**RESUMEN LAB NO. 1: CONSTRUYENDO EL AMBIENTE DE DESARROLLO (VS CODE + GITHUB + PYTHON)**

Este es un resumen del material, dinámica y actividades asociadas con el primer laboratorio de la clase. Especifica el tiempo global, los objetivos y las actividades relacionadas.

# Objetivos

Crear un ambiente de trabajo distribuido propio del grupo, para los laboratorios, los retos y el proyecto del curso.

1. Entender los conceptos y aspectos generales de un ambiente de trabajo distribuido (Repositorio GIT, IDE VS Code, Python 3.7).
2. Administrar las versiones y ramas de código en un repositorio GIT (GitHub).
3. Ejecutar los procesos del repositorio GIT por medio de la consola y línea de comandos.
4. Implementar consultas básicas para procesar datos estructuras de datos.
5. Analizar los tiempos de ejecución de la consulta con diferentes volúmenes de datos.

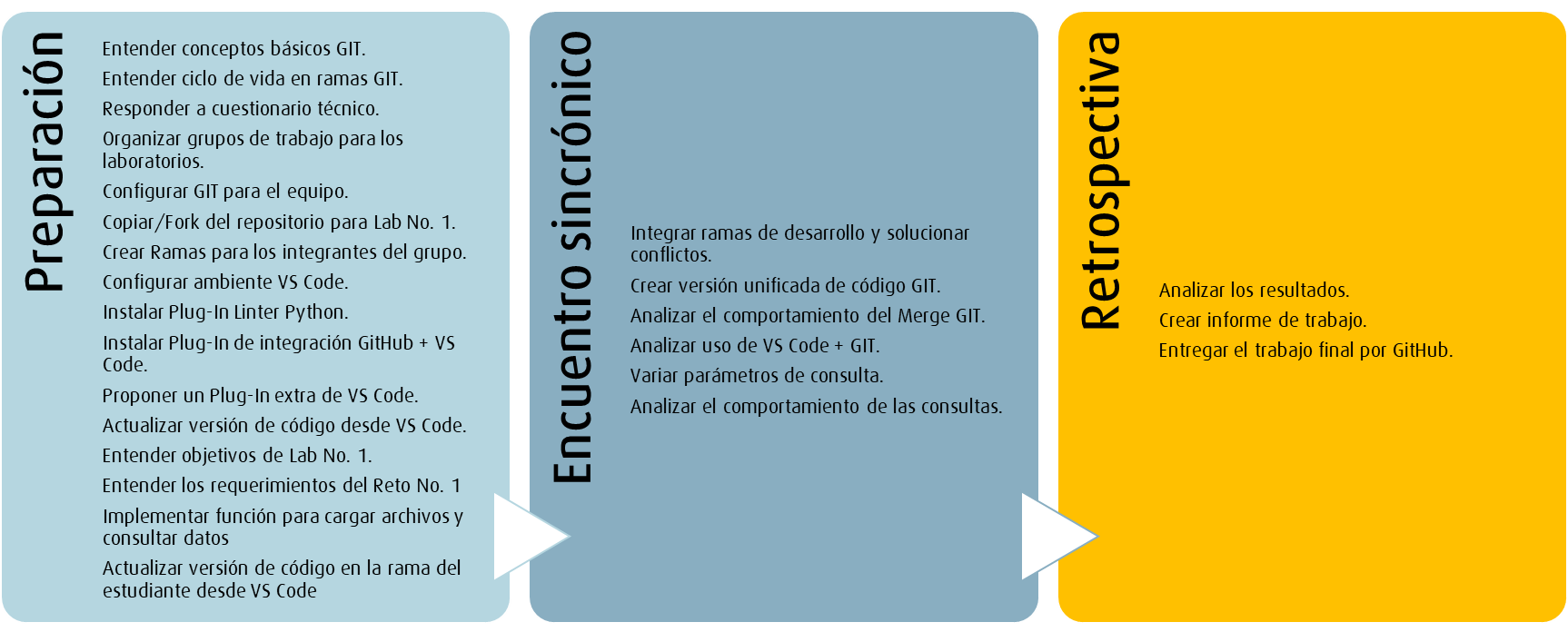
# Metodología

Las actividades del laboratorio No. 1 están planeadas para la primera semana del semestre. La dinámica se divide en tres etapas con recursos de Uniandes y de otras fuentes.

