

Universidad Mariano Gálvez De Guatemala
Sede De Boca Del Monte, Villa Canales
Ingeniería en Sistemas de Información
Ing. Ezequiel Urizar Araujo
Compiladores I - Sección B

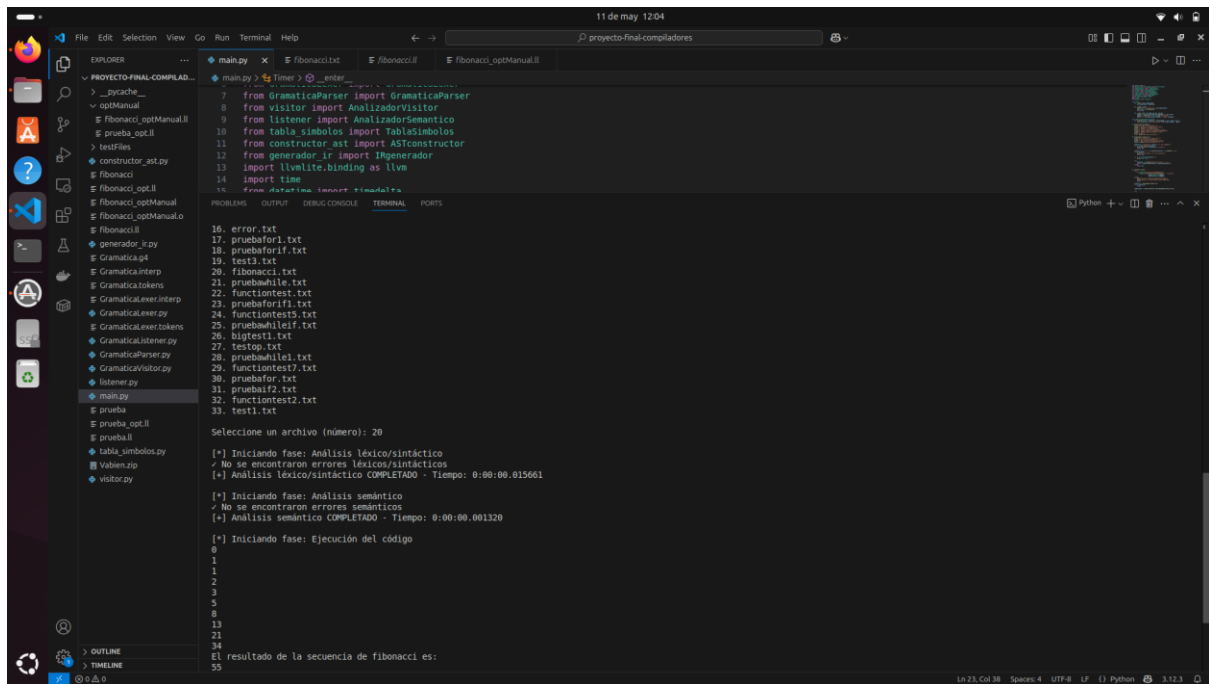


Practica

Nombre:	Carnet:
Bryan Manuel Pineda Orozco	7690-16-8869
Jonathan Joel Chán Cuellar	7690-22-1805
Roberto Antonio Ramirez Gómez	7690-22-12700
Jean Klaus Castañeda Santos	7690-22-892
Edwin Rolando Ixcoy Tot	7690-16-1582

