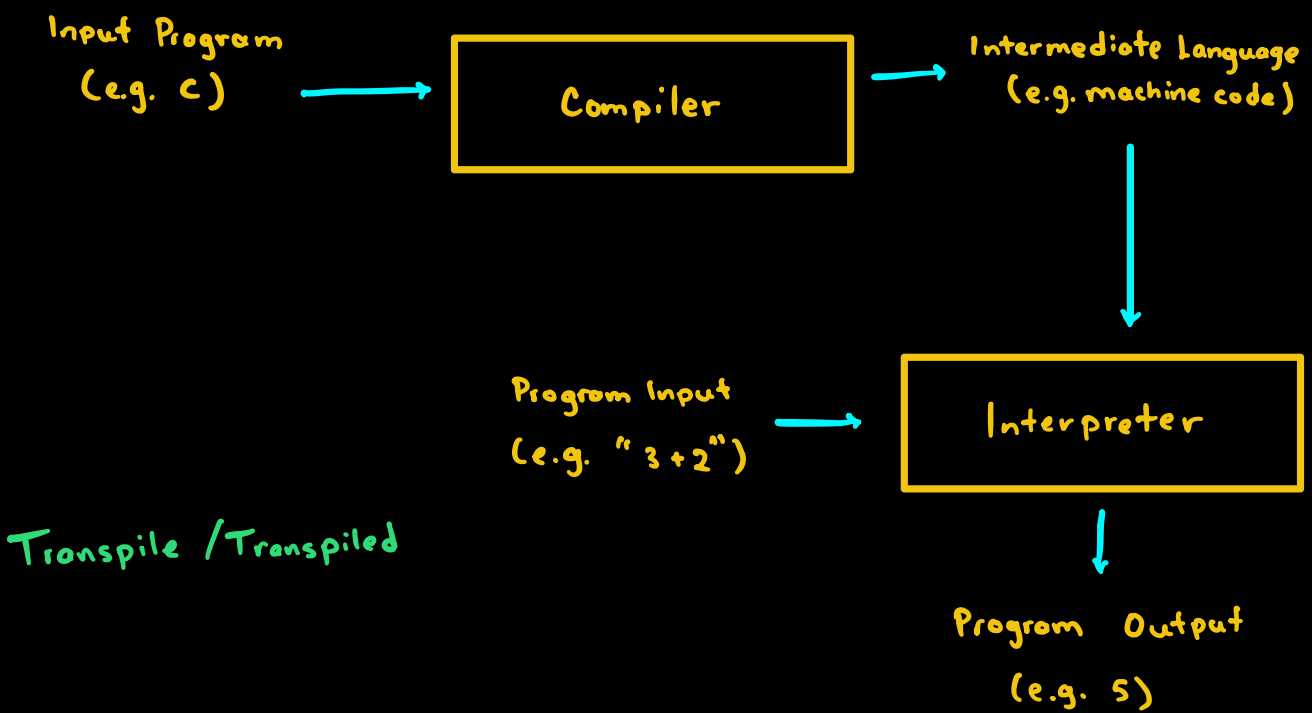
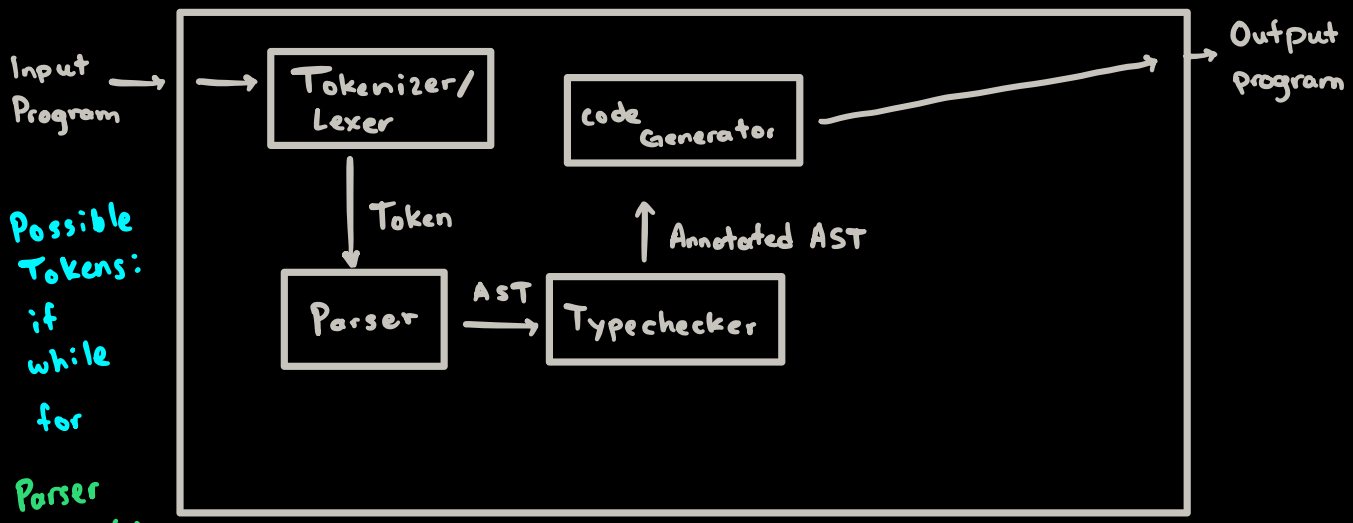
Intepreted Style