El trabajo preparatorio es principalmente individual, el encuentro sincrónico permite ver aclarar dudas y ver el estado de las tareas. Por último, la retrospectiva da el espacio para mostrar un entregable concreto a los monitores y profesores demostrando el nivel de apropiación de los conceptos.

# Desarrollo

La muestra Ilustración 1. Que el laboratorio 1 tiene tres etapas, una preparatoria antes del encuentro sincrónico, el encuentro del laboratorio y la retrospectiva donde se entrega el producto del trabajo. El laboratorio trata tres (3) temas globales. Primero, el manejo del repositorio GIT por medio de GitHub. Segundo, instalar y configurar el ambiente de desarrollo (IDE) Visual Studio Code (VS Code). Y tercero modificar, implementar y actualizar las funcionalidades del código proporcionado en el repositorio del laboratorio.



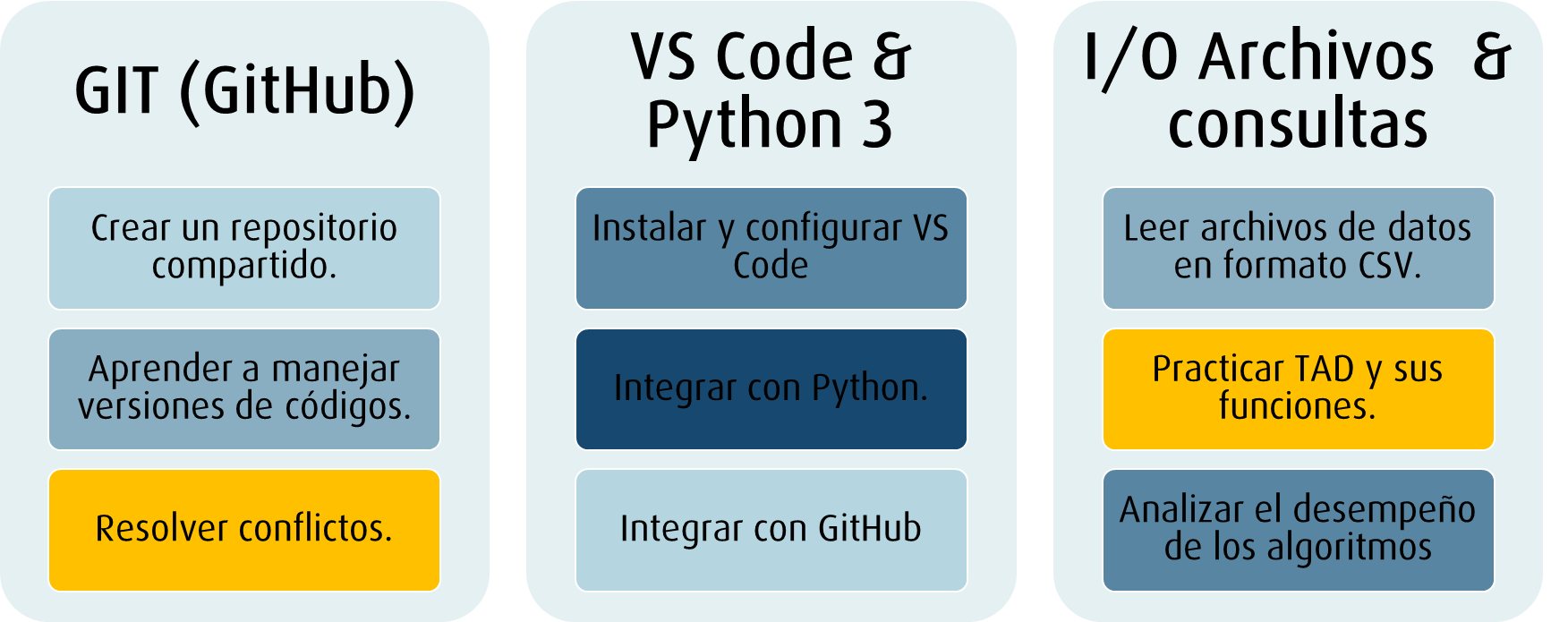


Ilustración 1. Resumen de temas y actividades generales para laboratorio No 1.

