Student name: \_\_\_\_\_\_\_\_Aashis Rimal\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Student ID: \_\_\_\_\_\_\_\_\_\_\_1304965\_\_\_\_\_\_\_\_\_\_\_\_

Score: \_\_\_\_\_\_ /40 (Please do not fill this field)

**For easier assessment, please write your answers in the following table:**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **1** | **C** | **2** | **A** | **3** | **D** | **4** | **B** | **5** | **A** |
| **6** | **A** | **7** | **A** | **8** | **B** | **9** | **D** | **10** | **D** |
| **11** | **B** | **12** | **B** | **13** | **B** | **14** | **A** | **15** | **D** |
| **16** | **C** | **17** | **C** | **18** | **A** | **19** | **C** | **20** | **C** |
| **21** | **D** | **22** |  | **23** | **B** | **24** | **A** | **25** | **C** |
| **26** | **C** | **27** | **D** | **28** | **D** | **29** | **A** | **30** | **A** |
| **31** | **B** | **32** | **B** | **33** | **B** | **34** | **D** | **35** | **A** |
| **36** | **A** | **37** | **D** | **38** | **B** | **39** | **B** | **40** | **A** |

|  |
| --- |
| **Note:**  You are not allowed to use class materials and any other materials.  Quiz has to be done individually. |

**Time: 2 hour**

**1. What does VCS stand for?**

A. Verified Controlled System

B. Version Continuous Software

C. Version Control System

D. Very Curious Software

**2. What does CI stands for?**

A. Continuous Integration

B. Continuous Implementation

C. Core Integration

D. Core Implementation

**3. Which one is NOT one of the characteristics of Version Control?**

A. Tracking Changes

B. Teamwork Collaboration

C. Faster Development

D. Slower Development

**4. Git license is \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_.**

A. Premium

B. Free

C. Free for developers and premium for organizations

D. It depends on the scale of projects.

**5. Does Git provides a complete long-term change history of every file?**

A. Yes

B. No

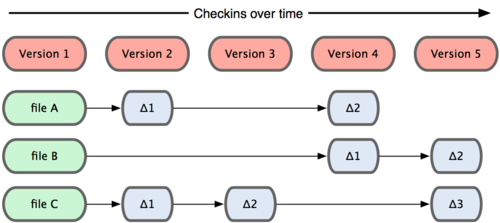
**6. Can we create different branches of our project in Git?**

A. Yes

B. No

C. Limited to the type of projects programming languages

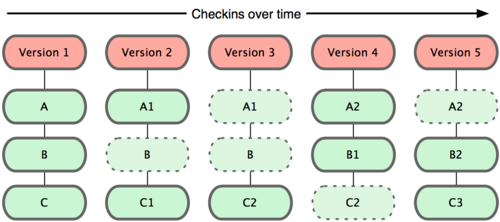
**7. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ store(s) information as a list of file-based changes.**



A. Most other version control systems

B. Git

**8. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ think(s) of its data more like a set of snapshots of a mini filesystem.**

****

A. Most other version control systems

B. Git

**9. In which platform we cannot install and use Git?**

A. Windows

B. Mac

C. Linux

D. In all the mentioned platforms we can install and use Git

**10. For which one of the targeted groups, Git cannot be useful?**

A. Developers

B. Designers

C. Marketing / Project Manager / Customer Service

D. Git is useful for all the mentioned target groups

**11. What is not one of the main features that Git repository hosting services provide?**

A. Git

B. Development suggestions

C. Issue Tracker

D. Code Review

**12. The \_\_\_\_\_\_\_\_\_\_\_\_\_\_ command creates a new Git repository. It can be used to convert an existing, unversioned project to a Git repository or initialize a new empty repository.**

A. git add

B. git init

C. git branch

D. git config

**13. The \_\_\_\_\_\_\_\_\_\_\_\_\_\_ command copies an existing Git repository.**

A. git pull

B. git clone

C. git add

D. git branch

**14. The \_\_\_\_\_\_\_\_\_\_\_\_\_\_command lets you configure your Git installation (or an individual repository) from the command line. This command can define everything from user info to preferences to the behavior of a repository.**

A. git config

B. git branch

C. git stash

D. git status

**15. The \_\_\_\_\_\_\_\_\_\_\_\_\_\_ command adds a change in the working directory to the staging area. It tells Git that you want to include updates to a particular file in the next commit.**

