Programming Languages and Compilers

Sumeet Agarwal
Department of Electrical Engineering
IIT Delhi

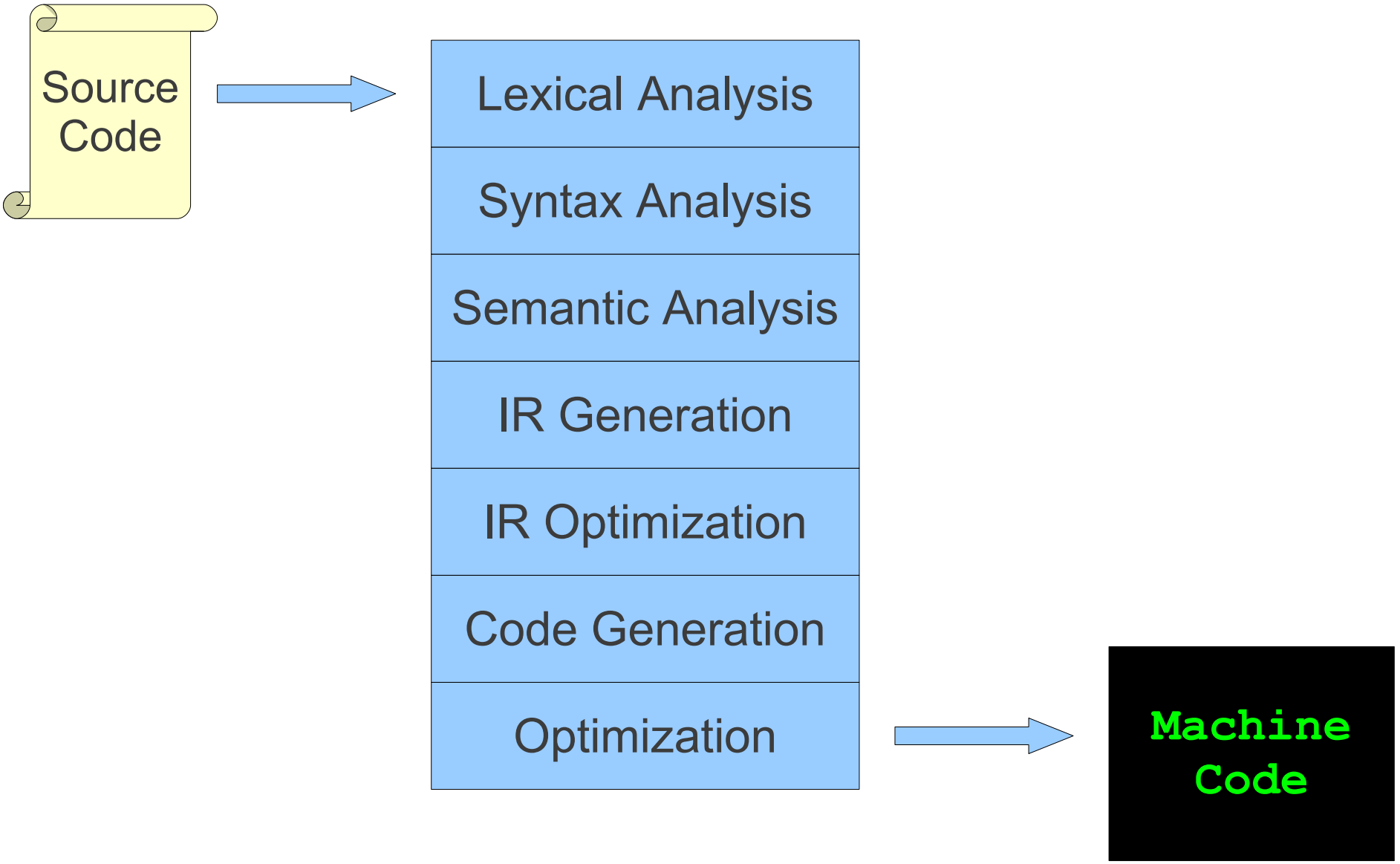
Primary Reference: Aho, Sethi, and Ullman. *Compilers: Principles, Techniques, and Tools*.

Introductory slides taken from Stanford's CS143 Compilers course: *http://www.stanford.edu/class/cs143/*.