## preparación:

|  |  |  |
| --- | --- | --- |
| ACTIVIDAD PEDAGÓGICA | TEMA TRATADO | MATERIAL ASOCIADO |
| Entender conceptos básicos GIT | GIT + GitHub | * Video Uniandes para registrar GitHub e instalar GIT (Win10), URL: [https://web.microsoftstream.com/video/656bd7fe-9115-479d-b410-4a076e1ac974](https://web.microsoftstream.com/video/656bd7fe-9115-479d-b410-4a076e1ac974?channelId=fe9079a6-678f-4114-9c7e-8b6d7463803f) * Video Uniandes para registrar GitHub e instalar GIT (MacOS), URL: [https://web.microsoftstream.com/video/e790876c-7d7e-4738-8370-000b4a64cd2f](https://web.microsoftstream.com/video/e790876c-7d7e-4738-8370-000b4a64cd2f?channelId=fe9079a6-678f-4114-9c7e-8b6d7463803f) * Video Uniandes para Copiar/Fork repositorios GitHub, URL: [https://web.microsoftstream.com/video/9be3bd7c-26a8-4227-a0a9-e0d7043e0322](https://web.microsoftstream.com/video/9be3bd7c-26a8-4227-a0a9-e0d7043e0322?channelId=fe9079a6-678f-4114-9c7e-8b6d7463803f) * Videos y recursos extra sobre de GitHub y GIT:   + Fork a repo, URL: <https://docs.github.com/en/github/getting-started-with-github/fork-a-repo>   + Forking a GitHub Repository, URL: <https://www.youtube.com/watch?v=f5grYMXbAV0>   + Git & GitHub Tutorial for Beginners #12 - Forking (& Contributing), URL: <https://www.youtube.com/watch?v=HbSjyU2vf6Y>   + GitHub Tutorial 2020 - Beginner's Training Guide, URL: <https://www.youtube.com/watch?v=iv8rSLsi1xo> |
| Entender ciclo de vida en ramas GIT | GIT + GitHub | * Video Uniandes para crear y administrar ramas en GitHub, URL: [https://web.microsoftstream.com/video/db9799cb-b47c-44ab-b100-917514b649de](https://web.microsoftstream.com/video/db9799cb-b47c-44ab-b100-917514b649de?channelId=fe9079a6-678f-4114-9c7e-8b6d7463803f) * Videos y recursos extra sobre ciclo de vida y ramas GIT en GitHub:   + Managing branches, URL: <https://docs.github.com/en/desktop/contributing-to-projects/managing-branches>   + Learn Git from Scratch - How to Create Branches and push to Github, URL: <https://www.youtube.com/watch?v=pDmYNK68IEc>   + Creating a git branch, URL: <https://www.youtube.com/watch?v=tICdLNBJVFA> |
| Responder a cuestionario técnico | GIT + GitHub | * Cuestionario con preguntas de conceptos básicos GIT + GitHub, IMPORTANTE, revisar en las secciones oficiales del curso. |
| Organizar grupos de trabajo para los laboratorios | GIT + GitHub | * Instrucciones para organizar los grupos del laboratorio (ver 3.1.1) * Canal de Discord para comunicación informal entre estudiantes y monitores, IMPORTANTE, revisar en las secciones oficiales del curso. * Sesión de Zoom para romper el hielo (2 verdades y 1 mentira, rotando varias veces de grupos) IMPORTANTE, revisar en las secciones oficiales del curso. |