Guatemala, 13 de Mayo del 2025

Fibonacci recursivo ingenuo



```
11 de may 12:04
proyecto-final-compiladores

main.py x fibonacci.txt fibonacci.ll fibonacci_optManual.ll
7 from GramaticaParser import GramaticaParser
8 from visitor import AnalizadorVisitor
9 from listener import AnalizadorSemantico
10 from tabla_simbolos import TablaSimbolos
11 from constructor_ast import ASTconstructor
12 from generador_ir import IRgenerador
13 import llwite.binding as llw
14 import time
15 from datetime import timedelta

16 error.txt
17 pruebafor1.txt
18 pruebaforif.txt
19 test3.txt
20 fibonacci.txt
21 pruebawhile.txt
22 functiontest.txt
23 pruebaforifl.txt
24 functiontest5.txt
25 pruebawhileif.txt
26 bigtest1.txt
27 testop.txt
28 pruebawhilel.txt
29 functiontest7.txt
30 pruebafor.txt
31 pruebaif2.txt
32 functiontest2.txt
33 test1.txt

Seleccione un archivo (número): 20

[*] Iniciando fase: Análisis léxico/sintáctico
✓ No se encontraron errores léxicos/sintácticos
[*] Análisis léxico/sintáctico COMPLETADO - Tiempo: 0:00:00.015661

[*] Iniciando fase: Análisis semántico
✓ No se encontraron errores semánticos
[*] Análisis semántico COMPLETADO - Tiempo: 0:00:00.001320

[*] Iniciando fase: Ejecución del código
0
1
1
2
3
5
8
13
21
34
El resultado de la secuencia de fibonacci es:
55
```

```
=== Archivos optimizados disponibles ===
1. fibonacci_optManual.ll
2. prueba_opt.ll

Seleccione un archivo (número): 1

[*] Iniciando fase: Compilación desde IR optimizado

Compilando a binario...

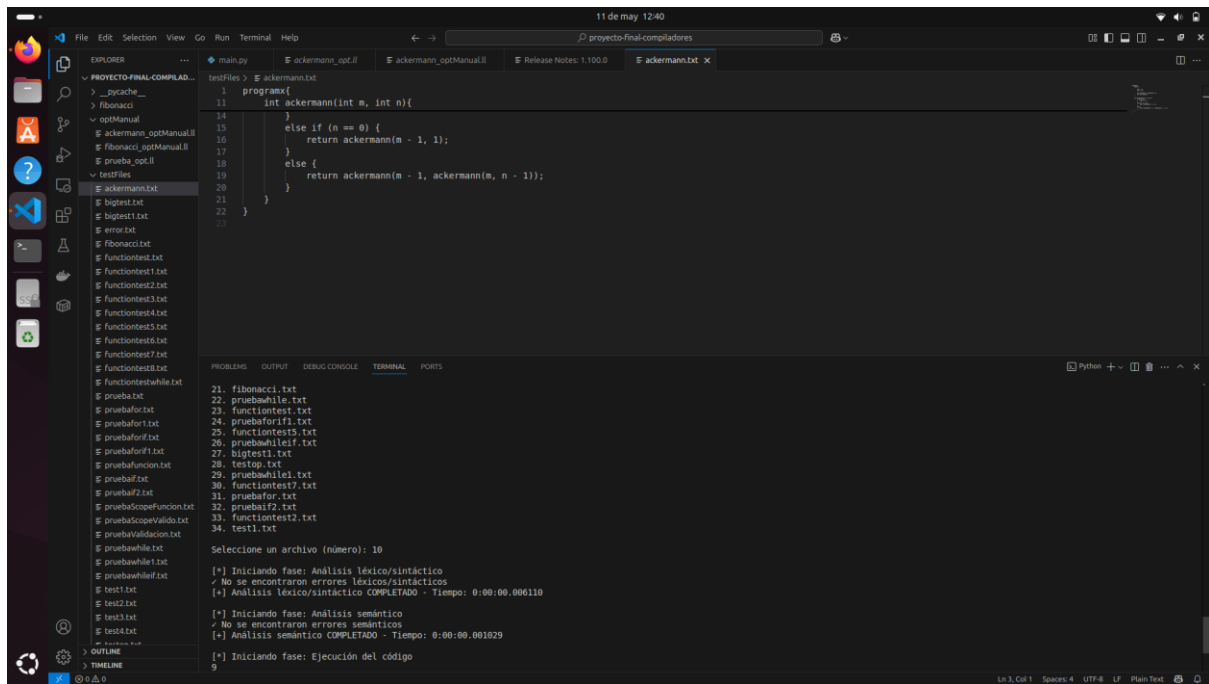
✓ Ejecutable generado: ./fibonacci_optManual

¿Desea ejecutar el programa? (s/n): s

[*] Iniciando fase: Ejecución del programa

=== Salida del programa ===
0
1
1
2
3
5
8
13
21
34
El resultado de la secuencia de fibonacci es:
55
```

Función de Ackermann



1. Compilar con optimización (--opt)
2. Compilar sin optimización
3. Generar solo código intermedio (.ll)
4. Compilar desde .ll optimizado manualmente
5. Generar ejecutable Windows (.exe)
6. Salir

