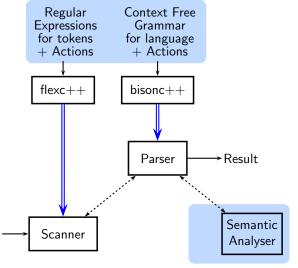
### Introduction to Flexc++ and Bisonc++

Department of Computer Science and Engineering, Indian Institute of Technology, Bombay



Jan. 2014



expression

with +, \*,

and integers

CS 306

Introduction to Lex and Yacc

# flexc++.bisonc++:

- lex script num.l to show token identification
- exp1.y to show a simple expression. Use the lex script exp.l
- exp2.y to show construction of a PLUS expression Interesting input: 1 + 2 + 3 + 4 + 5 + 6
- exp3.y fixes the problem by making + left associative
- exp4.y includes + and \* Interesting inputs: 1 + 2 \* 3 and 1 \* 2 + 3
- exp5.y fixes the above problem

IIT Bomb

CS 306 flexc++,bisonc++:

## The Interaction Between Scanner and Parser

3/1

Grammar

- Terminals and non-terminals get defined by the grammar
- Scanner identifies the tokens and communicates the details to the parser

Token Name	Token Lexemes	Token Value	Token Code
Number	"10"	10	NUM
	"345"	345	
	"03"	3	
+ operator	"+"		'+'
* operator	" *"		'*'

Uday Khedker IIT Bombay

scanner.ll parser.yy

flexc++,bisonc++:

The Interaction Between Scanner and Parser

flexc++

bisonc++

Uday Khedker

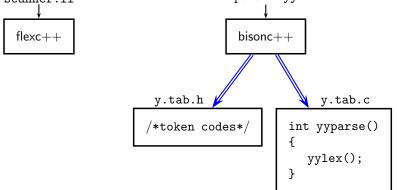
CS 306

IIT Bombay

scanner.ll parser.yy

flexc++,bisonc++:

The Interaction Between Scanner and Parser



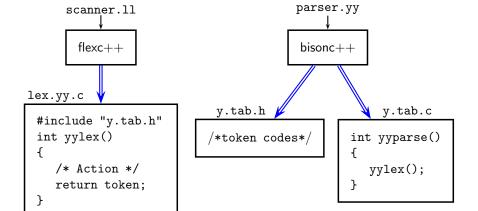
IIT Bombay

4/1

CS 306

# flexc++.bisonc++:

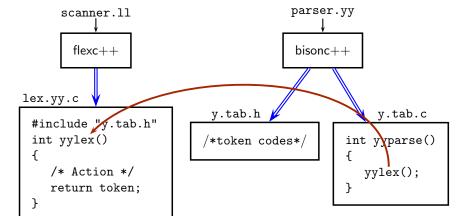
4/1



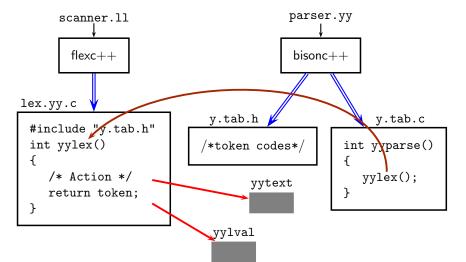
**Uday Khedker IIT Bombay**  CS 306 flexc++,bisonc++:

# The Interaction Between Scanner and Parser

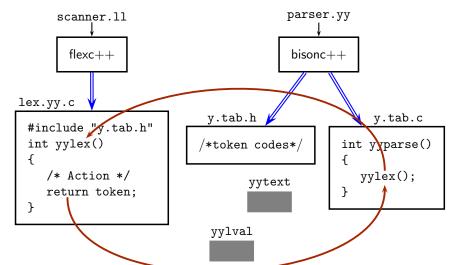
4/1



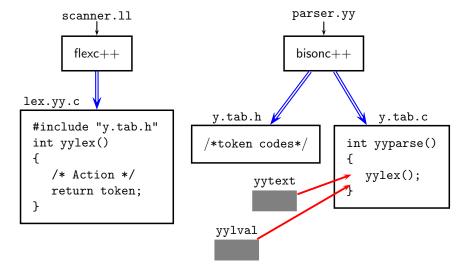
Uday Khedker IIT Bombay



4/1



Uday Khedker IIT Bombay



- Syntax: Precedences and associativities
- Semantics: Values of
  - Terminals (from scanner)
  - Non-terminals (in the parser)
- Unit testing:

Use of #if SCANTEST to test the scanner



**Notes** 

- Sample inputs
   a 20 + b 3
- a 20 | 1

- Use of "."
  - If a string can match a pattern and its prefix can also match a
  - ▶ If a string can match a pattern and its prefix can also match a pattern, the prefix matching is ignored in favour of the longer string.
- The Lex and Yacc Page http://dinosaur.compilertools.net/



{

};

class Scanner:

private:

public ScannerBase

flexc++.bisonc++:

```
public ParserBase
Scanner d_scanner;
public:
```

```
public:
   Scanner(...); /* explicit? */
   int lex();
```

int parse();

IIT Bombay

};

CS 306

{

class Parser:

private:

class Scanner:

public:

flexc++.bisonc++:

```
public ParserBase
                           {
Scanner d_scanner;
public:
```

```
int lex();
```

public ScannerBase

```
private:
  int parse();
protected: /*inherited*/
 STYPE_
           d_val_;
```

class Parser:

private:

Scanner(...); /\* explicit? \*/

};

CS 306

{

Uday Khedker

};