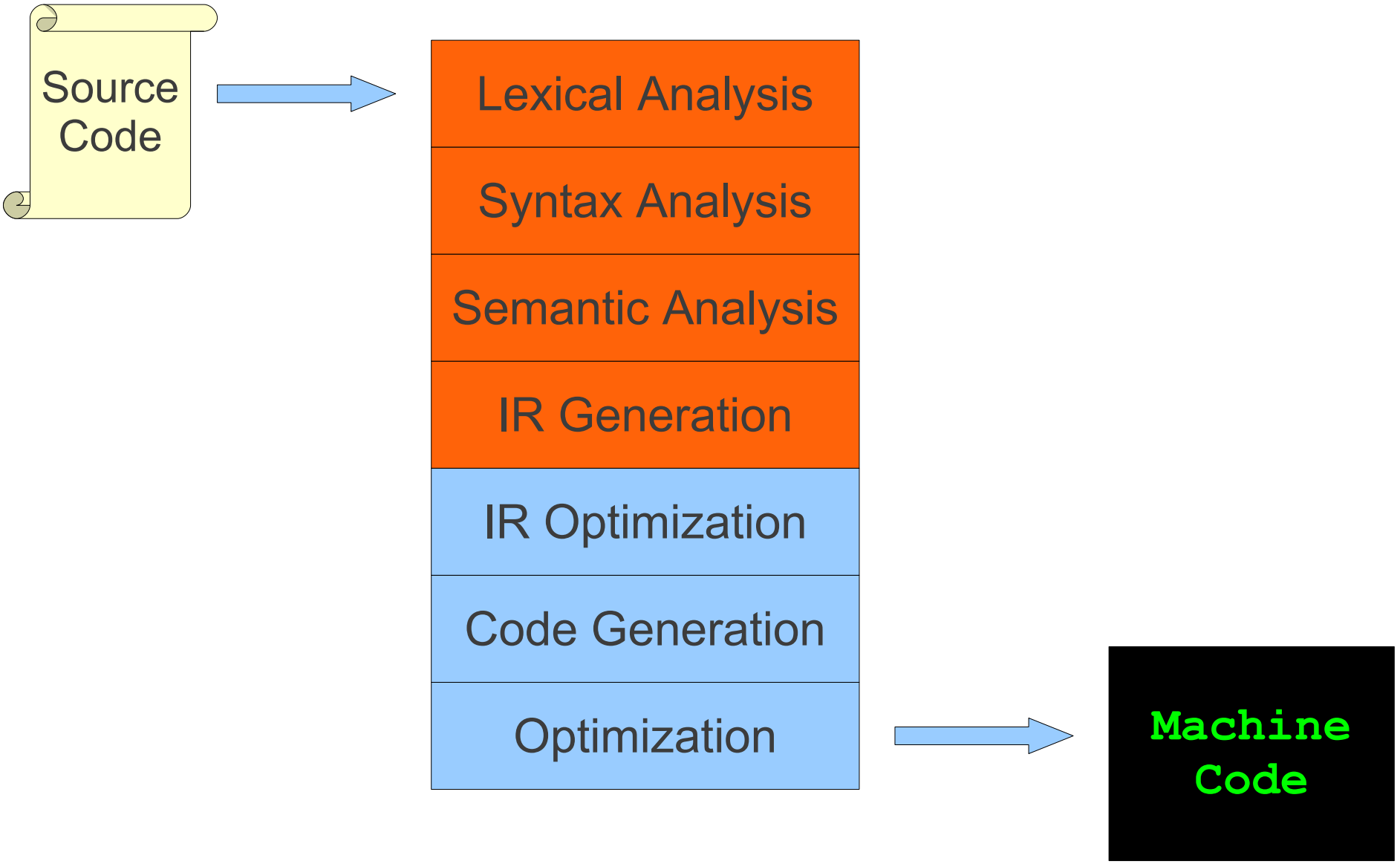
From Description to Implementation

- **Lexical analysis (Scanning):** Identify logical pieces of the description.
- **Syntax analysis (Parsing):** Identify how those pieces relate to each other.
- **Semantic analysis:** Identify the meaning of the overall structure.
- **IR Generation:** Design one possible structure.
- **IR Optimization:** Simplify the intended structure.
- **Generation:** Fabricate the structure.
- **Optimization:** Improve the resulting structure.