| Configurar GIT para el equipo | GIT + GitHub | * GitHub compartido entre los miembros de la organización * Video Uniandes para crear un equipo en GitHub, URL: [https://web.microsoftstream.com/video/0ba7d21e-d515-4fdb-bb68-d4abbb1e3582](https://web.microsoftstream.com/video/0ba7d21e-d515-4fdb-bb68-d4abbb1e3582?channelId=fe9079a6-678f-4114-9c7e-8b6d7463803f) * Videos y recursos extra sobre equipos, ramas y conflictos en GitHub y GIT:   + Inviting users to join your organization, URL: <https://docs.github.com/en/github/setting-up-and-managing-organizations-and-teams/inviting-users-to-join-your-organization>   + Git Tutorial - 25 - GitHub Organizations and Teams, URL: <https://www.youtube.com/watch?v=SUu6cmcC-3A>   + GitHub best practices, URL: <https://resources.github.com/videos/github-best-practices/>   + How To Add Collaborators To a Repository in GitHub, URL: <https://www.youtube.com/watch?v=sOciDss1EEY> |
| Copiar/Fork del repositorio para Lab No. 1 | GIT + GitHub | * GitHub con código actualizado en la rama Dev del estudiante * Video Uniandes para Copiar/Fork repositorios GitHub, URL: [https://web.microsoftstream.com/video/8f28c48f-5acf-4ef7-961c-eebb30ea3fdf](https://web.microsoftstream.com/video/8f28c48f-5acf-4ef7-961c-eebb30ea3fdf?channelId=fe9079a6-678f-4114-9c7e-8b6d7463803f) |
| Crear Ramas para los integrantes del grupo | GIT + GitHub | * Instrucciones para crear ramas de desarrollo GitHub para el grupo en el Lab No. 1 (ver 3.1.2) * GitHub con código actualizado en la rama Dev del estudiante |
| Configurar ambiente VS Code | VS Code | * Video Uniandes para Instalar VSCode y Configurar VS Code, URL: [https://web.microsoftstream.com/video/ad44bb24-8cc2-4876-81b2-500166128ed8](https://web.microsoftstream.com/video/ad44bb24-8cc2-4876-81b2-500166128ed8?channelId=fe9079a6-678f-4114-9c7e-8b6d7463803f) * Video Uniandes para Instalar Python 3.XX MacOS, URL: [https://web.microsoftstream.com/video/206921f5-3939-4a9d-8e19-560cdb31939e](https://web.microsoftstream.com/video/206921f5-3939-4a9d-8e19-560cdb31939e?channelId=fe9079a6-678f-4114-9c7e-8b6d7463803f) * Video Uniandes para Instalar Python 3.XX Windows, URL: [https://web.microsoftstream.com/video/34b2c85e-7498-42ea-a8e8-e4f472f26d21](https://web.microsoftstream.com/video/34b2c85e-7498-42ea-a8e8-e4f472f26d21?channelId=fe9079a6-678f-4114-9c7e-8b6d7463803f) * Videos y recursos extra sobre Python 3.XX y VS Code:   + Getting started with Visual Studio Code, URL: <https://code.visualstudio.com/docs/introvideos/basics>   + Setting up Visual Studio Code, URL: <https://code.visualstudio.com/docs/setup/setup-overview>   + VS Code - Setup and Basics, URL: <https://www.youtube.com/watch?v=SYRwSyjD8oI>   + Download the latest version for Windows, URL: <https://www.python.org/downloads/>   + Python in Visual Studio Code, URL: <https://code.visualstudio.com/docs/languages/python>   + Visual Studio Code (Windows) - Setting up a Python Development Environment and Complete Overview, URL: <https://www.youtube.com/watch?v=-nh9rCzPJ20>   + Getting Started with Python in VS Code, URL: <https://code.visualstudio.com/docs/python/python-tutorial>   + Editing Python in Visual Studio Code, URL: <https://code.visualstudio.com/docs/python/editing#_autocomplete-and-intellisense> |
