+Instituto Superior de Formación Técnica Nº 151

Carrera: Analista de Sistemas

2 Año. Algoritmos y Estructuras de Datos II

Trabajo Práctico Nº 1.4-1.5	**Unidad 1**
Modalidad: Semi-Presencial	**Estratégica Didáctica:** Trabajo Individual.
Metodología de Desarrollo: acordar	**Metodología de Corrección:** acordar docente
Carácter de Trabajo: Obligatorio – con Nota	**Fecha Entrega:** A confirmar por el Docente.

Marco Teórico:

Introducción a Sistemas de Subversionado

1. ¿Cómo se desarrolla o que aspectos hay que tener en cuenta cuando trabajamos en sistemas Colaborativos?
2. ¿Porque es importante la Inclusión de un sistema de Subversionado?
3. ¿Qué es Git que características tiene?
4. Que SVN y que permite
5. Establecer las Diferencias entre Git y SVN
6. ¿Cuándo recomendaría usar Git o SVN?

GIT

7. Que es GT y que es Representa Un flujo de Trabajo
8. Describir y Ejemplificar el Uso de;
    a. Working Directory Area
    b. Stagin Area
    c. Repository Area
    d. Que rol representa GiyHub en Remote Area
9. Que entiende por Branch (Ramas)
10. Que es Tags o sistema de Versionado

GitHub

11. Que es GitHub? ¿Que Permite o Facilita?
12. Que entiende por un Repositorio GitHub (¿o similar?)
13. Que es Push y Pull en GitHub? Que permite?
14. Que es clonar un Repo en GitHub
15. Proponer al menos 2 alternativas a GitHub que funciones con Git, describirlas?
16. Como se trabaja en entornos colaborativos?

Git & GitHub - Marco Practico:
Installs Needed tools

Install Visual Studio code

1. Instalar Visual Studio Code
    a. Descargar e Instalar: https://code.visualstudio.com/download#
    b. Install the extensión c++ para VC Code
        i. Open VS Code.
        ii. Select the Extensions view icon on the Activity bar or use the keyboard shortcut (Ctrl+Shift+X).
        iii. Search for 'C++'. (buscar el primero: “C/C++ms-vscode.cpptools”)
        iv. Select Install.
    c. Instalar el Compilador C++ (ejecuta “g++ --version” desde consola sin comillas) si responde omitir este paso, caso contrario descargar MinGW y reiniciar la Pc y continuar desde Aquí.
    d. Install code runner (Code Runnerformulahendry.code-runner) o danielpinto8zz6.c-cpp-compile-run
    (Run your code using Code Runner, Use the shortcut (Ctrl+Alt+N) Or press F1 and then select/type Run Code Or right-click the Text Editor and then click Run Code in the editor context menu)
    e. En VS Code -> File -> new fle luego Presiona Select Lenguage C++ (si esta todo Ok), caso contrario revisar que hay error en plugin.

Install Git

2. Descargar e Instalar Git:
    https://github.com/git-for-windows/git/releases/download/v2.31.1.windows.1/Git-2.31.1-64-bit.exe
    Aceptar las Opciones x Defecto.

3. Agregar Git path al Ambiente:
    To add into PATH:
    Right-Click on My Computer.
    Click on Advanced System Settings.
    Click on Environment Variables.
    Then, under System Variables, look for the path variable and click edit.
    Add the path to git's bin and cmd at the end of the string like this: ;C:\Program Files\Git\bin\git.exe;C:\Program Files\Git\cmd.

    process to refresh environment variables without reboot Windows
    open cmd commend prompt window.
    input set PATH=C -> this will refresh the environment variables.
    close and restart cmd window.
    input echo %PATH% to test

    Cerrar y Reiniciar Visual Studio code

GitHub

4. Conectarse a GitHub https://github.com/ y crear una cuenta.

Tutorial Git & GitHub – AyED II

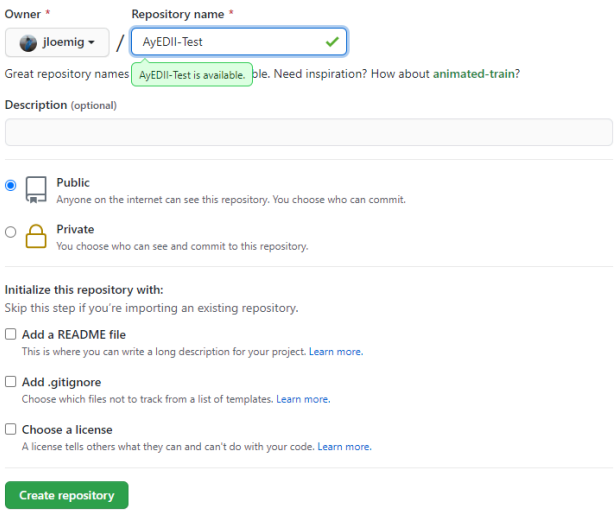
5. Arrancar Vs Code
6. Vamos a Open folder y abrimos la carpeta del proyecto.
7. Crear un “uno.cpp” (no importa el Código sino los Cambios)