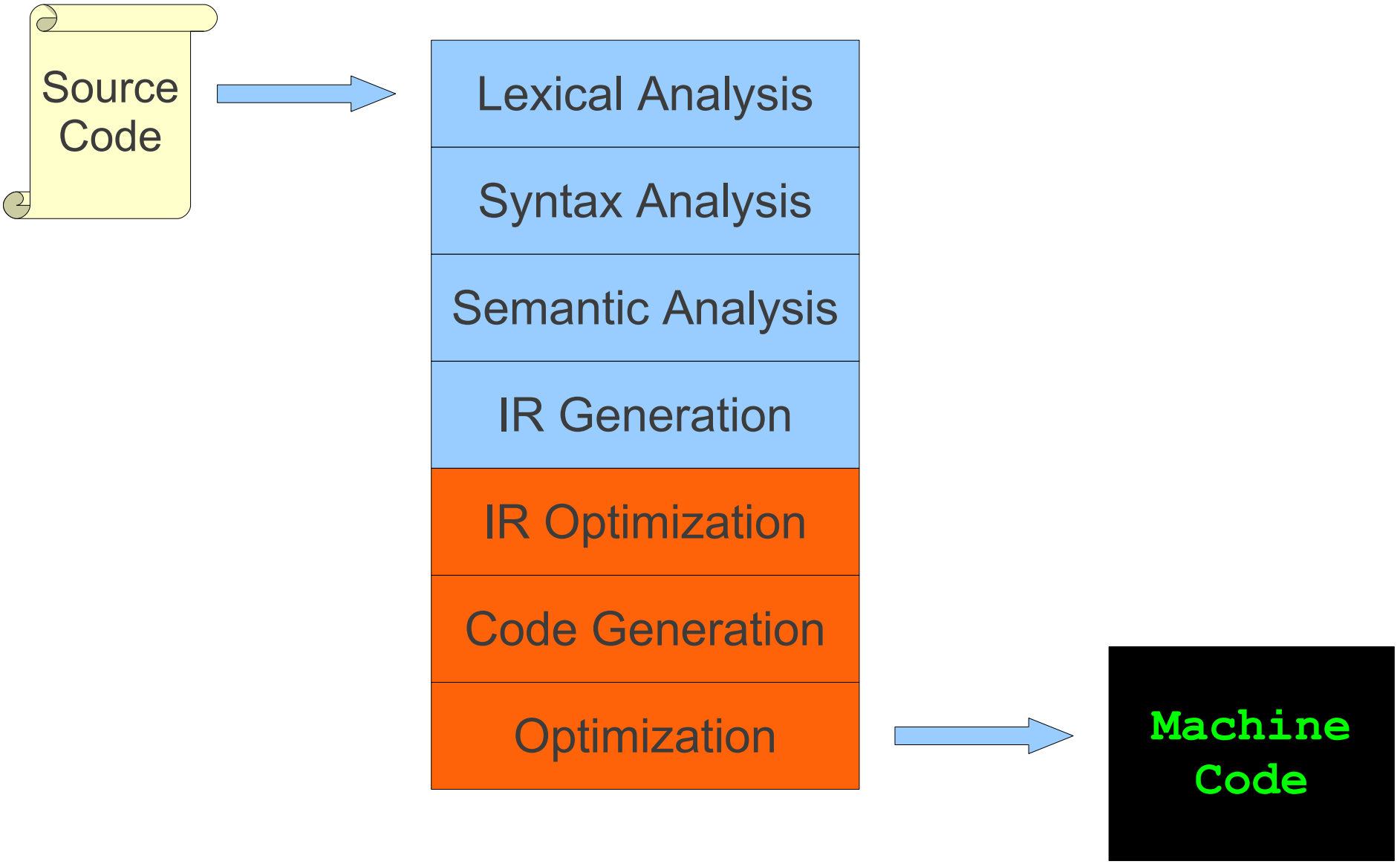
The Structure of a Modern Compiler



The Structure of a Modern Compiler



The Structure of a Modern Compiler

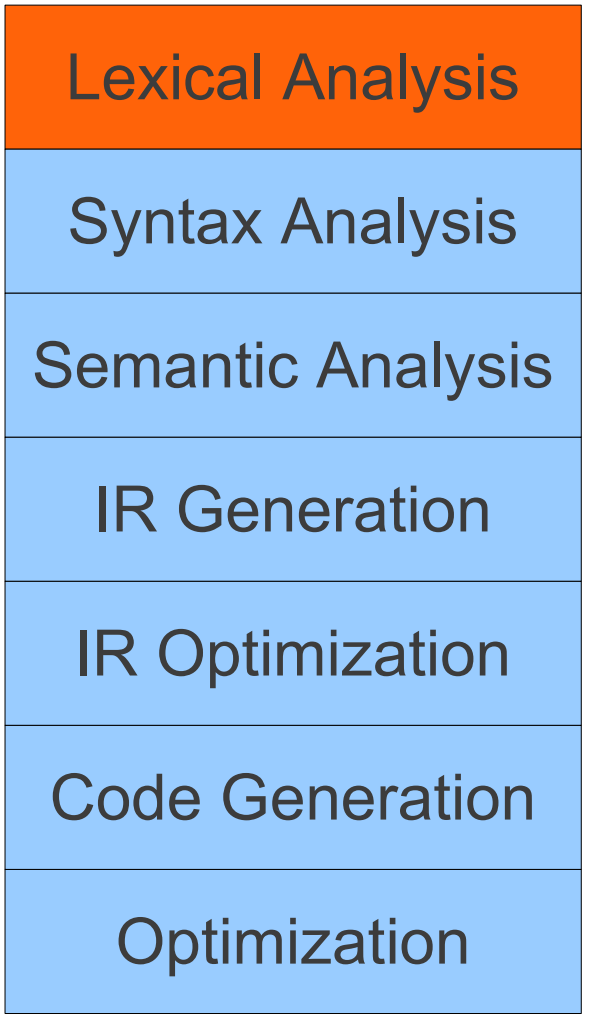


```
while (y < z) {
    int x = a + b;
    y += x;
}
```

```
while (y < z) {
    int x = a + b;
    y += x;
}
```

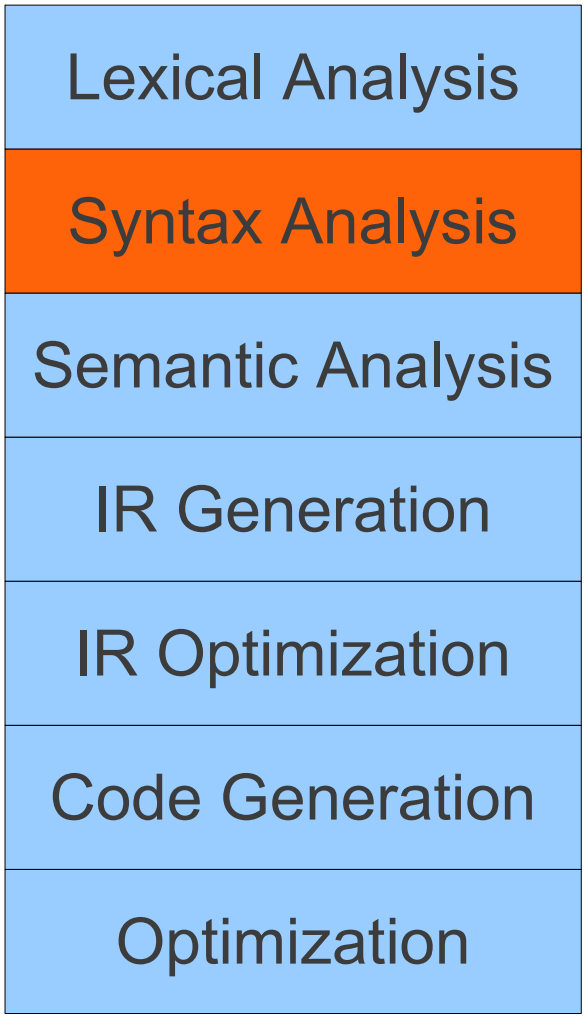
```
while (y < z) {
    int x = a + b;
    y += x;
}
```

```
T_While
T_LeftParen
T_Identifier y
T_Less
T_Identifier z
T_RightParen
T_OpenBrace
T_Int
T_Identifier x
T_Assign
T_Identifier a
T_Plus
T_Identifier b
T_Semicolon
T_Identifier y
T_PlusAssign
T_Identifier x
T_Semicolon
T_CloseBrace
```

