



Universidad Nacional de San Luis

Práctico de Máquina

San Luis - 2023

Asignatura:

Diseño y Construcción de Compiladores

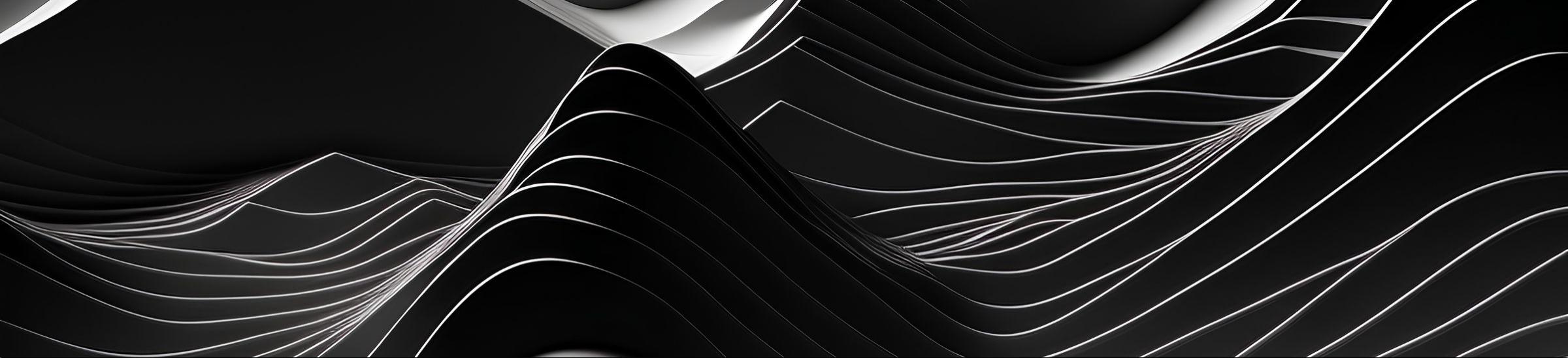
Estudiante:

Kwist Daniel

Docentes:

Aragón Victoria Soledad

Ferreti Edgardo



Temas a desarrollar

- Introducción
- Desarrollo
 - Entrega 1
 - Entrega 2
- Conclusiones

Introducción

Se hará un recorrido sobre las decisiones tomadas en el diseño, desarrollo e implementación del práctico de maquina, y así como las experiencias y dificultades encontradas.

Se busca implementar un:

- Análisis sintáctico: mediante la implementación de una estrategia de recuperación de errores antipánico.
- Análisis semántico: de un sistema de tipos dado mediante el manejo de una tabla de símbolos.



Desarrollo

Se analizara el recorrido desde el primer contacto con el trabajo práctico hasta el desarrollo en ambas entregas, especificando las decisiones de diseño e información pertinente.

Etapas:

Entrega 1:
Análisis Sintáctico

Implementación de la estrategia de recuperación de errores antipánico.



Análisis de archivos y gramática

Nos encontramos con:

Gramática

Presentada en formato bnf y bnf-e de un subconjunto del lenguaje C++.

Parser

Se ofrece un Parser Descendiente Recursivo (TopDown) ya construido en C.

Archivo Makefile

Útil para integrar todos los archivos y ficheros utilizados para ejecutar el compilador.

Extras

- Analizador Lexicográfico
- Códigos de errores, constantes y funcionalidades complementarias.
- Compilador UCC.

Decisión sobre el debugging

- Implementado mediante impresiones de pantalla (printf).
- Ventaja de facilidad y rapidez de uso.
- Desconocimiento de alternativas.
- Encapsulamiento hasta el error.



