**RESUMEN LABORATORIO NO. 2: LISTAS Y ORDENAMIENTOS BÁSICOS**

# Objetivos

Crear una implementación de estructura de datos tipo lista.

1. Implementar algoritmos de ordenamiento básicos.
2. Evaluar los tiempos de ejecución de los algoritmos implementados.
3. Desarrollar las pruebas para validar las operaciones sobre la estructura de datos.
4. Utilizar adecuadamente el ambiente de trabajo distribuido configurado.

# Desarrollo

El laboratorio dos (2) tienen dos (2) temas. Primero, crear un TAD lista (ADT List) utilizando la implementación del curso. Y segundo, utilizar algoritmos de ordenamiento (Sorting) básicos para ordenar elementos de una lista.

El resumen de las actividades y temas puede verse en la Ilustración 1, mientras que el material y recursos asociados a las actividades pueden consultarse en la sección 3 de Materiales & Recursos.

Las tareas específicas de esta práctica son:

* Estudiar el código del laboratorio No. 2 (ver sección 2.1).
* Completar el examen técnico del laboratorio no. 2 (ver sección 2.2).
* Implementar las pruebas en ADT List (ver sección 2.3).
* Modificar los ordenamientos básicos (ver sección 2.4).
* Integrar las implementaciones de listas y ordenamientos (ver sección 2.5).
* Analizar los resultados de listas y ordenamientos (ver sección 2.6).
* Compartir el producto final con los evaluadores (ver sección 2.7).

Por último, recuerde que lo aprendido en el laboratorio No. 1 servirá para configurar y desarrollar el trabajo de este.

## Estudiar el código del laboratorio No. 2:

Para familiarizarse con la implementación del código del laboratorio dos, lea rápidamente el código y documentación de las siguientes rutas:

* La implementación de lista en la dirección relativa *\*\Lab2\_202020\ADT\list.py*
* La implementación del arreglo y las listas encadenadas en las siguientes direcciones relativas:
  + Código en *\*\Lab2\_202020\DataStructures\arraylist.py*
  + Código en *\*\Lab2\_202020\DataStructures\singlelinkedlist.py*
  + Código en *\*\Lab2\_202020\DataStructures\listnode.py*
* La implementación de los algoritmos de ordenamiento en las siguientes direcciones relativas:
  + Código en *\*\Lab2\_202020\Sorting\selectionsort.py*
  + Código en *\*\Lab2\_202020\Sorting\insertionsort.py*
  + Código en *\*\Lab2\_202020\Sorting\shellsort.py*

## Completar el examen técnico del laboratorio No. 2:

Para responder al cuestionario técnico de los temas del laboratorio No. 2, diríjase a la página de evaluaciones de su sección en Sicuaplus.

## Implementar las pruebas en ADT List:

Después de familiarizarse con la estructura de ADT List y sus implementaciones de lista enlazada (Singlelinkedlist) y arreglo (Arraylist) implemente las pruebas Pytest con las siguientes instrucciones:

* Encontrar la carpeta de pruebas del código en la dirección relativa *\*\Lab2\_202020\Test\\*.*
* Crear una carpeta *lab2* con los archivos de prueba *test\_arraylistmovies.py* y *test\_singlelinkedlistmovies.py.*
* Copiar y pegar un archivo *config.py* en la carpeta lab2 para no tener problemas de ejecución.
* Completar las pruebas Pytest de listas utilizando los datos de los archivos \*-small.csv de películas.

## Modificar los ordenamientos básicos:

Después de familiarizarse con los algoritmos implementados de ordenamiento, siga los siguientes pasos para implementar las pruebas Pytest de *SelectionSort*, *InsertionSort* y *ShellSort.*

* Encontrar la carpeta de pruebas del código en la dirección relativa *\*\Lab2\_202020\Test\sorting\\*.* Y ubicar los archivos:
  + *test\_selectionsortbig.py*
  + *test\_shellsortbig.py*
  + *test\_insertionsortbig.py*
* Encontrar la carpeta de pruebas del código en la dirección relativa *\*\Lab2\_202020\Test\\*.*
* En la carpeta *lab2* crear los archivos de prueba *test\_selectionsortmovies.py*, *test\_insertionsortmovies.py* y *test\_shellsortmovies.py* copiando los archivos \*.py mencionados previamente*.*
* Modificar el código de la función ***less*** que se envía por parámetro a los algoritmos para que funcionen con los campos del archivo del reto No. 1 y pueda cumplir con el requerimiento 2 y 3.
* Crear una función para comparar llamada ***greater*** para que realice un ordenamiento descendente al ser utilizada como parámetro en los algoritmos.
* Completar las pruebas de ordenamiento para que se puedan utilizar los archivos de películas.