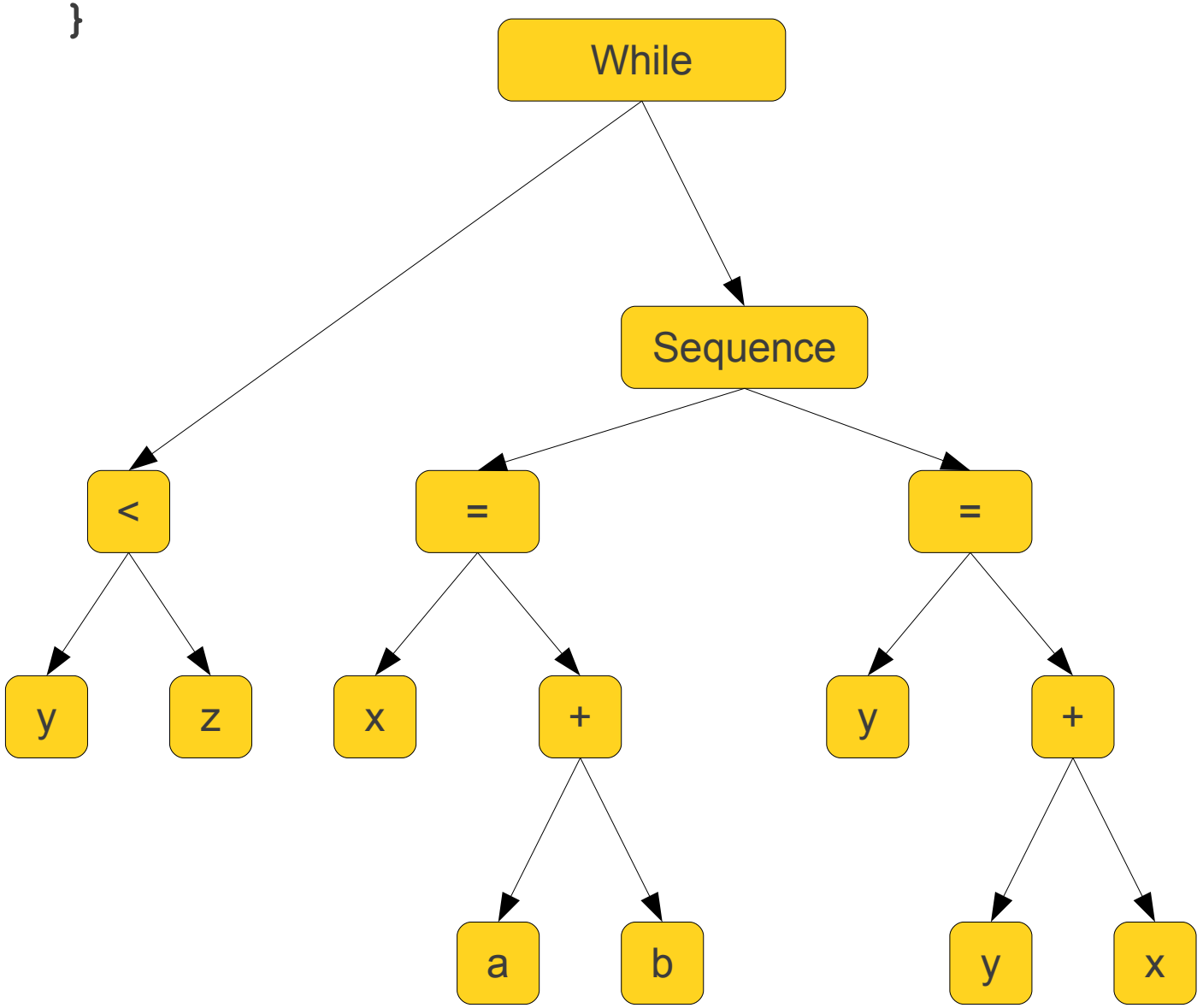


```
while (y < z) {
    int x = a + b;
    y += x;
}
```

```
T_While
T_LeftParen
T_Identifier y
T_Less
T_Identifier z
T_RightParen
T_OpenBrace
T_Int
T_Identifier x
T_Assign
T_Identifier a
T_Plus
T_Identifier b
T_Semicolon
T_Identifier y
T_PlusAssign
T_Identifier x
T_Semicolon
T_CloseBrace
```

