Manual de Metodología y Operación para el Laboratorio de Compiladores (Clave 0817)

Versión: 2.2 (Ciclo 2026-1)

16 de agosto de 2025

1. Filosofía Pedagógica: Integridad, Profundidad y Flexibilidad

La metodología de este laboratorio se fundamenta en un enfoque que busca garantizar la **integridad académica** y un **aprendizaje profundo**. Se combina un marco de trabajo estructurado (rigor) con un sistema que fomenta la iniciativa y la excelencia individual (flexibilidad). El énfasis se pone en la correcta implementación de los **algoritmos y conceptos teóricos**.

La implementación de esta filosofía se basa en tres pilares:

- El diseño de las prácticas y su evaluación se centran en el **proceso** y la **demostración de maestría**, no solo en el producto final. Se busca verificar que el estudiante comprende genuinamente los conceptos, forzándolo a razonar y defender sus decisiones técnicas en un entorno donde la IA no puede sustituir su conocimiento.
- Se aplicarán de manera consistente y transparente las reglas de entrega, los protocolos de evaluación y los criterios de calificación. El objetivo es desarrollar un sentido de responsabilidad y profesionalismo en los estudiantes.
- Se promoverá activamente un sistema de incentivos. Aunque el entorno técnico está estandarizado, se reconoce a quienes destacan y se proporcionan desafíos opcionales que inviten a la exploración.

2. Estructura y Calendarización del Proyecto Semestral

El laboratorio se organiza en un plan de 14 sesiones que guía la construcción incremental del compilador, dejando una semana de margen para imprevistos, repasos o actividades extra.

Sesiones	Unidad(es) del Temario	Objetivo Práctico de la Fase
1	I. Introducción	**Setup y Primeros Pasos:** Configuración del entorno de desarrollo Java y familiarización con el sistema de validación.
2-3	II. Análisis Léxico	**El Scanner:** Implementación de un procesador que convierte un flujo de caracteres (`stdin`) en una secuencia de tokens (`stdout`).
4-6	III. Análisis Sintáctico	**El Parser:** Implementación de un procesador que valida la estructura sintáctica y construye un Árbol de Sintaxis Abstracta (AST).
7-9	IV. Análisis Semántico	**Chequeo de Tipos y Alcance:** Implementación de un procesador que recorre un AST y realiza validaciones semánticas.

Sesiones	Unidad(es) del Temario	Objetivo Práctico de la Fase
10-12	V. Gen. de Código Intermedio	**Traducción a Código de Tres Direcciones:** Implementación de un procesador que traduce un AST a código intermedio.
13-14	-	**Integración y Presentación Final:** Integración de todas las fases y demostración final del compilador funcional.

3. Entorno Técnico y Metodología de Validación

- **Lenguaje de Programación Estándar:** Para asegurar una base homogénea y facilitar el soporte técnico, todas las prácticas del laboratorio se desarrollarán utilizando el lenguaje de programación **Java**.
- **Contrato de Interacción (Entrada/Salida):** Todas las prácticas leerán desde la **entrada estándar** (`stdin`) y escribirán los resultados en la **salida estándar** (`stdout`).
- **Validadores Automáticos:** Para cada fase se proporcionará un validador para que los estudiantes verifiquen la correctitud de su solución de forma autónoma y objetiva.

4. Metodología de Evaluación y Calificación

Nota Importante sobre Flexibilidad: *El flujo de trabajo y las ponderaciones aquí descritas representan el marco general del curso. Sin embargo, la metodología específica y el valor de cada componente evaluable (como la "Verificación de Maestría") podrán adaptarse a los objetivos de cada práctica en particular. Todos los ajustes serán comunicados claramente al inicio de cada sesión.*

4.1. Modalidad de Trabajo

- **Trabajo en Equipo:** Solamente los estudiantes que asistan al laboratorio presencialmente y cumplan con un **mínimo del 80 % de asistencia** a la sesión de 2 horas podrán trabajar en equipo en las actividades que se indiquen. Esta política podrá estar sujeta a cambios dependiendo de la naturaleza de cada práctica.