## Integrar las implementaciones de Listas y Ordenamientos

Después de completar las pruebas se debe extender la aplicación con los siguientes pasos:

* Extender el menú de la aplicación *(\*\Lab2\_202020\App\app.py)* y crear la opción 5 y modifique la función *orderElementsByCriteria*, la cual ordenará la lista utilizando las funciones de ordenamiento e imprimirá los tiempos de ejecución y cumplirá con el Req No 2. Del reto.
* Modificar *countElementsByCriteria* para cumplir con el requerimiento No. 3 del reto, de ser necesario puede adicionar más parámetros a la función.
* Comparar el desempeño de las implementaciones de Lista (arreglo y lista encadenada) al cargar los datos del reto registrando los tiempos promedios en la Tabla 1 utilizando las versiones \*-large.csv de los archivos.
* Comparar el desempeño de los tres tipos de ordenamientos contra las implementaciones de ADT List (arreglo y lista encadenada) al procesar los datos del reto registrando los tiempos promedios de ejecución en la Tabla 2 utilizando las versiones \*-large.csv de los archivos.

**IMPORTANTE:** Para obtener resultados consistentes para este análisis se recomienda ejecutar por lo menos 5 veces la prueba y promediar los resultados que de la Tabla 1. y Tabla 2.

|  |  |  |
| --- | --- | --- |
| Fuente de datos | Arraylist [ms] | Singlelinkedlist [ms] |
| Películas (MoviesDetailsCleaned) |  |  |
| Elenco (MoviesCastingRaw) |  |  |

* Tabla 1. Tabla de rendimiento para cargar archivos como lista enlazada y arreglo.

|  |  |  |
| --- | --- | --- |
| Algoritmo | Arraylist [ms] | Singlelinkedlist [ms] |
| SelectionSort |  |  |
| InsertionSort |  |  |
| ShellSort |  |  |

Tabla 2. Tabla de rendimiento para los distintos algoritmos de ordenamiento en listas enlazadas y arreglos.

## Analizar los resultados de listas y ordenamientos básicos:

Con los resultados de la Tabla 1. y Tabla 2 reflexione y responda las siguientes preguntas con respecto a las estructuras y algoritmos.

* ¿Qué diferencias se observan en desempeño de la carga de datos entre arreglo (*Arraylist*) y lista sencillamente encadenada (*Singlelinkedlist*)?
* ¿Cuál de las dos implementaciones (*Arraylist* y *Singlelinkedlist*) tiene mejor desempeño? Y ¿Por qué?
* ¿Qué diferencias existen entre cargar los archivos de películas (*MoviesDetailsCleaned*) y elenco (*MoviesCastingRaw*)?, ¿Por qué se presentan estas diferencias?
* ¿Qué diferencias en el desempeño se observan entre los tres algoritmos de ordenamiento?
* ¿Qué efectos tienen los dos tipos de lista en los tres algoritmos de ordenamiento?

## Compartir el producto final con los evaluadores:

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).
* Incluir la Tabla 1. y la Tabla 2 diligenciadas en un documento.
* Responder las preguntas de análisis en el documento Word (ver sección 2.6).
* Incluir en el documento los datos completos de los integrantes del grupo (nombre completo, correo Uniandes y código de estudiante).
* Incluir el documento en formato Word o PDF en el repositorio GIT.
* 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.

# Materiales & Recursos

Ilustración 1. Diagrama de actividades y temas generales para el laboratorio No 2.