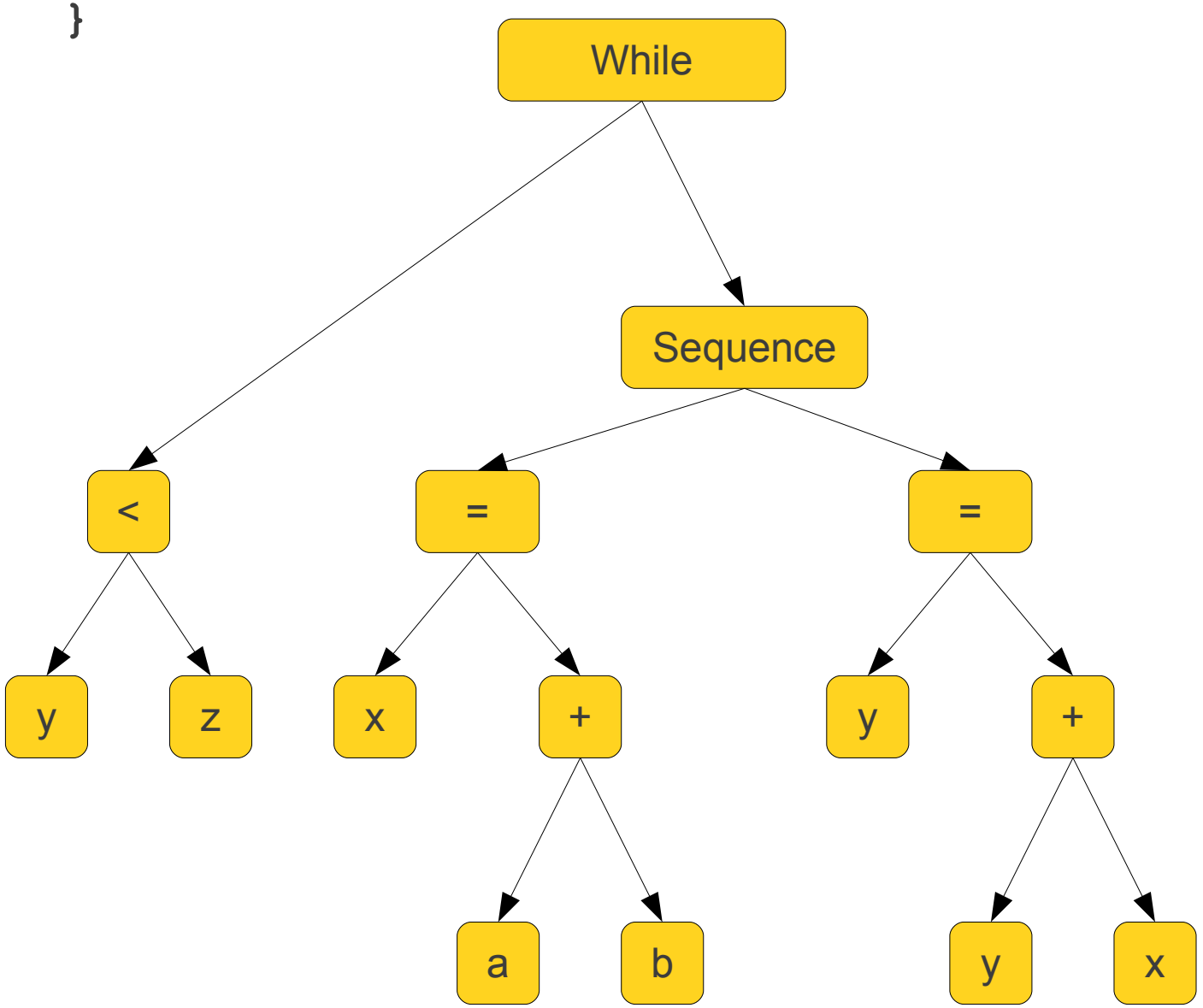


```
while (y < z) {
    int x = a + b;
    y += x;
}
```



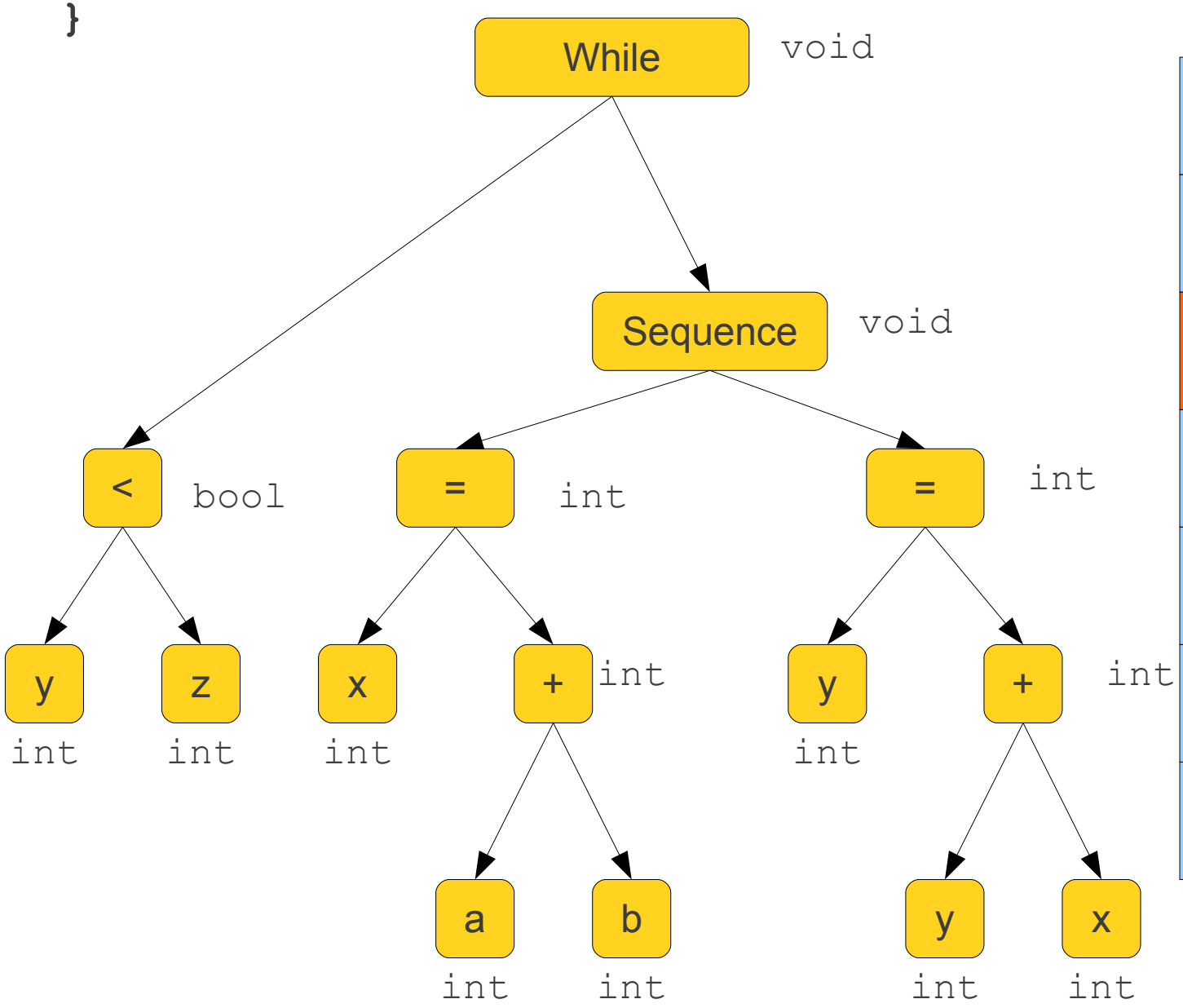
- Lexical Analysis
- **Syntax Analysis**
- Semantic Analysis
- IR Generation
- IR Optimization
- Code Generation
- Optimization

