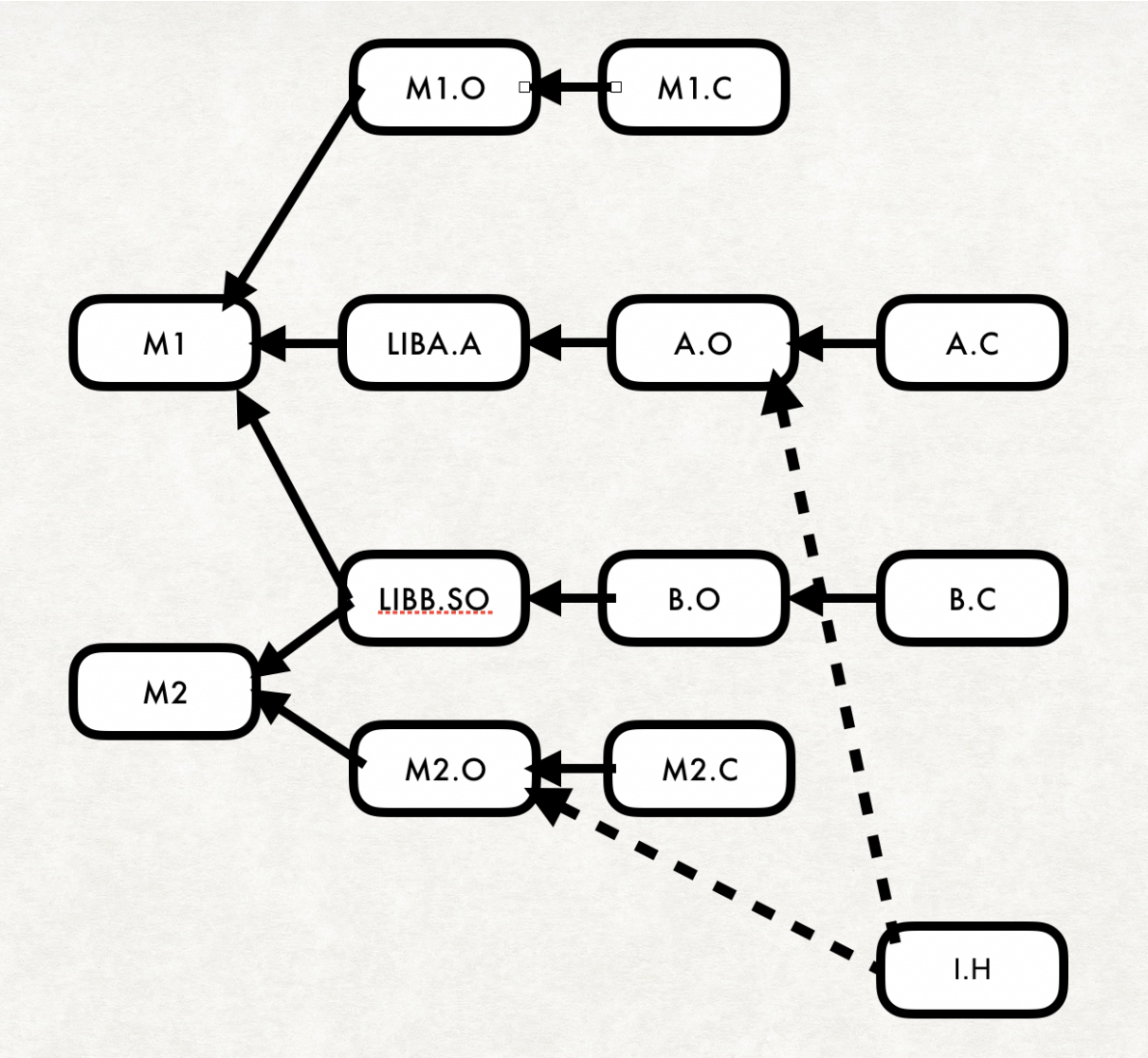
   Write your git commands below

```
git add a.cxx
git add -u
git commit -m "Committing the local changes"
git pull origin master
git pull upstream master
git push origin master
```

7. Consider the "Makefile" shown below.

```
m1: m1.o liba.a libb.so
   cc m1.o liba.a libb.so -o m1 -Wl,-rpath .
m2: m2.o libb.so
   cc m1.o libb.so -o m1 -Wl,-rpath .
m1.o: m1.c
   cc -c m1.c -o m1.o
m2.o: m2.c
   cc -c m1.c -o m1.o
libb.so: b.o
   cc -shared -o libb.so b.o
liba.a: a.o
   ar qc liba.a a.o
b.o: b.c
   cc -fPIC -c b.c -o b.o
a.o: a.c
   cc -c a.c -o a.o
m2.o: i.h
a.o: i.h
```

(a) Draw the dependency graph that depicts the Makefile below. Use dotted lines for implicit dependencies. (10 pts)



(b) Makefiles should always have "all" and "clean" targets. Write the dependencies and commands for all and clean below (5pts):

```
all: m1 m2
clean:
   rm *.o *.so hi1 hi2
```

(c) Now turn the Makefile into a CMakeLists.txt file. Add the commands to generate an install target to install "m1" and "m2" into "bin" and the shared library into "lib" (10 pts):

```
cmake_minimum_required(VERSION 3.0)
project(Hello C)

add_library(b SHARED b.c)
add_library(a STATIC a.c i.h)

add_executable(m1 m1.c)
target_link_libraries(m1 a b)

add_executable(m2 m2.c i.h)
target_link_libraries(m2 b)
```

```
install(TARGETS m1 DESTINATION bin)
install(TARGETS m2 DESTINATION bin)
install(TARGETS b DESTINATION lib)
```