Entrega 1

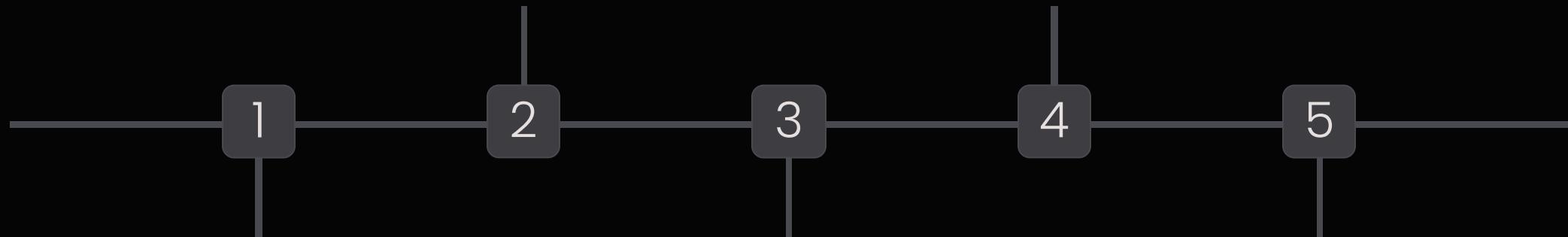
- Ejercicio 11 - Unidad Temática 2
- Decisiones de diseño.
- Lotes de prueba.

Análisis de Tests

Realizar el análisis para incorporar los tests iniciales y finales según corresponda

Decisiones de Diseño

Modificaciones elegidas por el alumno



Conjunto first

Armar los conjuntos first de cada no terminal

Parámetros

Añadir los parámetros folset a cada procedimiento e incorporar los argumentos de tests y llamadas a procedimientos

Lotes de prueba

Confección de los lotes de prueba

1. En lista_declaraciones_param: se agrega en el while el first de declaración parámetro para forzar la entrada
2. En declaracion_parametro: se agrega el CCOR_CIE por si se olvida al usuario la inclusión del mismo.
3. En lista_declaraciones_init: se agrega el first de declarador_init al while para forzar la entrada al while.
4. En declaracion_variable: se agrega el first de lista_declarac_init en el if para forzar la entrada a la sentencia condicional por si el usuario se olvida la coma.
5. En declarador_init: se fuerzan entradas al switch con flotante y carácter para la rama de asignación o con], { y } para la otra rama
6. En lista_inicializadores: se agrega el first de constante al while para forzar la entrada por si el usuario se olvida la coma.
7. En proposición: se agregan para forzar la entrada en entrada y salida el CSHL y CSHR.
8. En proposicion_e_s: se fuerzan respectivamente los CSHL y CSHR donde corresponden por el ítem (7).
9. En expresion_simple: se fuerza en el while con el first de termino por si el usuario se olvida un operando.
10. En termino: se fuerza en el while con el first de factor por si el usuario se olvida un operando.
11. En lista_expresiones: se fuerza el while con el first de expresion por si el usuario se olvida la coma.



Entrega 2

- Tarea 9 - Unidad Temática 4.
- Decisiones de diseño.
- Códigos de errores propios.
- Lotes de prueba.

Semántica

Se realiza el análisis de la guía semántica, identificando los controles semánticos.

Decisiones de Diseño

Modificaciones elegidas por el alumno

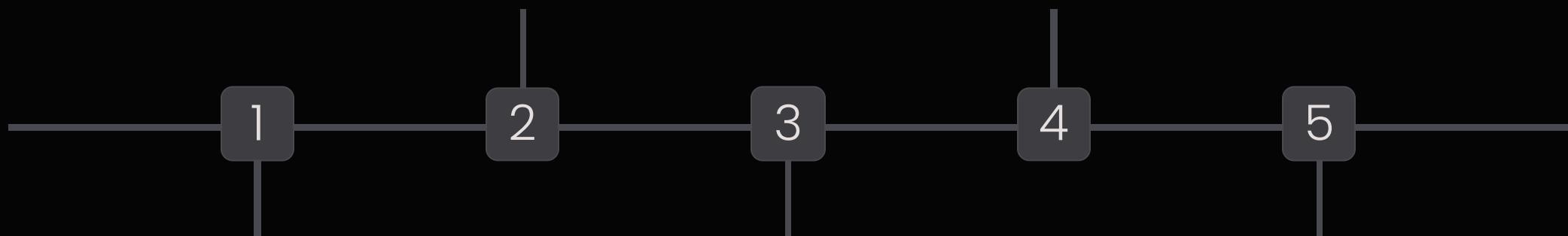


Tabla de simbolos

Se realiza el análisis sobre el funcionamiento y estructura de la tabla de símbolos.