Seleccione una opción: 4

=== Archivos optimizados disponibles ===

```
1. ackermann_optManual.ll
2. fibonacci_optManual.ll
3. prueba_opt.ll
```

```
Seleccione un archivo (número): 1
```

```
[*] Iniciando fase: Compilación desde IR optimizado
```

Compilando a binario...

```
✓ Ejecutable generado: ./ackermann optManual
```

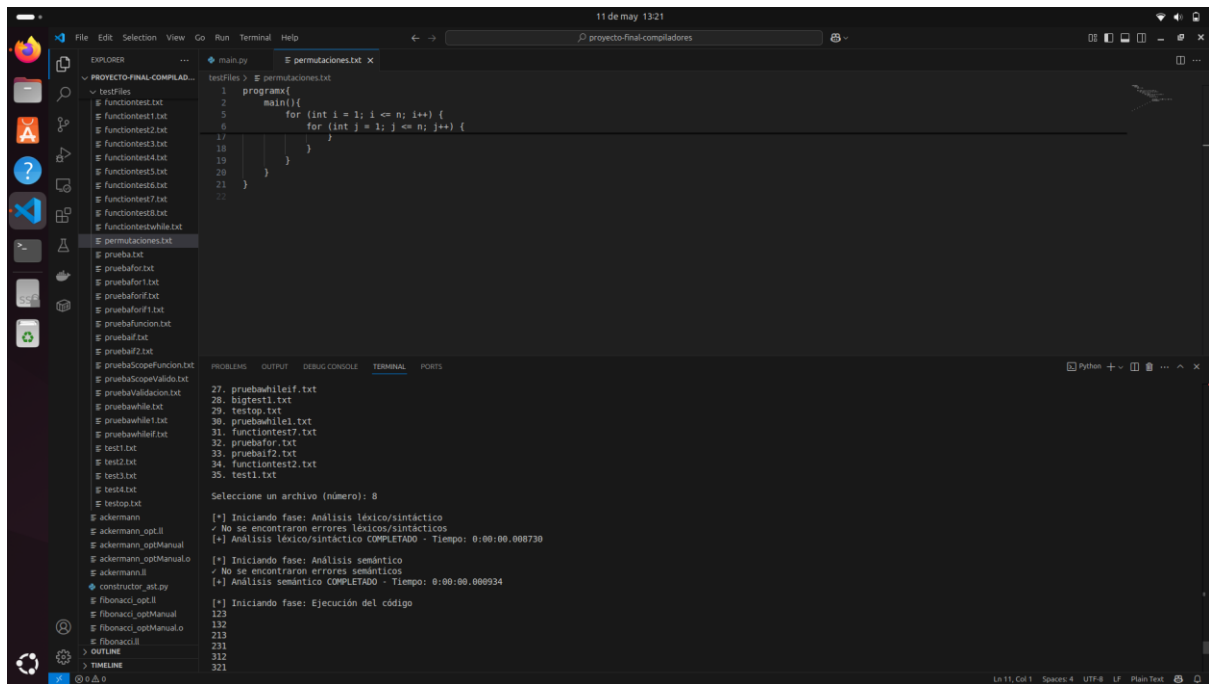
```
¿Desea ejecutar el programa? (s/n): s
```

```
[*] Iniciando fase: Ejecución del programa
```

```
=== Salida del programa ===
```

Resultado: 9

Permutaciones en fuerza bruta



```
1 program()
2 {
3     main()
4 }
5
6 for (int i = 1; i <= n; i++) {
7     for (int j = 1; j <= n; j++) {
8         // ...
9     }
10 }
11
12 // ...
13
14 // ...
15
16 // ...
17
18 // ...
19
20 // ...
21
22 // ...
23
24 // ...
25
```

27. prueba1.txt
28. prueba2.txt
29. prueba3.txt
30. prueba4.txt
31. prueba5.txt
32. prueba6.txt
33. prueba7.txt
34. prueba8.txt
35. prueba9.txt

Seleccione un archivo (número): 8

[*] Iniciando fase: Análisis léxico/sintáctico
✓ No se encontraron errores léxicos/sintácticos
[*] Análisis léxico/sintáctico COMPLETADO - Tiempo: 0:00:00.008730

[*] Iniciando fase: Análisis semántico
✓ No se encontraron errores semánticos
[*] Análisis semántico COMPLETADO - Tiempo: 0:00:00.000934

[*] Iniciando fase: Ejecución del código

123
132
213
231
312
321

Seleccione una opción: 4

=== Archivos optimizados disponibles ===

1. ackermann_optManual.ll
2. fibonacci_optManual.ll
3. prueba_opt.ll
4. permutaciones_optManual.ll

Seleccione un archivo (número): 4

[*] Iniciando fase: Compilación desde IR optimizado

Compilando a binario...