| Instalar Plug-In Linter Python | VS Code | * Video Uniandes para Instalar plugin de Python (Pylinter + bandit) en VSCode, URL: [https://web.microsoftstream.com/video/04fcd88e-20fe-466b-b5da-89d2438c7ffe](https://web.microsoftstream.com/video/04fcd88e-20fe-466b-b5da-89d2438c7ffe?channelId=fe9079a6-678f-4114-9c7e-8b6d7463803f) * Videos y recursos extra sobre plugin de VS Code:   + Como configurar Visual Studio Code para PYTHON, URL: <https://www.youtube.com/watch?v=QtWry1YNLks>   + Linting Python in Visual Studio Code, URL: <https://code.visualstudio.com/docs/python/linting>   + Autocomplete and IntelliSense, URL: <https://code.visualstudio.com/docs/python/editing#_autocomplete-and-intellisense> |
| Instalar Plug-In de integración GitHub + VS Code | VS Code | * Instrucciones generales para instalar Plug-In GitHub + VS Code (ver 3.1.3), URL: [https://web.microsoftstream.com/video/e8a65263-aa16-4dd3-8291-f53a53258a84](https://web.microsoftstream.com/video/e8a65263-aa16-4dd3-8291-f53a53258a84?channelId=fe9079a6-678f-4114-9c7e-8b6d7463803f) |
| Proponer un Plug-In extra de VS Code | VS Code | * Instrucciones generales para buscar y proponer un Plug-In de VS Code interesante para el estudiante en el foro (Ver 3.1.4) * Recursos extra sobre Plug-In de VS Code:   + Python extension for Visual Studio Code, URL: <https://marketplace.visualstudio.com/items?itemName=ms-python.python>   + Extensions for the Visual Studio family of products, URL: <https://marketplace.visualstudio.com/vscode>   + The Python Podcast.\_\_init\_\_, URL: <https://open.spotify.com/show/5LblXHyyIfoGOL8s3ZcCxe>   + Talk Python To Me, URL: <https://talkpython.fm/>   + The Real Python Podcast, URL: <https://open.spotify.com/show/41Av6Rq81LfOT3Volz7W9D> |
| Actualizar versión de código desde VS Code | VS Code | * Enlace GitHub con código actualizado en la rama de desarrollo de cada estudiante. |
| Entender objetivos de Lab No. 1 | I/O Archivos | * Video Uniandes explicando la dinámica del Lab No. 1, URL: <https://web.microsoftstream.com/video/2973986b-63e2-4825-810d-158cb7dce19b> * Diapositivas del Laboratorio No. 1 |
| Entender los requerimientos reto 1 | I/O Archivos | * Video Uniandes explicando la dinámica del reto No. 1, URL: <https://web.microsoftstream.com/video/91eab175-1b5d-42de-a502-be8b7a711703> * Diapositivas de requerimientos del reto No. 1 |
| Implementar función para cargar archivos y consultar datos | I/O Archivos | * Instrucciones para modificar la función para cargar archivos y consultar datos (ver 3.1.5) |
| Actualizar versión de código en la rama desde VS Code | I/O Archivos | * Enlace GitHub con código actualizado en la rama Dev del estudiante |