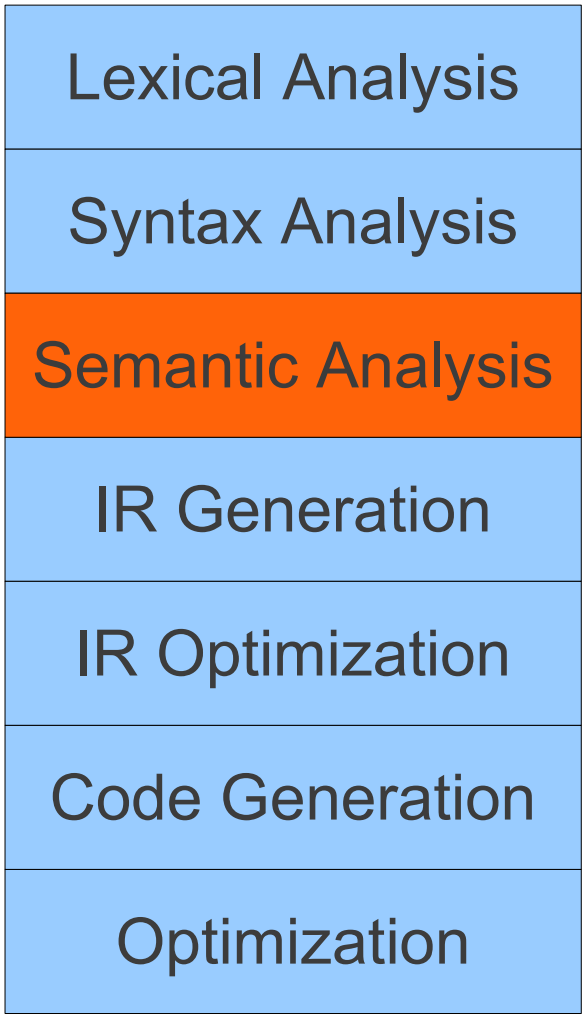
```
while (y < z) {
    int x = a + b;
    y += x;
}
```



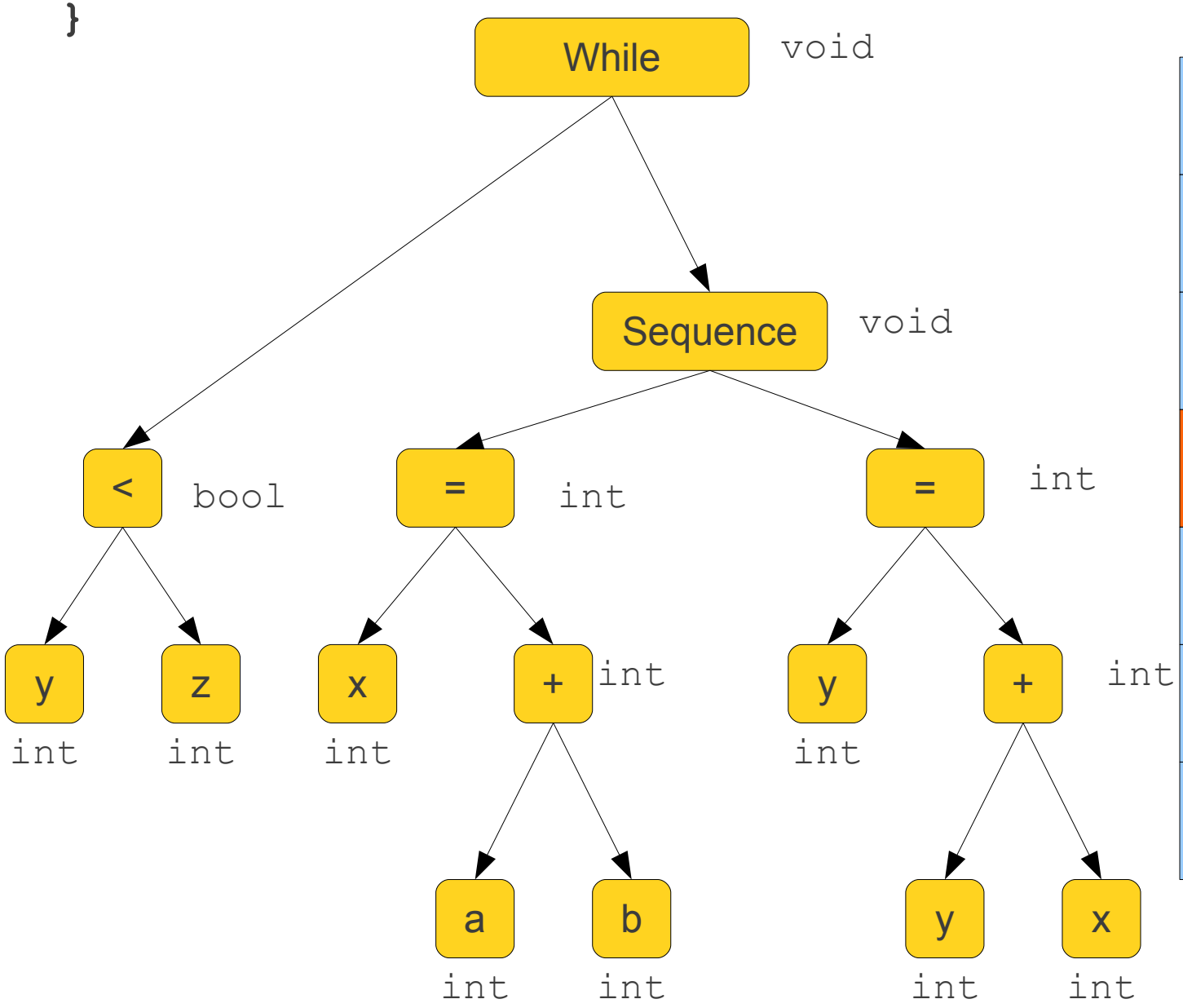
- Lexical Analysis
- Syntax Analysis
- **Semantic Analysis**
- IR Generation
- IR Optimization
- Code Generation
- Optimization