✓ Ejecutable generado: ./permutaciones_optManual

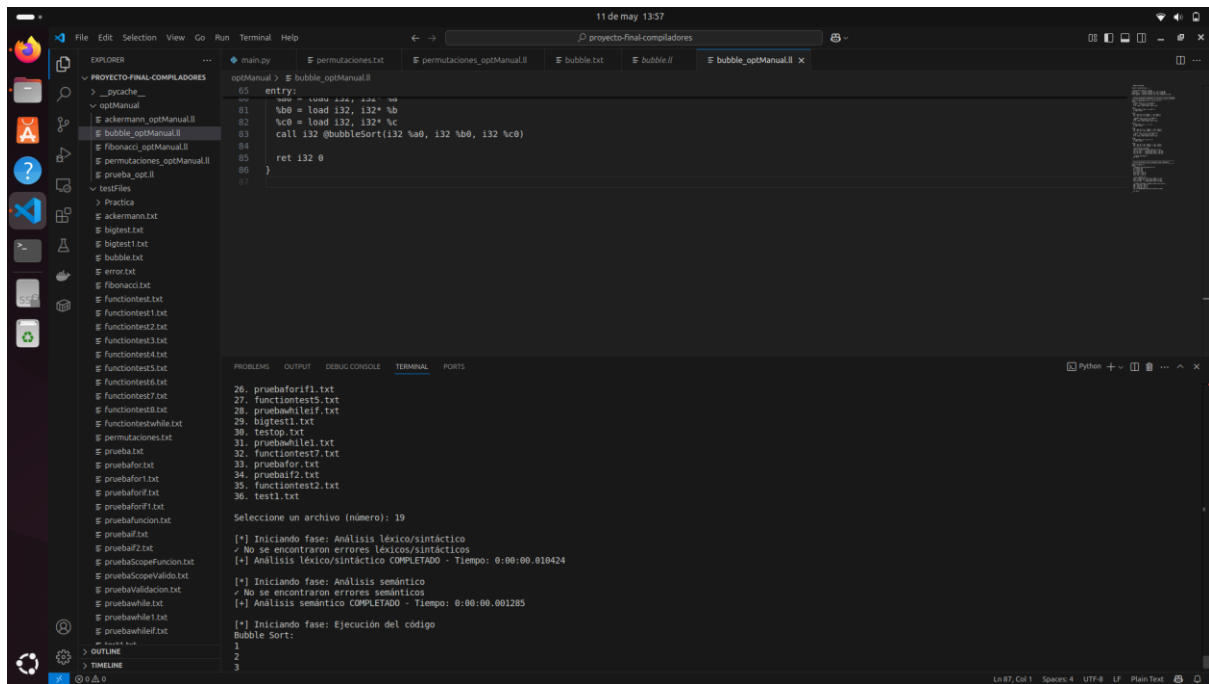
¿Desea ejecutar el programa? (s/n): s

[*] Iniciando fase: Ejecución del programa

=== Salida del programa ===

123
132
213
231
312
321

Bubble Sort



```
11 de may 13:57
proyecto-final-compiladores
main.py permutaciones.txt permutaciones_optManual.ll bubble.txt bubble.ll bubble_optManual.ll
bubble_optManual.ll
80: entry:
81:     %bb = load i32, i32* %b
82:     %cd = load i32, i32* %c
83:     call i32 @bubbleSort(i32 %a0, i32 %b0, i32 %c0)
84:
85:     ret i32 0
86:
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
26. pruebaforif1.txt
27. funciontest5.txt
28. pruebawhileif.txt
29. bigtest1.txt
30. testop.txt
31. pruebawhile1.txt
32. funciontest7.txt
33. pruebafor.txt
34. pruebaif2.txt
35. funciontest2.txt
36. test1.txt
Seleccione un archivo (número): 19
[*] Iniciando fase: Análisis léxico/sintáctico
✓ No se encontraron errores léxicos/sintácticos
[*] Análisis léxico/sintáctico COMPLETADO - Tiempo: 0:00:00.010424
[*] Iniciando fase: Análisis semántico
✓ No se encontraron errores semánticos
[*] Análisis semántico COMPLETADO - Tiempo: 0:00:00.001285
[*] Iniciando fase: Ejecución del código
Bubble Sort:
1
2
3
```

6. Salir

Seleccione una opción: 4

=== Archivos optimizados disponibles ===

1. ackermann_optManual.ll
2. fibonacci_optManual.ll
3. bubble_optManual.ll
4. prueba_opt.ll
5. permutaciones_optManual.ll

Seleccione un archivo (número): 3

[*] Iniciando fase: Compilación desde IR optimizado

Compilando a binario...