### Instrucciones para organizar los grupos del laboratorio:

Para realizar los laboratorios los estudiantes deben conformar parejas de trabajo. En caso de ser necesario un grupo podrá ser de tres personas.

Para iniciar el trabajo del laboratorio 1 sigua los siguientes pasos:

1. Cree una organización para los laboratorios con el nombre de *EDA\_SEC\_<<Numero de la Sección>>\_EQUIPO<<Numero de equipo asignado>>.* Ej.: *EDA\_SEC\_02\_EQUIPO\_07* si es de la sección 2 de la clase y fue el séptimo grupo en registrarse.
2. Copie (Fork) el repositorio *“Lab0\_202020”* del repositorio oficial del curso a su grupo de trabajo en GitHub.
3. Edite el nombre de su repositorio a *“Lab1\_202020”.*
4. Invite a sus compañeros al repositorio del laboratorio/

Recuerden que para facilitar la dinámica del curso todos los estudiantes deberían:

1. Tener una cuenta habilitada en GitHub.
2. Trabajar individualmente y preparar las reuniones de grupo.
3. En caso de tener dudas comunicarse con los monitores por medio del foro, correo o Discord en los horarios habilitados.
4. Ser puntual con las entregas y compromisos adquiridos.
5. Ser respetuosos con los otros estudiantes y miembros del curso.
6. Mantener la honestidad en las entregas y trabajos asignados.

### Instrucciones para crear ramas de desarrollo GitHub para el grupo en el Lab No. 1:

El equipo de trabajo debe crear ramas de desarrollo en el repositorio del laboratorio para que cada uno de los estudiantes. Es decir, que en un grupo de 2 estudiantes deberían existir 3 ramas, la Master, estudiante 1 y estudiante 2.

Adicionalmente, la rama de cada estudiante debe ser nombrada con la identificación Uniandes, ej.: el estudiante con correo [sa-artea@uniandes.edu.co](mailto:sa-artea@uniandes.edu.co) deberá tener una rama en GitHub designada “sa-artea” donde trabajará y actualizará sus compromisos individuales previo al encuentro sincrónico.

### Instrucciones generales para instalar Plug-In GitHub + VS Code:

Después de instalados los Plug-In de VS Code para Python debe instalar algún Plug-In de integración de GitHub y VS Code desde el Marketplace VSC de Microsoft. Para ello se recomiendan cualquiera de las siguientes opciones:

1. GitHub Pull Requests and IssuesPreview (soportado por GitHub).
2. bandit de Linting Python (soportado por VS Code).

### Instrucciones generales para buscar y proponer un Plug-In de VS Code interesante para el estudiante en el foro:

Previo al encuentro virtual, cada estudiante debe buscar un Plug-In de VS Code de su interés y que considere ayude a desarrollar las actividades del curso. La publicación resultante en el foro deberá tener la siguiente información:

1. Nombre del Plug-In seleccionado.
2. Razón por la cual fue interesante.
3. Enlace del Marketplace VS Code.

En caso de que ya otro estudiante halla publicado el Plug-In de su interés, responder a la publicación con una razón mas de porque fue de su interés o alguna alternativa para este Plug-In.

### Instrucciones para modificar la función para cargar archivos y consultar datos:

Durante la sesión sincrónica del laboratorio se quiere ejecutar la aplicación con los datos del Reto No. 1. Para ello debe seguir las siguientes instrucciones:

1. Descargue los datos del reto No. 1 desde el enlace oficial en Sicua.
2. Descomprima el archivo y agréguelo en la carpeta “Data” del proyecto.
3. Modifique la función para cargar el archivo *(loadCSVFile)* en la opción 1 del menú de *app.py.*
4. Modifique la función *countElementsFilteredByColumn* en la opción 2 del menú de app.py para filtrar una de las columnas especificadas en el dataset del reto.
5. Ejecute de nuevo las operaciones de las opciones 1, 2 y 3 de la aplicación y compare los tiempos de carga con respecto al archivo *test.csv.*
6. Modifique la función *countElementsByCriteria* en la opción 2 del menú de *app.py* para cumplir con el requerimiento 1 del Reto No 1.