A. git merge

B. git clone

C. git commit

D. git add

**16. The \_\_\_\_\_\_\_\_\_\_\_\_\_\_ command commits the staged snapshot to the project history.**

A. git merge

B. git clone

C. git commit

D. git add

**17. \_\_\_\_\_\_\_\_\_\_\_\_\_\_ temporarily shelves changes you‘ve made to your working copy so you can work on something else, and then come back and re-apply them later on.**

A. git reset

B. git revert

C. git stash

D. git commit

**18. The \_\_\_\_\_\_\_\_\_\_\_\_\_\_ command displays the state of the working directory and the staging area. It lets you see which changes have been staged, which haven’t, and which files aren’t being tracked by Git.**

A. git status

B. git config

C. git log

D. git stash

**19. The \_\_\_\_\_\_\_\_\_\_\_\_\_\_ command displays committed snapshots. It lets you list the project history, filter it, and search for specific changes.**

A. git status

B. git config

C. git log

D. git stash

**20. The \_\_\_\_\_\_\_\_\_\_\_\_\_\_ command serves three distinct functions: checking out files, checking out commits, and checking out branches.**

**(So many hints! You really should pick the right choice for this question!)**

A. git add

B. git commit

C. git checkout

D. git branch

**21. The \_\_\_\_\_\_\_\_\_\_\_\_\_\_ command undoes a committed snapshot. But, instead of removing the commit from the project history, it figures out how to undo the changes introduced by the commit and appends a new commit with the resulting content.**

A. git clean

B. git reset

C. git commit

D. git revert

**23. When you undo with \_\_\_\_\_\_\_\_\_\_\_\_\_\_ (and the commits are no longer referenced by any ref or the reflog), there is no way to retrieve the original copy—it is a permanent undo.**

A. git clean

B. git reset

C. git commit

D. git revert

**24. The \_\_\_\_\_\_\_\_\_\_\_\_\_\_ command removes untracked files from your working directory.**

A. git clean

B. git reset

C. git commit

D. git revert

**25. The \_\_\_\_\_\_\_\_\_\_\_\_\_\_ command is a convenient way to fix up the most recent commit. It lets you combine staged changes with the previous commit instead of committing it as an entirely new snapshot. It can also be used to simply edit the previous commit message without changing its snapshot.**

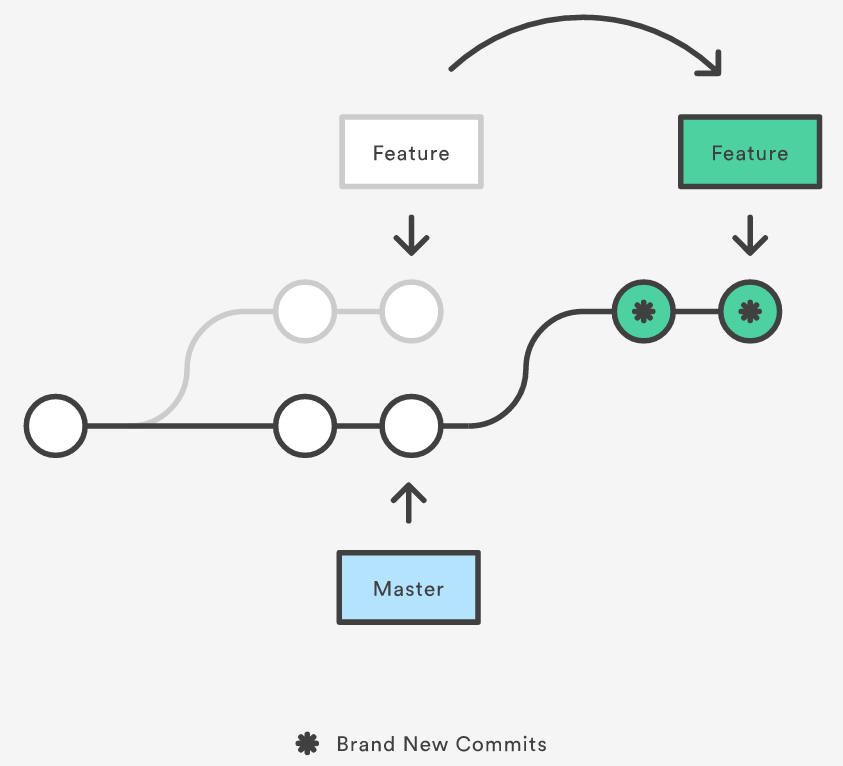
A. git pull

B. git push

C. git commit --amend

D. git checkout

**26. The \_\_\_\_\_\_\_\_\_\_\_\_\_\_ command is the process of moving a branch to a new base commit. The general process can be visualized as the following:**

