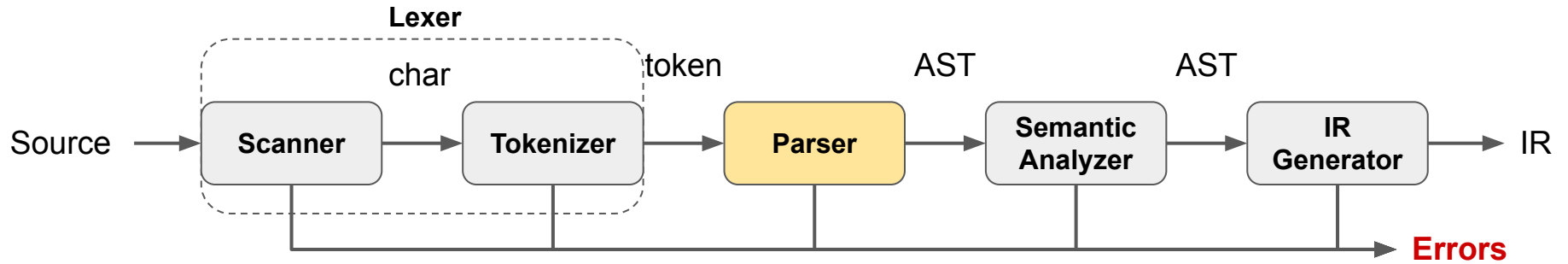


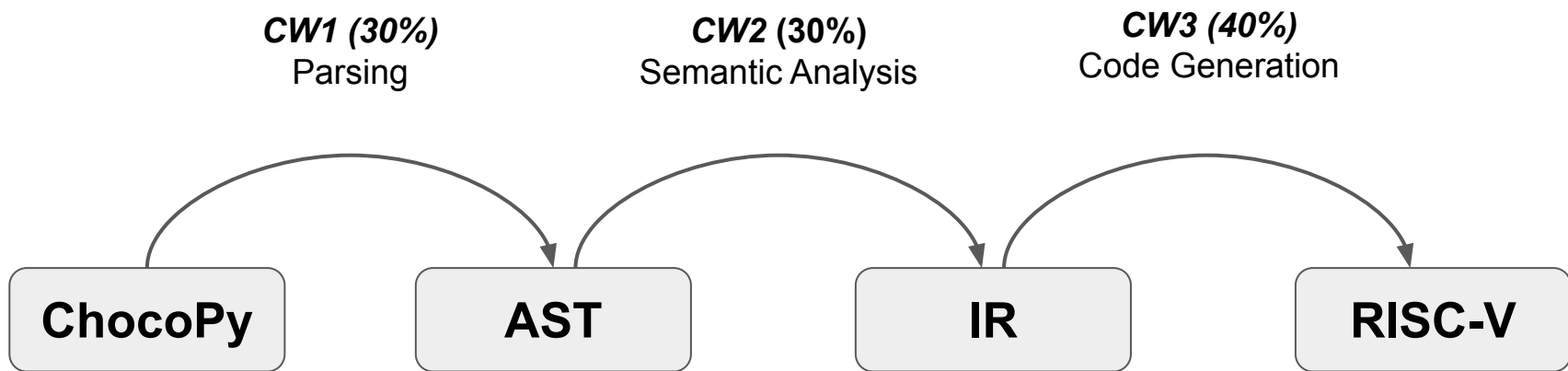
Compiling Techniques

Lecture 7: Coursework 1 - Intro

The Frontend



Coursework: A Python to RISC-V Compiler



Coursework Schedule

Week 1 (Jan 15)		Week 6 (Feb 26)	CW2
Week 2 (Jan 22)		Week 7 (Mar 4)	
Week 3 (Jan 29)		Week 8 (Mar 11)	
Week 4 (Feb 5)	CW1	Week 9 (Mar 18)	CW3
Week 5 (Feb 12)		Week 10 (Mar 25)	
Learning Week		Week 11 (Apr 1)	

Deadlines: [Friday noon](#)

Coursework

“Check out Learn” → “Compiling Techniques” → “Assessment”

A recursive-descent parser

CFG for function call

```
funcall ::= ID "(" arglist ")"  
arglist ::= ID argrep |  $\epsilon$   
argrep  ::= ", " ID argrep |  $\epsilon$ 
```

```
def parse_funcall():  
    match(ID)  
    match(LPAREN)  
    parse_arglist()  
    match(RPAREN)  
  
def parse_arglist():  
    if check(ID):  
        match(ID)  
        parse_argrep()  
  
def parse_argrep():  
    if check(COMMA):  
        match(COMMA)  
        match(ID)  
        parse_argrep()
```

Parser Class

```
class Parser:
```

```
    def check(self, expected : TokenKind) -> bool:  
        return self.lexer.peek().kind == expected
```

```
    def match(self, expected : TokenKind) -> Token:  
        if self.check(expected):  
            token = self.lexer.peek()  
            self.lexer.consume()  
            return token
```

```
        raise Exception(f"Error: token of kind ${expected} not found")
```

What is a token?

A token consists of a token class and other additional information.

Example: some token classes

IDENTIFIER	→	foo , main , cnt , ...
NUMBER	→	0 , -12, 1000 , ...
STRING_LITERAL	→	"Hello world!" , "a" , ...
EQ	→	==
ASSIGN	→	=
PLUS	→	+
LPAR	→	(
...	→	...

```
class Token:  
    Kind: TokenKind  
    Value: Any = None
```