Desarrollando estas instrucciones y considerando el archivo de datos del Reto No. 1 reflexione sobre las siguientes preguntas ¿Cómo implementaría la función? ¿Qué orden de complejidad tendría la función implementada?

¿Cómo implementaría la función?

R/ Nuestro grupo no entendió inicialmente las inscricciones y la implementamos como tal. Para impletarla como criterio le pediamos al usuario el calificación minima que debía tener la pelicula; y como columna tomamos el nombre del director que se desea filtrar. Ya teniendo esto agregamos el ID de todas las peliculas con este director a una lista. Luego en la base de datos, buscabamos las peliculas que tuviesen dicho ID, y que tuvieran una calificación que cumpliese con el criterio (puntaje minimo). Usabamos un contador auxiliar para ver el numero de veces que sucedia esto.

¿Qué orden de complejidad tendría la función implementada?

R/ N \* (peliculas del director). Porque cada pelicula del director se evalua N veces

## encuentro sincrónico:

|  |  |  |
| --- | --- | --- |
| ACTIVIDAD PEDAGÓGICA | TEMA TRATADO | MATERIAL ASOCIADO |
| Integrar ramas de desarrollo y solucionar conflictos | GIT + GitHub | * Presentación de dinámica general de trabajo y convivencia en el laboratorio. * Instrucciones para integrar y actualizar la rama Master actualizada del código (ver 3.2.1) * Enlace GitHub con código actualizado en la rama Master * Video Uniandes para Integrar ramas de desarrollo y solucionar conflictos, URL: <https://web.microsoftstream.com/video/6cd0616f-b783-46ea-921b-02dd811506e9> * Videos y recursos extra de solución de conflictos Merge GitHub:   + Merging a pull request, URL: <https://docs.github.com/en/github/collaborating-with-issues-and-pull-requests/merging-a-pull-request>   + About merge methods on GitHub, URL: <https://docs.github.com/en/github/administering-a-repository/about-merge-methods-on-github>   + Resolving a merge conflict on GitHub, URL: <https://docs.github.com/en/github/collaborating-with-issues-and-pull-requests/resolving-a-merge-conflict-on-github>   + About merge conflicts, URL: https://docs.github.com/en/github/collaborating-with-issues-and-pull-requests/about-merge-conflicts   + How to solve git merge conflict, on Github or with terminal, URL: <https://www.youtube.com/watch?v=zOx5PJTY8CI>   + A Step by Step Guide for How to Resolve Git Merge Conflicts, URL: <https://www.youtube.com/watch?v=__cR7uPBOIk> |
| Crear versión unificada de código GIT | GIT + GitHub | * Enlace GitHub con código actualizado en la rama Master |
| Analizar el comportamiento del Merge GIT | GIT + GitHub | * Discusión en clase de la experiencia del Merge |
| Analizar uso de VS Code + GIT | VS Code | * Discusión en clase de la experiencia del Merge |
| Variar parámetros de consulta | I/O Archivos | * Enlace GitHub con código actualizado en la rama Dev del estudiante |
| Analizar el comportamiento de las consultas | I/O Archivos | * Discusión en clase del comportamiento del algoritmo |

### Instrucciones para integrar y actualizar la rama Master actualizada del código:

Durante el encuentro del laboratorio, los estudiantes ya deberían tener un grupo de trabajo y comenzado su trabajo individual en las ramas particulares de cada uno de ellos en el repositorio.

Durante este tiempo asignado los estudiantes deberán:

* Unificar o hacer Merge de cada una de las ramas con la rama Master.
* Solucionar conflictos de versión sobre las funciones de lectura de archivos y consulta de datos, esto incluye.
  + Seleccionar la mejor función de lectura del archivo entre los miembros del grupo (ej.: *loadCSVFile*).
  + Seleccionar la mejor función de consultas de datos entre los miembros del grupo (ej.: *countElementsFilteredByColumn*, *countElementsByCriteria*).
  + Mantener las otras versiones de la función de consulta como métodos alternativos (ej.: adicionar métodos countElementsFilteredByColumn1, countElementsFilteredByColumn2, etc.).