✓ Ejecutable generado: ./bubble_optManual

¿Desea ejecutar el programa? (s/n): S

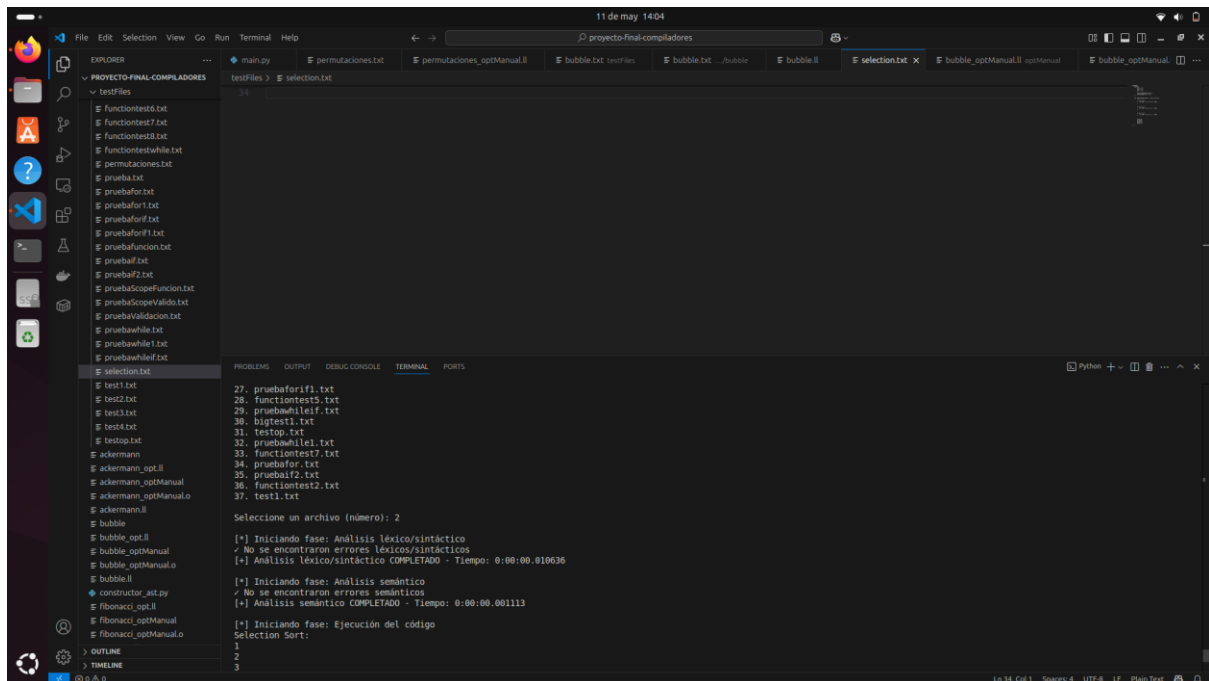
[*] Iniciando fase: Ejecución del programa

=== Salida del programa ===

Bubble Sort:

- 1
- 2
- 3

Selection Sort



Selecione una opción: 4

=== Archivos optimizados disponibles ===

1. selection_optManual.ll
2. ackermann_optManual.ll
3. fibonacci_optManual.ll
4. bubble_optManual.ll
5. prueba_opt.ll
6. permutaciones_optManual.ll

Selecione un archivo (número): 1

[*] Iniciando fase: Compilación desde IR optimizado

Compilando a binario...