****

A. git clone

B. git commit

C. git rebase

D. git stash

**27. The \_\_\_\_\_\_\_\_\_\_\_\_\_\_ command lets you create, view, and delete connections to other repositories.**

A. git pull

B. git push

C. git fetch

D. git remote

**28. When you clone a repository with git clone, it automatically creates a remote connection called \_\_\_\_\_\_\_\_\_\_ pointing back to the cloned repository.**

A. remote

B. first

C. master

D. origin

**29. The \_\_\_\_\_\_\_\_\_\_\_\_\_\_ command imports commits from a remote repository into your local repo. The resulting commits are stored as remote branches instead of the normal local branches that we’ve been working with.**

A. git pull

B. git push

C. git fetch

D. git remote

**30. The \_\_\_\_\_\_\_\_\_\_\_\_\_\_ command is Git's version of svn update. It’s an easy way to synchronize your local repository with upstream changes.**

A. git pull

B. git push

C. git fetch

D. git remote

**31. The \_\_\_\_\_\_\_\_\_\_\_\_\_\_ command publishes your local changes to a central repository.**

A. git pull

B. git push

C. git fetch

D. git remote

**32. The \_\_\_\_\_\_\_\_\_\_\_\_\_\_ command lets you create, list, rename, and delete branches.**

A. git clone

B. git branch

C. git add

C. git stash

**33. The \_\_\_\_\_\_\_\_\_\_\_\_\_\_ command lets you navigate between the branches created by git branch.**

A. git branch

B. get checkout

C. git rebase

D. git stash

**34. The \_\_\_\_\_\_\_\_\_\_\_\_\_\_ command lets you take the independent lines of development created by git branch and integrate them into a single branch.**

A. git branch

B. git push

C. git pull

D. git merge

**35. Git lets you completely ignore files by placing paths in a special file called \_\_\_\_\_\_\_\_\_\_\_\_\_\_.**

A. .gitignore

B. .gitno

C. .gitremove

D. .githide

**36. A \_\_\_\_\_\_\_\_\_\_\_\_\_\_.merge can occur when there is a linear path from the current branch tip to the target branch.**

A. fast-forward

B. 3-way

**37. What is one of the responsibilities of using Continuous Integration development practice?**

A. Check in frequently

B. Don’t check in broken code

C. Don’t check in untested code

D. All of the above

**38. A \_\_\_\_\_\_\_\_\_\_\_\_\_\_ is a way to ask another developer to merge one of your branches into their repository.**

A. Merge

B. Pull Request

C. Clone

D. Push

**39. it’s generally not possible to push commits to an HTTP address. For read-write access, you should use \_\_\_\_\_\_\_ protocol instead.**

A. SMTP

B. SSH

C. HTTPS

D. FTP

**40. Give this point to yourself because you deserve it!**

A. Yes! I did great in this course and I deserve it!

B. Yes! I did great in this course and I deserve it!

C. Yes! I did great in this course and I deserve it!

D. Yes! I did great in this course and I deserve it!

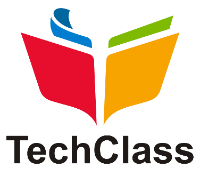
**You need to return this document for your final assessment.**

**Good luck!** ☺ **Farhad Eftekhari**

Copyright © 2016 by Farhad Eftekhari

All rights reserved. No part of this publication may be reproduced, distributed, or transmitted in any form or by any means, including photocopying, recording, or other electronic or mechanical methods, without the prior written permission of the publisher, except in the case of brief quotations embodied in critical reviews and certain other noncommercial uses permitted by copyright law. For permission requests, write to the publisher, addressed “Attention: Permissions Coordinator,” at the address below.

Helsinki Metropolia UAS  
Vanha maantie 6  
02650 Espoo, Finland  
www.techclass.co

  ****