## De Preparación:

|  |  |  |
| --- | --- | --- |
| ACTIVIDAD PEDAGÓGICA | TEMA TRATADO | MATERIAL ASOCIADO |
| Copiar/Fork del repositorio para Lab No. 2 | ADT List | * Repositorio GIT Laboratorio No. 2, URL: <https://github.com/ISIS1225DEVS/Lab2_202020> |
| Estudiar la implementación de ADT List y Datastructures | ADT List | * Repositorio GIT Laboratorio No. 2. * Instrucciones para estudiar el código del laboratorio No. 2 (ver sección 2.1) * Recursos y material adicionales:   + Abstract Data Type (ADT) In Python, URL: <https://medium.com/@tssovi/abstract-data-type-adt-in-python-33e6ce1f961e>   + Python Data Structures Tutorial, URL: <https://www.datacamp.com/community/tutorials/data-structures-python>   + Introduction to Linked Lists (Data Structures & Algorithms #5), URL: <https://www.youtube.com/watch?v=WwfhLC16bis&list=PLBZBJbE_rGRV8D7XZ08LK6z-4zPoWzu5H&index=5>   + Python Data Structures #2: Linked List, URL: <https://www.youtube.com/watch?v=JlMyYuY1aXU> |
| Implementar pruebas ADT List | ADT List | * Archivos CSV del Reto No. 1. * Repositorio GIT Laboratorio No. 2. * Instrucciones para implementar las pruebas ADT List (ver sección 2.3) * Recursos y material adicionales:   + Pytest: helps you write better programs, URL: <https://docs.pytest.org/en/stable/>   + Pytest Course, URL: <https://www.youtube.com/playlist?list=PLJsmaNFr5mNqSeuNepT3IaMrgzRMm9lQR>   + Effective Python Testing With Pytest, URL: <https://realpython.com/pytest-python-testing/> |
| Integrar objetivos del Laboratorio No. 2 con el Reto No. 1. | Ordenamientos básicos | * Video Reto No. 1, URL: <https://web.microsoftstream.com/video/91eab175-1b5d-42de-a502-be8b7a711703> * Video Laboratorio No 2, URL: <https://web.microsoftstream.com/video/cc8d3460-534e-46a4-a096-24ddc5dad2ff> |
| Estudiar los ordenamientos de listas. | Ordenamientos básicos | * Repositorio GIT Laboratorio No. 2. * Instrucciones para estudiar el código del laboratorio No. 2 (ver sección 2.1) * Recursos y material adicionales:   + Insertion Sort Algorithm Explained (Full Code Included) - Python Algorithm Series for Beginners, URL: <https://www.youtube.com/watch?v=byHi41L9vTM>   + Selection Sort Algorithm Explained (Full Code Included) - Python Algorithms Series for Beginners, URL: <https://www.youtube.com/watch?v=4CykZVqBuCw>   + Quick Sort Algorithm Explained (Full Code Included) - Python Algorithm Series for Beginners, URL: <https://www.youtube.com/watch?v=kFeXwkgnQ9U> |
| Responder cuestionario sobre listas y ordenamientos. | Ordenamientos básicos | * Cuestionario con preguntas de conceptos básicos de ADT List y Ordenamientos básicos *(SelectionSort, InsertionSort, ShellSort).* **IMPORTANTE,** revisar los anuncios y actividades en las secciones oficiales del curso. (ver sección 2.2) |
| Iniciar modificaciones de los ordenamientos para el Req uno del Reto No. 1. | Ordenamientos básicos | * Repositorio GIT Laboratorio No. 2. * Instrucciones para modificar los ordenamientos básicos (ver sección 2.4). |

## En el Encuentro Sincrónico:

|  |  |  |
| --- | --- | --- |
| ACTIVIDAD PEDAGÓGICA | TEMA TRATADO | MATERIAL ASOCIADO |
| Completar las modificaciones de los ordenamientos. | Ordenamientos básicos | * Repositorio GIT Laboratorio No. 2. * Instrucciones para modificar los ordenamientos básicos (ver sección 2.4). |
| Analizar el comportamiento de pruebas de ordenamiento y ADT List | Ordenamientos básicos + ADT List | * Instrucciones para integrar las implementaciones y ordenamientos (ver sección 2.5) * Instrucciones para analizar el desempeño de los ordenamientos (ver sección 2.6). |
| Crear versión unificada del código en GIT | Ordenamientos básicos + ADT List | * Enlace GitHub con código actualizado en la rama Master |

## Para la Retrospectiva:

|  |  |  |
| --- | --- | --- |
| ACTIVIDAD PEDAGÓGICA | TEMA TRATADO | MATERIAL ASOCIADO |
| Crear informe de trabajo | ADT List+ Ordenamientos básicos | * Instrucciones para compartir el producto final con los evaluadores del Lab No. 2 (ver 2.7) * Enlace GitHub con código actualizado en la rama Master para Lab No. 2. * Invitación a los evaluadores (monitores y profesor) como colaboradores de la organización o proyecto. |