* **Requerimientos de entrega del proyecto**:
* El proyecto debe entregarse como un archivo con compresión “.7z” en un enlace dentro del Campus Virtual de la asignatura. El tamaño máximo del archivo “.7z” es de 20MB, por lo que de no cumplir con esta disposición, no podrá subir el archivo al Campus Virtual. Dentro del archivo debe haber un directorio “Integrantes”, “**Documentación de Análisis**” y  “Código”, cada uno con el contenido respectivo. Los integrantes deben ir como una fotografía de la “lista de integrantes” donde se incluye el nombre de la clase, sección y fecha, junto con los nombres completos, números de cuenta y firma de cada integrante. El nombre del archivo 7z debe ser LLP-[SECCIÓN]-ProyectoFinal.7z donde debe reemplazarse [SECCIÓN] por la sección de su clase.
* En el directorio Código debe existir un proyecto que integra todos los lenguajes de programación y que resuelve la problemática planteada en clase.
* Su carpeta código debe estar compuesta por un único archivo **main.py**  y por una carpeta **Core** que contenga todos los archivos necesarios para la funcionalidad del proyecto.
* **Evidencia de funcionamiento y explicación**
* Su proyecto tiene que tener una evidencia de Video la cual deberá subirse como un archivo separado en la entrega del proyecto usando el Campus Virtual.
* Mínimo 5 minutos de duración, máximo 8 minutos en total. Si su proyecto no está acompañado de esta evidencia de video, sin importar que el código de su proyecto esté al 100% de funcionalidad y correctitud, se le asignará como máximo el 50% de la calificación del proyecto.
* Debe contener **una demostración del funcionamiento visual de su proyecto** con una duración de 3 minutos, junto a **una explicación de todos los componentes (archivos) de código** con una duración de 2 minutos.
* Más de una persona puede participar en la explicación del video mediante voz donde cada uno debe identificarse antes de hablar indicando su primer nombre (e.g.: “Mi nombre es Pedro y a continuación voy a explicar....”). **Es obligatorio** el uso de audio de explicación incluído en el video.
* El video debe estar en codificación MP4/H. 264. Puede revisar [https://ffmpeg.org/](https://www.google.com/url?q=https://ffmpeg.org/&sa=D&ust=1597480694074000&usg=AOvVaw0E_wHdXXTitrAdO-3n5dHg) para más información.
* El vídeo debe tener un máximo de 20MB de tamaño. La calidad debe ser aceptable, es decir, debe poder leerse y escucharse todo lo que se explica en el video. El equipo de estudiantes debe investigar con suficiente tiempo cómo aplicar la compresión al mismo para cumplir con este requerimiento ([https://ffmpeg.org/](https://www.google.com/url?q=https://ffmpeg.org/&sa=D&ust=1597480694074000&usg=AOvVaw0E_wHdXXTitrAdO-3n5dHg)). Al ser este tamaño una limitación del Campus Virtual, los estudiantes no podrán subir el archivo de video de no cumplir con este requerimiento de tamaño en MB.
* Antes de enviar su video **revise detenidamente el audio**. Un video con un audio “de volumen bajo” donde no se pueda distinguir la voz o la explicación, tendrá una penalización hasta del 50% inclusive cuando el video muestre correctamente lo solicitado.
* El video debe ser grabado mediante "captura de pantalla" no mediante cámara de celular.
* **Entregas.**
* **No se aceptará entrega de proyectos fuera del Campus Virtual.** No se aceptará proyectos donde el estudiante no suba un archivo “.7z” con los componentes antes mencionados, junto con el video de evidencia “.mp4” con los contenidos antes descritos. Para este último escenario se marcará como sin entrega dando una nota final de 0%.
* No se permitirán entregas tardías ya que la recepción a través del Campus Virtual tienen una fecha de entrega límite.
* **Documentación.**
* La **Documentación de Análisis** refiere a un documento Markdown que debe contener cada elemento conceptual y de lluvia de ideas que usted incluya en su programación desde el inicio de su proyecto. Debe incluir también descriptivamente todos los hallazgos de lo que está planificando hacer desde el inicio del proyecto, junto con todos los hallazgos encontrados en el camino. Estos hallazgos deben incluir pero no limitarse a: clases, métodos, funcionalidades, algoritmos, componentes, condicionales, restricciones y otros que usted ha encontrado, que son necesarios para implementar la programación necesaria que cumplirá con su proyecto y que por tanto deben quedar plasmadas para su lectura por parte del docente. La documentación, adicionalmente, deberá incluir imágenes de cualquier diagrama o diseño conceptual que usted generó para el proyecto. La documentación debe redactarse **impersonalmente**, en **tiempo** **presente atemporal**.
* Su proyecto debe poseer documentación en código, declarando la funcionalidad, autor, versión de cada clase/archivo, y la funcionalidad de cada método, algoritmo y gramática usted ha creado en el programa.

Guion para vídeo:

1. Edgar Benedetto habla presentando el proyecto final de la asignatura de lenguajes de programación en la sección 1100 con el ingeniero **José Inestroza**, Empezando por la interpretación se puede hacer **uso del siguiente comando** para la realización de los **análisis léxico y sintáctico semántico** en un archivo que contiene codigo del lenguaje de programación JavaScript, estos **últimos 2** se realizan al mismo tiempo, lo cual es **mencionado en las lecturas** de la clase como una forma recurrente y optima dado que **al mismo tiempo que se analiza** el **orden en el que están escritos** los lexemas se analiza si semánticamente está correcto, haciendo uso de lark, con este **primer comando** sólo se muestra **encabezado y lo ejecutado** por el código interpretado, con este segundo comando se **realiza la impresión de la tabla de símbolos** del codigo interpretado, **mostrando los lexemas y su token** correspondiente, dado que en el análisis léxico se hizo **uso de autómatas finitos y no de Plex**, el generador de analizadores léxicos de python dado que necesita de **una versión anterior** a python3, por lo tanto **la lectura se realiza carácter por carácter**, **el ultimo comando implica** la parte de reconocimeinto de lenguaje de programación a partir **de su gramatica, en este** caso solo están incluidos los lengujes de programación Bash y Ruby, **El archivo se lee y se utilizan ambas gramáticas** para determinar si **el código reconocido pertenece a uno** o a otro o a ninguno de los lenguajes de programación aceptados.
2. **Presentando Recognition 🡪** Encargado de reconocer los parámetros leídos de la consola junto con el contenido del archivo.
3. **Presentando el Reader 🡪** Es donde se leen las instrucciones introducidas en la consola donde una de ellas es el archivo a interpretar o reconocer y el comando a ejecutar si es que se escribió alguno.
4. **Presentando Análisis Léxico 🡪**  Aquí se encuentra la abstracción del token

luego está el autómata el cual es el encargado de la lectura carácter por carácter y por último el creador de tokens donde se realiza la subdivisión de estados, partiendo de los meta estados de formado y en formación

1. **Presentando Dictionary 🡪** Aquí encapsulamos palabras reservadas, booleanos, operadores lógicos. asignación, fin de instrucción inicio y fin de agrupación con su identificador para la creación de la tabla de símbolos.
2. **Presentando Análisis Semantico 🡪** Es donde se engloban todas las funciones declaradas en la gramatica de JavaScript para la captura de datos, en algunas se hace uso del árbol generado por lark donde del mismo se obtienen los lexemas y se almacenan en objetos temporales para luego en funciones internas decidir si se almacenan en los objetos tipo diccionario de variables, funciones o argumentos.