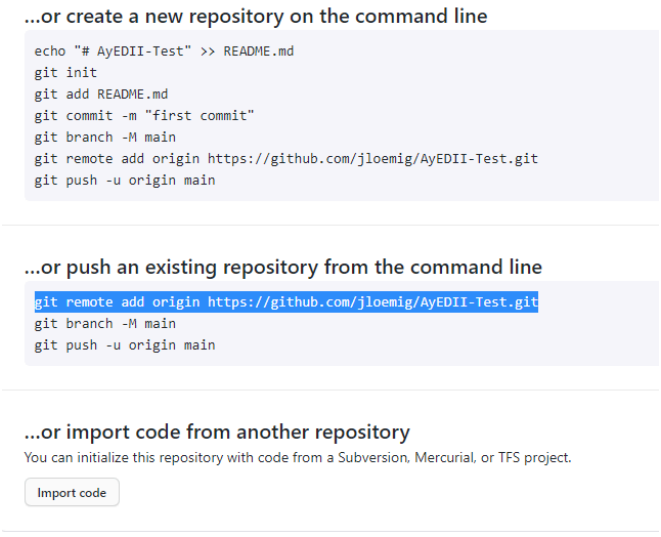
```
// Git & GitHub in C++ Program
#include <iostream>
int main() {
   std::cout << "Hello World!";
   return 0;
}
```

8. Guardarlo en una carpeta “exclusiva del Proyecto” (cleancode…)
9. Abrir la terminal de VC Code y moverse al directorio de “uno.cpp” IMPORTANTE
10. Ejecutar en la terminal -> git –version
11. Ejecutar en la terminal -> git init (solo una vez, nota que los archivos están en verde x q no tienen seguimiento…)
12. Ejecutar en la terminal -> git status –s (para ver el estado de los archivos **?? indica CRUD**
13. Ejecutar en la terminal -> git add uno.cpp (esto pasa del Work Dir-> stage área
14. Ejecutar en la terminal -> git status –s y git status –s (note que ahora dice **A** Esta Agregado al área temporal)
15. Ejecutar en la terminal -> git commit -m "Agregamos uno.cpp" (esto manda al local repo) Puede aparecer un mensje que pide: “Author identity unknown *** Please tell me who you are.” Si lo pide escibir: git config user.email jloemig@gmail.com y reintroducir git commit -m "Agregamos uno.cpp", ahora tenemos uno.cpp en la repo de Git ¡!!
16. Ejecutar en la terminal -> git status –s ,note que uno.cpp esta sin cambios, por que? Por que no se le hizo el git add uno.exe (note el cambio **uno.exe A**)
17. Ejecutar git status –s (note ahora ya no esta **?** sino **A**)
18. Ejecutar en la terminal -> git add . (agregame todos los archivos)
19. Ejecutar en la terminal -> git commit -m "Segundo Commit"
20. Ejecutar en la Terminal -> git log --oneline (muestra todos los commit, el que dice **HEAD** es el que estoy “parado” en el Local, noten que tenemos 2 Commit en juego, podemos “Viajar” es decir restaurar el archivo al commit que quiera, como….
21. Para volver el(os) Archivo(s) a un estado previo -> git reset --hard d594300
22. Ejecutar en la Terminal -> git log --oneline (muestra todos los commit,pero ahora cambio por le Reset)
23. Ejecutar en la Terminal -> git reset --hard d9dd314 vuelve al Master Anterior ¡!!