Controles Semánticos

Se seleccionan e implementan los controles semánticos

Lotes de prueba

Confección de los lotes de prueba





Controles semánticos implementados

- Variables:
 - 4. Las variables deben ser declaradas antes de su uso.
 - 5. Las variables simples no pueden ser declaradas con tipo void.
 - 6. Las variables tienen alcance de bloque.
- Arreglos:
 - 7. El tipo base de los arreglos puede ser: char, int o float.
 - 8. La cantidad de elementos de un arreglo puede estar dada por un número natural (es decir, mayor a 0) y/o a través de la inicialización del mismo.
- Asignación:
 - 10. Los tipos de ambos lados de la asignación deben ser estructuralmente equivalentes.
 - 11. No se permite la asignación de arreglos como un todo.
 - 13. En la inicialización de arreglos en el momento de la declaración se debe chequear que la cantidad de valores sea igual o menor a la constante y que el tipo también se corresponda. El ítem 11 es realizado de manera indirecta por decisiones de implementación que para utilizar una variable definida como arreglo en una proposición se requiere un índice (excepto cuando se utiliza como parámetro en la invocación de una función).

- Coerciones:
 - Para la asignación.
- Entrada/Salida:
 - 19. Las proposiciones de E/S aceptan variables y/o expresiones de tipo char, int y float.
- Funciones/Parámetros:
 - 23. El reconocimiento de los parámetros se realiza por posición.
 - 26. La cantidad de parámetros reales debe coincidir con la cantidad de parámetros formales.
 - 27. El nombre de un arreglo en el parámetro real implica proveer la dirección del arreglo. En este caso el parámetro formal debe ser definido como pasaje por valor.
 - 28. No se permite & [] en la definición de un parámetro.
 - 31. main() debe ser un procedimiento sin parámetros que tiene que aparecer exactamente una vez en el programa fuente.
- Lógica Booleana:
 - 33. Las condiciones de las proposiciones de selección e iteración pueden ser de tipo char, int y float.

A large, abstract graphic on the left side of the slide features a series of white, wavy, ribbon-like lines against a dark gray background. These lines create a sense of depth and motion, resembling waves or folds in a fabric. They are layered and curve across the frame, with some lines being more prominent than others.

Decisiones de Diseño e Implementación

- Guardar la posición de los tipos en variables globales.
- Tipos auxiliares.
- Seguimiento de tipos.
- Flag para bloques.
- Bifurcación en factor según la clase de un identificador.

Códigos de errores agregados

- 101: La cantidad de valores inicializadores no puede ser 0.
- 102: La función main() ya se encuentra declarada.
- 103: El índice de un arreglo debe ser una constante entera.
- 104: Tipo de la asignación no valido.
- 105: Los tipos de ambos lados de los operadores lógicos o aritméticos deben ser estructuralmente equivalentes.
- 106: Los operandos de los operadores lógicos o aritméticos solo pueden ser de tipo char, int o float.



Conclusiones

En resumen, el compilador es funcional para los aspectos implementados, y su funcionamiento depende de las decisiones tomadas durante la construcción.

Hubo distintas situaciones que han aminorado el desarrollo, como la falta de maduración del contenido y la falta de práctica en el lenguaje.

Fue gratificante integrar los contenidos de la asignatura en un trabajo que facilita la aplicación de los temas estudiados y mejora la comprensión. Destacando el apoyo del cuerpo docente ante las dificultades encontradas durante el desarrollo.

El trabajo continua con la implementación de la generación de código intermedio.



¿Consultas?

¡Fin!