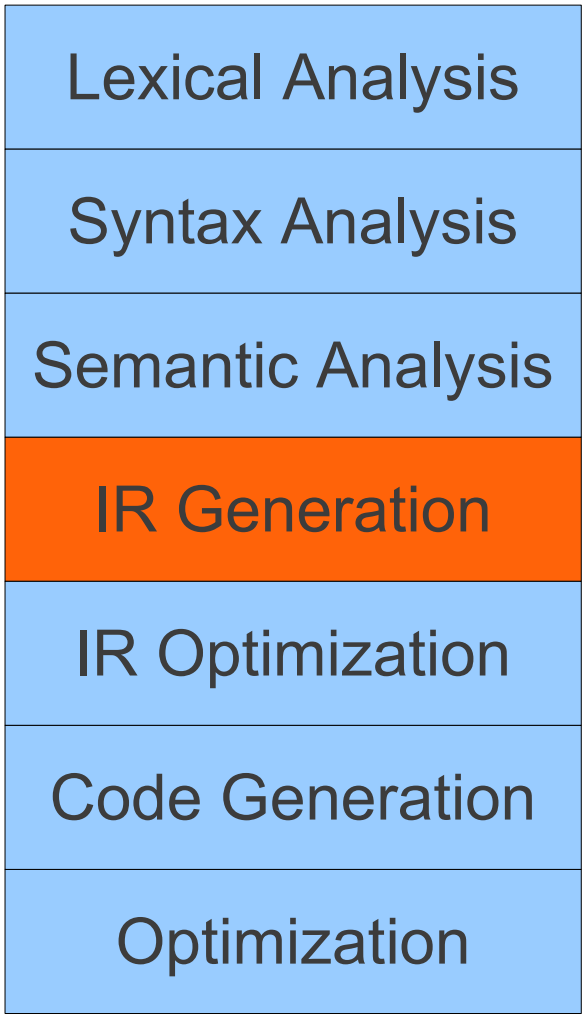
```
while (y < z) {
    int x = a + b;
    y += x;
}
```





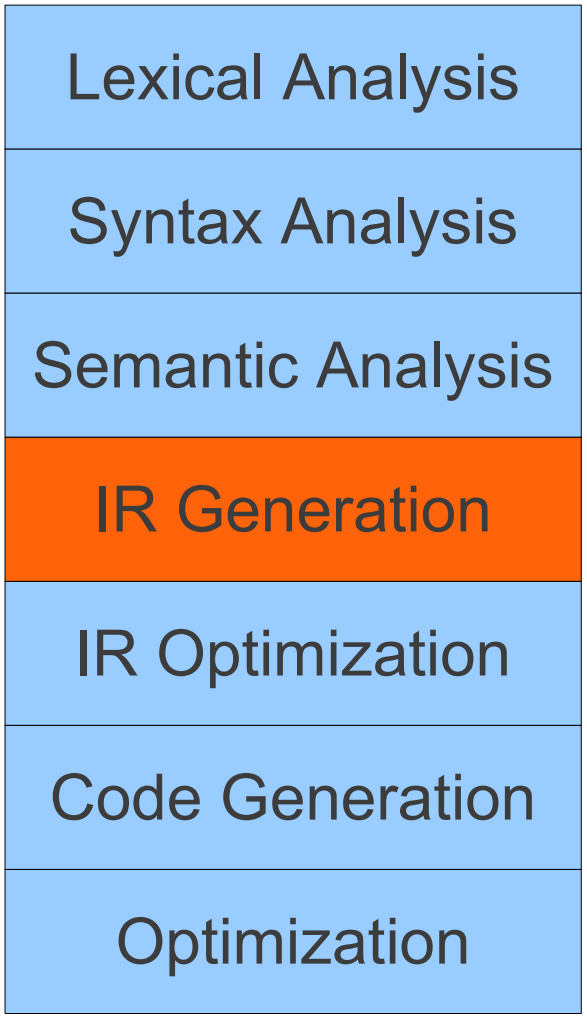
```
while (y < z) {
    int x = a + b;
    y += x;
}
```





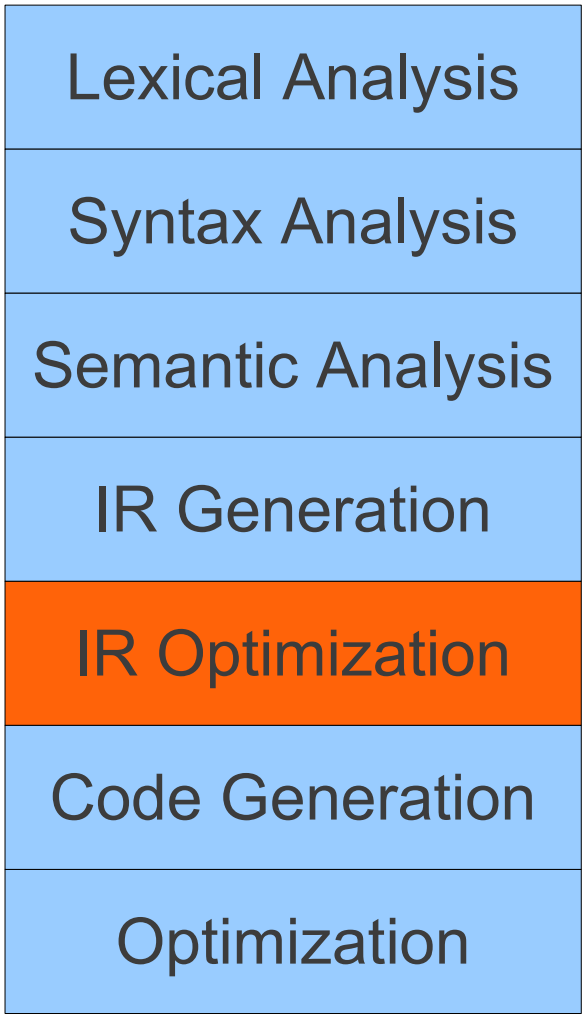
```
while (y < z) {
    int x = a + b;
    y += x;
}
```

```
Loop: x   = a + b
      y   = x + y
      _t1 = y < z
      if _t1 goto Loop
```