Github Lab

24. Conectarse a GitHub https://github.com/ e Iniciar Sesión
25. Crear un Repositorio (Ponerlo publico sin agregados)





26. Ejecutar en la Terminal -> git remote add origin https://github.com/jloemig/AyEDII-Test.git con esto enlazamos el Repo Remoto (Puede solicitar nombre y clave)
27. Ejecutar en la Terminal -> git push -u origin master para subir el Local repo
28. Crear dos.cpp
29. Ejecutar en la Terminal -> git add . (agregamos dos.cpp)
30. Ejecutar en la Terminal -> git status –s (ver dos.cpp en verde)
31. Ejecutar en la terminal -> git commit -m "Ahora en Cloud"
32. Ejecutar en la Terminal -> git push (subimos al GitHub)
33. Modificar Archivos en el GitHub
34. Ejecutar en la Terminal -> git pull (actualiza local repo)

GitHub Readme & Pull

35. Entrar a Github en la Repo que se creo y busca “adding a README.” Hacer un Commit ver que esta en GitHub pero no en git ni en la copia Local.
36. Ejecutar en la Terminal -> git pull (tre el readme al local)

Tags – Introducción a Versionado

37. Ejecutar en la Terminal -> git tag version.1.0 -m "Version 1.0" (crea Tag versiondo)
38. Ejecutar en la Terminal -> git log --oneline (ver que hay un tag en Repo Local)
39. Ejecutar en la Terminal -> git push (para subir archivos, los tag´s no se suben)
40. Ejecutar en la Terminal -> git push –tags (ahora si se suben los Tags de Versionado.)
41. git push origin version.1.0

Branch´s (ramas)

42. Ejecutar en la Terminal -> git log --oneline (verifico que estoy en el master)
43. Ejecutar en la Terminal -> git branch ramaAyEDII
44. Ejecutar en la Terminal -> git log --oneline (solo muestra)
45. Ejecutar en la Terminal -> git checkout ramaAyEDII (nos posiciona en la rama)
46. Ejecutar en la Terminal -> git branch (nos muestra las ramas y en la que estoy parado)
47. Modificar uno.cpp
48. Ejecutar en la Terminal -> git add .
49. Ejecutar en la Terminal -> git commit -m "Ahora Branch" estoy trabjndo en la rama que cree, el master esta desactualizado y debo hacer un “merge” para unirlo.
50. Ejecutar en la Terminal -> git checkout master (me muevo al master, solo desde ahí puedo hacer el merge. UPS!!! Desaparecio el cambio, lógico “Watson” estamos en el Master y esta Desactualizado.
51. Ejecutar en la Terminal -> git merge ramaAyEDII (hace merge desde ramaAyEDII -> Master, esto pisa lo del Master y lo funde con el Branch ¡!! Ver con git log --oneline
52. Ejecutamos desde la terminal -> git branch -d ramaAyEDII (borra la rama ¡!!!).
    Nota: como el Master no se Modifico, no sufrio conflicto!!! Vamos a hacer lio!!!.
53. Ejecutar en la Terminal -> git branch ramaAyEDII (creemos una rama)
54. Ejecutar en la Terminal -> git checkout ramaAyEDII (me muevo de rama)
55. Ejecutar en la Terminal -> git branch (para ver)
56. Modificar el Codigo.
57. Ejecutar en la Terminal -> git add .
58. Ejecutar en la Terminal -> git commit –m “Commit en ramaAyEDIII”
59. Ejecutar en la Terminal -> git checkout master (me muevo al master)
60. Ejecutar en la Terminal -> git brach (para ver)
61. Modificar el Archivo en master (no guardar el archivo ;-) vc code se da cuenta!!!)
62. Ejecutar en la Terminal -> git add.
63. Ejecutar en la Terminal -> git commit –m “commit desde el Branch”
64. Ejecutar en la Terminal -> git brach (para ver)
65. Ejecutar en la Terminal -> git merge ramaAyEDII (para mezclar)
66. Visualiza los cambios en VS Code que propone solución x q Git no sabe o delega la Responsabilidad?
67. Se debe “Aceptar cambio Actual” o “Aceptar Cambio entrante” eso es un conflicto
68. “Aceptar cambio entrante”
69. Ejecutar en terminal -> git status –s (note **UU** Union con Conflicto)
70. Ejecutar en la Terminal -> git add .
71. Ejecutar en la Terminal -> git commit –m “Conflicto Solucionado”
72. Conflicto Solucionado.

Apéndice:

Download Visual studio Code: https://code.visualstudio.com/download
Install c++ PlugIn: https://code.visualstudio.com/docs/languages/cpp
Debug: https://code.visualstudio.com/docs/cpp/cpp-debug
Code runner:
https://marketplace.visualstudio.com/items?itemName=formulahendry.code-runner

Lic. Oemig José Luis.