

# Midterm Test - CSC301H5F, Fall 2014

Friday, Oct 24, 2014

Duration: 50 minutes  
Instructor: Joey Freund  
TA: Saad Ilyas

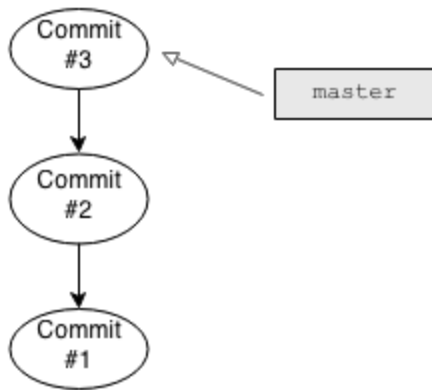
Please fill in your information.

<b>Student Name:</b>	
<b>Student Number:</b>	
<b>UTor ID:</b>	
<b>GitHub username:</b>	

<b>Q1</b>	<b>/ 12</b>
<b>Q2</b>	<b>/ 9</b>
<b>Q3</b>	<b>/ 12</b>
<b>Q4</b>	<b>/ 15</b>
<b>Q5</b>	<b>/ 27</b>
<b>Total Mark</b>	<b>/ 75</b>

## Question 1 (12 points)

Consider a local repo with the following graph of commits:



We open a terminal, `cd` into the root directory of this local repo, and see that it contains only a single regular file, `a.txt`.

(a) We continue by running the following commands:

```
echo "Hello" > b.txt
```

```
git add --all
```

```
git commit -m "Committing b.txt"
```

What does the graph of commits look like at this point?

(b) We continue by running the following commands:

```
git checkout -b new_feature
echo "Hello" > c.txt
git add --all
git commit -m "Committing c.txt"
git checkout master
git merge new_feature
```

What does the graph of commits look like at this point?

(c) We continue by running the following commands:

```
echo "Hello" > d.txt
git add --all
git commit -m "Committing d.txt"
```

What does the graph of commits look like at this point?

(d) We continue by running the following commands:

```
git checkout new_feature  
echo "Goodbye" > d.txt  
git add --all  
git commit -m "Committing d.txt"
```

What does the graph of commits look like at this point?

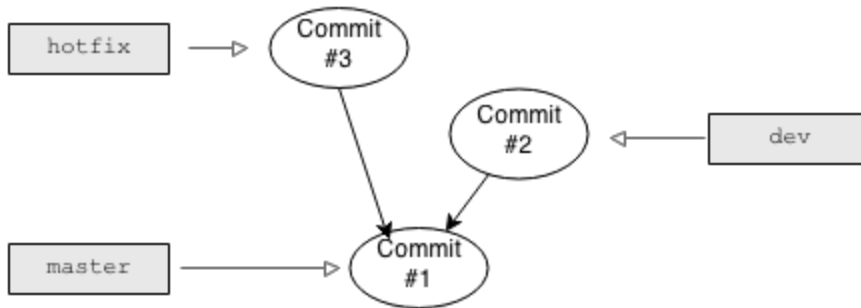
## Question 2 (9 points)

Consider a local repo with the following graph of commits:

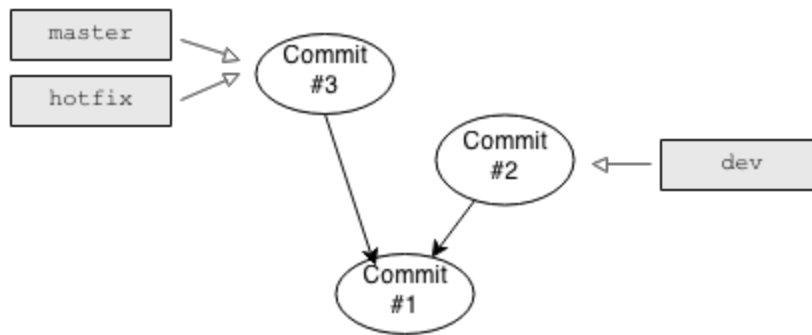


We open a terminal, `cd` into the root directory of this local repo, and see that it contains only a single regular file, `a.txt`.