### Discusión en clase sobre GitHub, VS Code y Código implementado.

Después de practicar la integración de ramas y conflictos del laboratorio por favor considere y discuta las siguientes preguntas:

* ¿Cuál es el ciclo regular para actualizar código en un repositorio GIT?

R/Se clona el repositorio de Github.com y se va hacia VSCode y se hacen los cambios de cualquier forma y al finalizar se aplica el comando Git:Pull para actualizar cambios hechos por los cambios de mis compañeros; luego uso Git:Stage All Changes para agregar mis cambios que yo realice, luego se usa Git: Commit para decir y señalar los cambios que yo realice y por ultimo usar Git:Push para actualizar en el repositorio todo lo realizado. Y apara confirmar ir a la pagina web y confirmar los cambios realizados.

* ¿Qué ventajas y limitantes tiene el uso de Ramas/Branches?

R/La limitante principal de las ramas/branches se mira a que es un procedimiento extenso para el más mínimo cambio efectuado, sin embargo para trabajos algo extensos y de muchos miembros seria muy fácil mirar los avances en conjunto.

* ¿Cuáles serían las buenas prácticas para solucionar conflictos?

R/ Es como un Google Drive, pero para editar codigo. Ofrece la ventaja de que un equipo que está trabajando simultaneamente en un mismo codigo, pueden trabajar en tiempo casi que real, en base al trabajo de sus demás compañeros.

Después de practicar y variar los parámetros de consultas de estructuras de datos considere y discuta las siguientes preguntas:

* ¿Qué orden de complejidad tendría las funciones (consulta y lectura de archivo)?

R/ Tendría un orden de complejidad N, debido a se tiene que hacer a=archivo.readline() N veces, y a su vez se hace a=archivo.split() N veces.

* ¿Cómo podría reducir o aumentar la complejidad de la consulta?

R/ Si al imprimir los datos, se impremen por columnas, el orden de complejidad aumentaria a N\*(# de columnas)

* ¿Cómo afecta un TAD en la complejidad?, ¿Qué alternativas existen?

Al usar un tipo abstracto de datos, el orden de complejidad aumenta porque se debe adaptar la estructura de datos de los datos en cuestion, para que funcionen en el TAD.

La alternativa sería no usar un TAD sino, funciones especificas a la estructura de datos que se está tratando, sin embargo esto tambien haría que se perdiesen muchas de las ventajas que traen los TAD en momentos como los de modificar el código.

## Retrospectiva:

|  |  |  |
| --- | --- | --- |
| ACTIVIDAD PEDAGÓGICA | TEMA TRATADO | MATERIAL ASOCIADO |
| Crear informe de trabajo | GIT + GitHub, VS Code, I/O Archivos | * Instrucciones para compartir el producto final con los evaluadores del Lab No. 1 (ver 3.3.1) * Enlace GitHub con código actualizado en la rama Master para Lab No. 1 basado en el Lab No. 0 * Invitación a los evaluadores (monitores y profesor) como colaboradores de la organización o proyecto. |

### Instrucciones para publicar producto final en el Foro habilitado para el Lab No. 1:

Para entregar exitosamente sus resultados de este laboratorio, por favor recuerde las siguientes indicaciones:

* Invitar al profesor y monitores del laboratorio asignados.
* Incluir en el README del repositorio los datos completos de los integrantes del grupo (nombre completo, correo Uniandes y código de estudiante).
* Entregar el enlace (URL) del repositorio por Sicuaplus.

Recuerden que cualquier documento solicitado durante las actividades debe incluirse en el repositorio GIT y que solo se calificara hasta el último **COMMIT** realizado dentro de las fechas límites.