Compiled Style



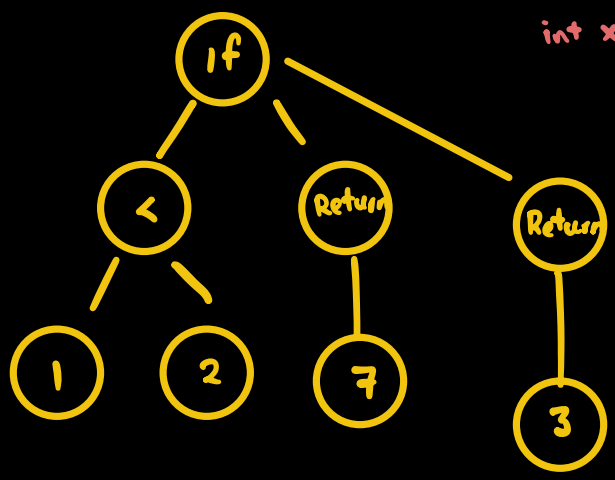
Transpile / Transpiled



Possible Tokens:
if
while
for

Parser Output:
Abstract Syntax Tree (AST)

```
if (1<2) {
    return 7;
} else {
  return 3;
}
```




```
int x = "foo"; //int = String
```

BNF

27:04

digit ::= '0' | '1'

number ::= digit | digit number

expression ::= number | expression '+' expression

example numbers:

0

1

01

101

example expression:

1101

101 + 110

(1 + 111) + (expression + expression

Language Design:

- Integers and bookeans
- Declare and initialize variables
- Perform typical arithmetic / logical operations

Var is a variable

num is a number

Type ::= 'int' | 'bool'

Vardec ::= '(' 'vardec' type var expression ')'

expression ::= num | 'true' | 'false' |

'(' op expression expression ')'

loop ::= '(' 'while' expression statement ')'

assign ::= '(' '=' var expression ')'

Statement ::= vardec | loop | assign

op ::= '+' | '-' | '&&' | '||'

program ::= Statement*

```
int x = 0;
while (x<10) {
    x = x+1;
}
```

```
(Vardec int x 0)
(while (< x 10)
    (= x (+ x 1)))
```

```
(Vardec int x 7)
(Vardec bool y true)
```

```
(Vardec int a (+ 1 2))
(Vardec bool b (&& false true))
```

Object language: Our language : My Lang

Metalanguage: what we are writing the compiler in : Java

Target language: what we are compiling to : Javascript

1:04:00