4.2. Composición de la Calificación de Prácticas

Cada práctica se evaluará sobre 10 puntos, compuestos por dos partes fundamentales:

- **Entrega Funcional (80 % - 8 puntos):** Corresponde a la entrega del **código fuente en Java** y un **mini-reporte técnico** en formato `Markdown`. Este reporte debe servir como una guía paso a paso de la implementación, documentando las decisiones de diseño y el proceso seguido. El objetivo es que sirva como material de consulta futuro, hecho por y para ustedes.

- **Verificación de Maestría (20 % - 2 puntos):** Corresponde a la demostración, por defecto presencial, de que el estudiante es el autor intelectual de la solución. Para casos especiales, y con previa justificación aprobada por el profesor, esta verificación podrá realizarse en línea. Este componente es **obligatorio**; sin su aprobación, la práctica se considera no entregada.

4.3. Flujo de Trabajo Estándar (Modalidad Presencial)

1. **Lanzamiento y Desarrollo (110 min):** Se presenta el problema y el **Mínimo Viable en Clase (MVC)**.
2. **Verificación de Maestría en Clase (10 min):** Al final de la sesión, el equipo demuestra que cumple con el MVC. Al hacerlo, obtiene los puntos de este componente.
3. **Entrega Funcional:** Se sube el código final y el mini-reporte al repositorio para su calificación.

4.4. Protocolos para Entregas Extemporáneas y Casos Especiales

4.4.1. A. Incumplimiento del MVC en Clase

- **Consecuencia:** El equipo recibe 0 en el componente de "Verificación de Maestría" (pierde 2/10 puntos, o el valor asignado para esa práctica).
- **Camino de Recuperación (Opcional):** Para recuperar los puntos, el equipo deberá resolver y entregar obligatoriamente una de las actividades del **Sistema de Incentivos** (ver sección 4.5), excluyendo aquellas que por su naturaleza solo pueden realizarse durante la sesión presencial (ej. "Excelencia en Clase", "Apoyo a la Comunidad").

4.4.2. B. Entrega Extemporánea

1. **Entrega del Código (Asíncrona):**
   - **Penalización por Retraso:** Se aplicará una penalización de **0.5 puntos** sobre la calificación final por cada 24 horas de retraso.
   - **Fecha Límite Absoluta:** No se aceptarán entregas después de **7 días naturales**. La razón es que la solución de la práctica se publicará aproximadamente en esa fecha. Además, para el equipo docente, es necesario tener un punto de corte claro para poder calificar y dar retroalimentación de manera ordenada; un flujo constante de entregas tardías haría muy difícil gestionar la evaluación y devolver las notas a tiempo.
2. **Verificación de Maestría (Síncrona y Obligatoria):**
   - El estudiante debe agendar su verificación, por defecto presencial, en la siguiente sesión de laboratorio a la que asista.
   - Para casos especiales, y con previa justificación aprobada por el profesor, esta verificación podrá realizarse en línea si el estudiante no puede asistir a la sesión.
   - Esta verificación es un **requisito indispensable** para que la práctica sea calificada. Sin ella, la calificación será 0.

4.5. Sistema de Incentivos (Puntos Extra)

Se otorgan puntos extra sobre la calificación final de cada práctica por:

- **Excelencia en Clase (+0.5)**
- **Documentación Profesional (+0.5)**
- **Desafío Técnico Opcional (+1.0 a +1.5)**
- **Apoyo a la Comunidad (+0.5)**
- **Profundización Algorítmica (+1.5 a +2.0)**

Nota sobre Incentivos: *La disponibilidad y el valor en puntos de cada incentivo podrán ser ajustados para cada práctica en particular, con el fin de alinear las recompensas con los objetivos de aprendizaje de la sesión. Incluso, se podrán agregar o eliminar incentivos según se considere pertinente. Cualquier cambio será comunicado con antelación.*