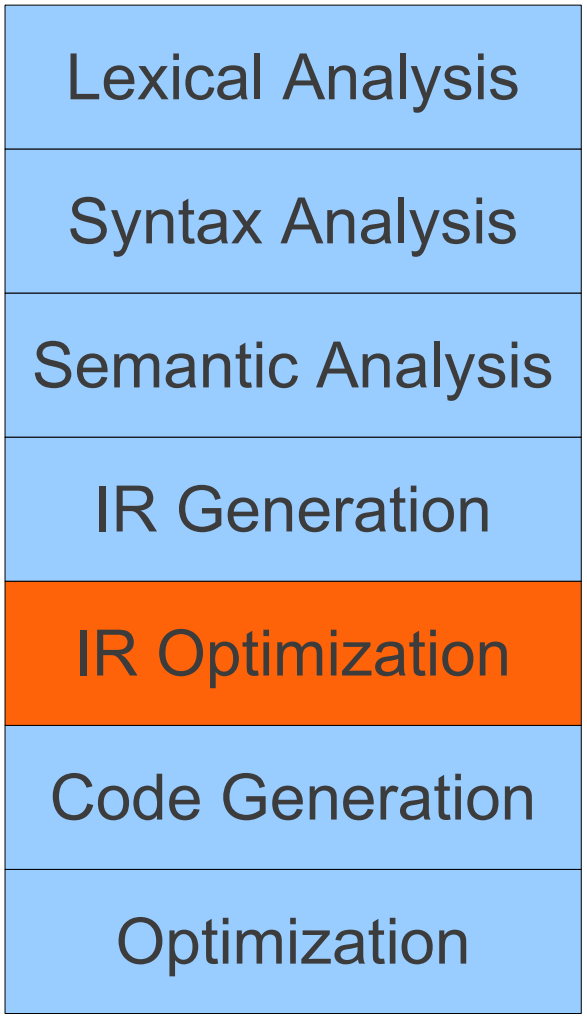
```
while (y < z) {
    int x = a + b;
    y += x;
}
```

```
Loop: x   = a + b
      y   = x + y
      _t1 = y < z
      if _t1 goto Loop
```



```
while (y < z) {
    int x = a + b;
    y += x;
}
```

```
        x    = a + b
Loop:   y    = x + y
        _t1  = y < z
        if _t1 goto Loop
```



```
while (y < z) {
    int x = a + b;
    y += x;
}
```

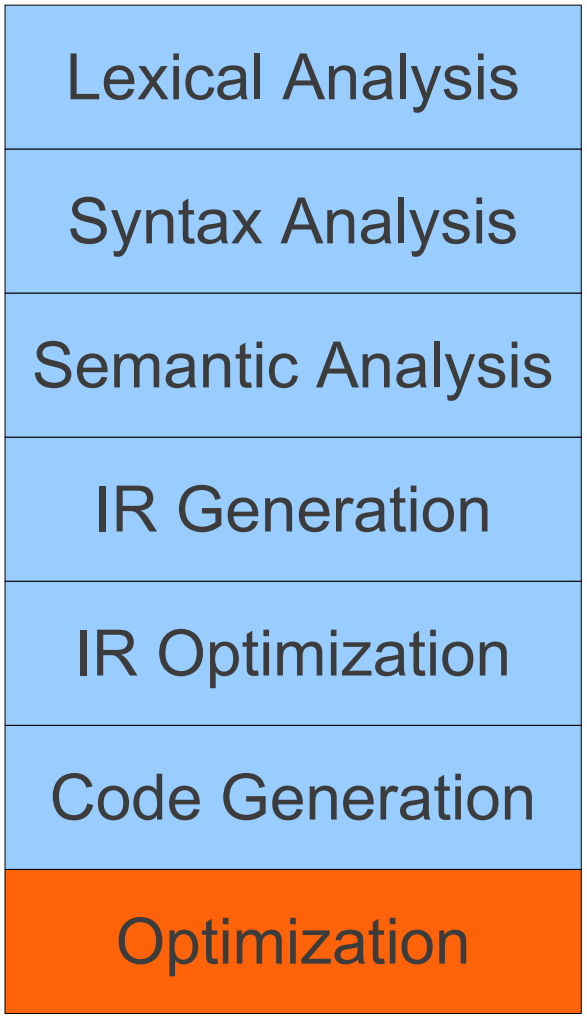
```
        x    = a + b
Loop:   y    = x + y
        _t1  = y < z
        if _t1 goto Loop
```

```
while (y < z) {
    int x = a + b;
    y += x;
}
```

```
        add $1, $2, $3
Loop:   add $4, $1, $4
        slt $6, $1, $5
        beq $6, loop
```

```
while (y < z) {
    int x = a + b;
    y += x;
}
```

```
        add $1, $2, $3
Loop:   add $4, $1, $4
        slt $6, $1, $5
        beq $6, loop
```



```
while (y < z) {
    int x = a + b;
    y += x;
}
```

```
        add $1, $2, $3
Loop:   add $4, $1, $4
        blt $1, $5, loop
```