✓ Ejecutable generado: ./selection_optManual

¿Desea ejecutar el programa? (s/n): s

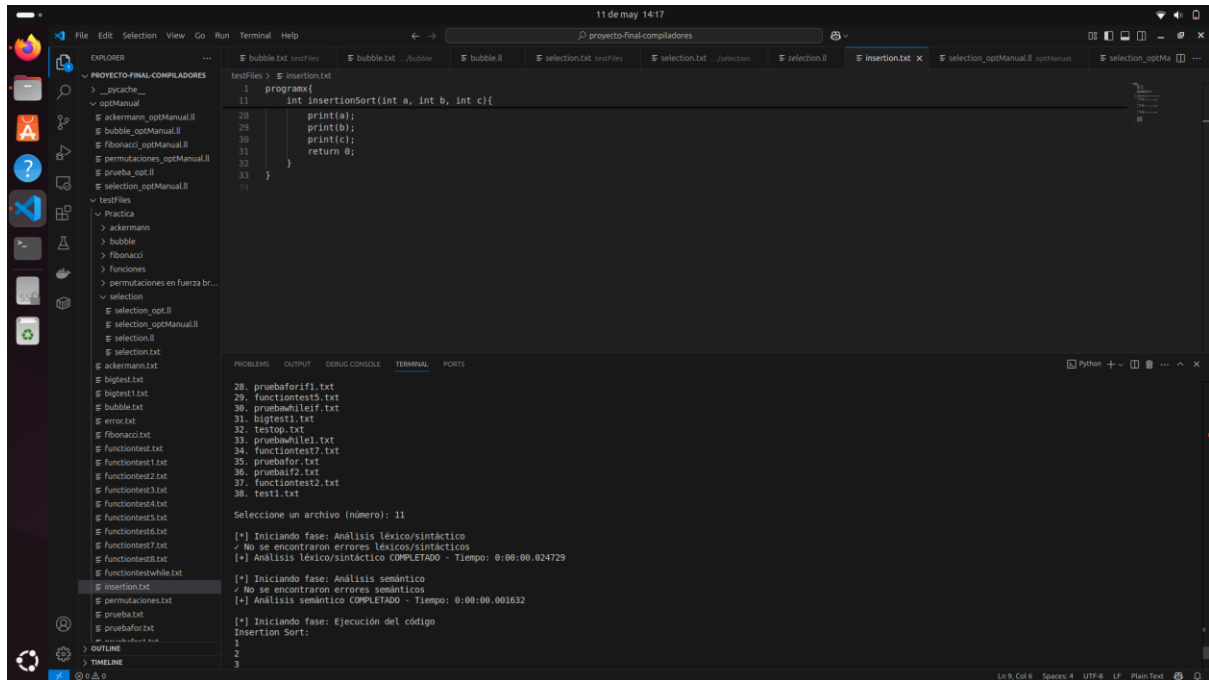
[*] Iniciando fase: Ejecución del programa

=== Salida del programa ===

Selection Sort:

- 1
- 2
- 3

Insertion Sort



```
testFile > E insertion.txt
1  program{
11     int insertionSort(int a, int b, int c){
28         print(a);
29         print(b);
30         print(c);
31         return 0;
32     }
33 }
34
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

28. pruebaforif1.txt
29. funciontest9.txt
30. pruebawhileif.txt
31. bigtest1.txt
32. testop.txt
33. pruebawhile1.txt
34. funciontest7.txt
35. pruebafor.txt
36. pruebaif2.txt
37. funciontest2.txt
38. test1.txt

Seleccione un archivo (número): 11

[*] Iniciando fase: Análisis léxico/sintáctico
✓ No se encontraron errores léxicos/sintácticos
[*] Análisis léxico/sintáctico COMPLETADO - Tiempo: 0:00:00.024729

[*] Iniciando fase: Análisis semántico
✓ No se encontraron errores semánticos
[*] Análisis semántico COMPLETADO - Tiempo: 0:00:00.001632

[*] Iniciando fase: Ejecución del código
Insertion Sort:
1
2
3

=== Archivos optimizados disponibles ===

1. selection_optManual.ll
2. ackermann_optManual.ll
3. fibonacci_optManual.ll
4. bubble_optManual.ll
5. prueba_opt.ll
6. insertion_optManual.ll
7. permutaciones_optManual.ll

Seleccione un archivo (número): 6

[*] Iniciando fase: Compilación desde IR optimizado

Compilando a binario...

✓ Ejecutable generado: ./insertion_optManual

¿Desea ejecutar el programa? (s/n): s

[*] Iniciando fase: Ejecución del programa

=== Salida del programa ===

Insertion Sort:

- 1
- 2
- 3