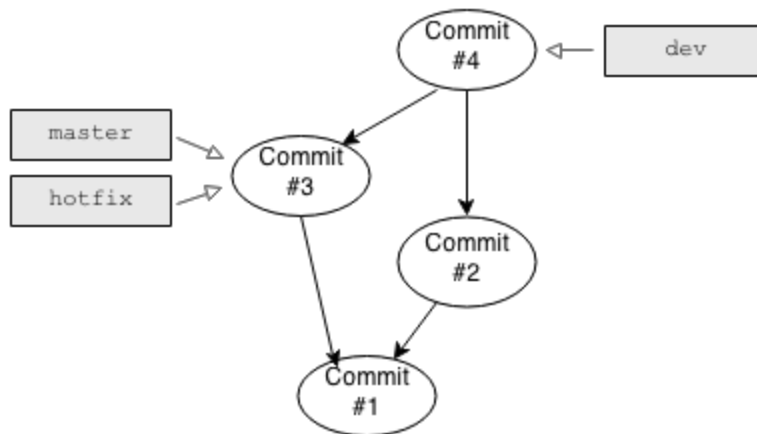
(a) Write a sequence of shell commands that will result in the following commit graph.



(b) Continue the sequence of shell commands to result in the following commit graph.



(c) Continue the sequence of shell commands to result in the following commit graph.



### Question 3 (12 points)

We run the following commands (without any errors):

```
cd /usr/home/my-repo
git init
```

We continue by running a few more shell commands, where

- Each command is one of the following:
  - A command that adds/modifies a regular text file.
  - `git add --all`
  - `git commit -m "Committing some changes"`
- We may run the commands in any order.
- We may run a command any number of times (including zero).
- All commands run without any errors.

After running the commands, the **working-directory and index** of our repository **are the same**, but **the history is different**.

For each statement below, indicate whether it **must be** true, **must be** false, or could be either.

Statement	True	False	Either
The directory <code>/usr/home/my-repo</code> is empty.			
The directory <code>/usr/home/my-repo</code> contains at least two files.			
We ran the <code>git commit</code> command.			
The last command we ran was the <code>git commit</code> command.			
We ran the <code>git add</code> command.			
The last command we ran was the <code>git add</code> command.			

## Question 4 (15 points)

In class, we learned about user stories, and how we use them to design a software product. We can use user-stories to design a non-software product as well.

Consider the following simple physical product - A protective case for a smartphone.

Write 3 of the **highest priority** user-stories for such a product.

For each story, don't forget to mention its benefit (i.e. Why does the user want this story?).

As a user, I want \_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

As a user, I want \_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

As a user, I want \_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_



## Question 5 (27 points)

Your team came up with the the following CRC cards for some system:

Product		StoreInventory	
<ul style="list-style-type: none"> <li>• Knows its product identifier.</li> <li>• Knows its name.</li> <li>• Knows its price.</li> <li>• Knows which store inventories have this product in stock.</li> </ul>	StoreInventory	<ul style="list-style-type: none"> <li>• Knows its store identifier.</li> <li>• Given a product identifier, knows how many units of this product are in stock.</li> </ul>	

(a) Your manager realized that this design does not satisfy one of the requirements - A product might have different prices at different stores.

Show a new CRC card design that satisfies this requirement.

Try to keep classes cohesive and avoid unnecessary dependencies.

(b) Your teammate noticed that the `Product` class depends on `StoreInventory`, and asked you to remove this dependency.

Update your CRC card design from part (a) to satisfy this new requirement.

Try to keep classes cohesive and avoid unnecessary dependencies.

(c) Play out the following scenario, using your CRC cards from part (b):

“Given a *Product*, I want to know which stores have this product in stock”