{

class Scanner:

public:

public ScannerBase

flexc++.bisonc++:

```
class Parser:
     public ParserBase
{
 Scanner d_scanner;
 public:
    Parser()
    { d_scanner.setSval(
                &d_val_);
    }
 private:
    int parse();
```

protected: /\*inherited\*/

d\_val\_\_;

```
int lex();
void setSval(
          ParserBase::STYPE_ * val);
ParserBase::STYPE_ * getSval();

private:
    ...
ParserBase::STYPE_ * dval;
};
```

Scanner(...); /\* explicit? \*/

};

STYPE\_

CS 306

{

};

class Scanner:

public:

public ScannerBase

flexc++.bisonc++:

```
class Parser:
     public ParserBase
{
  Scanner d_scanner;
  public:
    Parser()
    { d_scanner.setSval(
                 &d_val_);
  private:
    int parse();
  protected: /*.nherited*/
   STYPE_
              d_val_
};
```

```
int lex();
void setSval(
          ParserBase::STYPE_ * val);
ParserBase::STYPE_ * getSval();
private:
    ...
ParserBase::STYPE_ * dval;
```

Scanner(...); /\* explicit? \*/

**CS 306** 

{

class Scanner:

public ScannerBase

7/1

```
public ParserBase
Scanner d_scanner;
public:
  Parser()
  { d_scanner.setSval(
              &d_val_);
private:
  int parse();
protected: /*.nherited*/
 STYPE_
           d_val__
```

```
public:
   scarner(...); /* explicit? */
    int lex()
    void setSval(
          ParserBase::STYPE_ * val);
   ParserBase::STYPE_ * getSval();
 private:
   ParserBase::STYPE_ * dval;
};
```

};

**CS 306** 

class Parser:

{

class Scanner:

public:

public ScannerBase

flexc++.bisonc++:

```
class Parser:
     public ParserBase
  Scanner d_scanner;
  public:
    Parser()
    { d_scanner.setSval(
                &d_val_);
  private:
    int parse();
  protected: /*.nherited*/
```

d\_val\_\_

```
int lex();
void setSval(
    ParserBase::STYPE_ * val);
ParserBase::STYPE_ * getSval();

private:
    ...
ParserBase::STYPE_ * dval;
};
```

};

STYPE\_

**CS 306** 

Expr

CS 306

| Expr '\*' Expr | NUM

Expr '+' Expr

The process of parsing

Parsing Stack 10 + 20 \* 3 <u>eof</u> NUM

Action: Shift NUM

Uday Khedker

## 7 in Orderview or Count Reduce 1 areing

flexc++.bisonc++:

Expr

Grammar

CS 306

Expr '+' Expr

The process of parsing

NUM
Parsing
Stack

Action: Reduce by "Expr: NUM"

Expr

Grammar

CS 306

Expr '+' Expr

The process of parsing

Action: Shift +

Expr

Grammar

CS 306

Expr

Expr '+' Expr

The process of parsing

Action: Shift NUM Parsing Stack

**Uday Khedker** 

 $\mathsf{top} \to$ 

Expr

Grammar

CS 306

Expr '+' Expr Expr '\*' Expr NUM

Expr

The process of parsing

Action: Reduce by "Expr: NUM" Parsing Stack

**Uday Khedker** 

 $\mathsf{top} \to$ 

### isilig

8/1

Expr

 $\mathsf{top} \to$ 

Grammar

CS 306

Expr '\*' Expr

Expr '+' Expr

The process of parsing

Expr

+ Expr Action: Shift \* or Reduce by "Expr: Expr + Expr"?

Parsing Stack

Expr

 $\mathsf{top} \to$ 

Grammar

CS 306

Expr '\*' Expr NUM

Expr '+' Expr

The process of parsing

\* Expr

Expr

Stack

10 20 eof NUM

Action: Shift NUM Parsing

**Uday Khedker** 

Expr

Grammar

CS 306

Expr : Expr '+' Expr | Expr '\*' Expr | NUM

Expr

Stack

The process of parsing

+ Action: Reduce by "Expr: NUM"

Parsing

Uday Khedker

An Overview of Shift Reduce Parsing

• Grammar

flexc++.bisonc++:

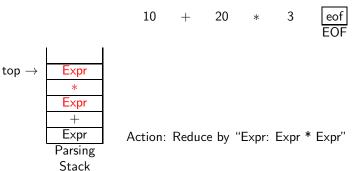
Expr

CS 306

| Expr '\*' Expr | NUM

Expr '+' Expr

The process of parsing



Uday Khedker

## 7 in Overview of Simt Reduce 1 dising

flexc++.bisonc++:

Expr

 $\mathsf{top} \to$ 

Grammar

CS 306

| Expr '\*' Expr | NUM

Expr '+' Expr

• The process of parsing

+ Expr Action: Reduce by "Expr: Expr + Expr"
Parsing Stack

Uday Khedker

Expr

Grammar

CS 306

Expr '+' Expr

The process of parsing

Expr

Stack

Action: Accept Parsing

**Uday Khedker** 

flexc++,bisonc++:

An Overview of Attribute Evaluation

10

NUM

20

eof

```
Expr : Expr '+' Expr { $$ = $1 + $3;}
| NUM
```

• The process of parsing and attribute evaluation

Stack

```
Parsing Value
```

Stack

IIT Bombay

**Uday Khedker** 

Grammar and the associated actions

Parsing

Stack

CS 306

The process of parsing and attribute evaluation

$$\mathsf{op} o oxed{\mathsf{NUM}} oxed{10}$$
 \$1

Value

Stack

Uday Khedker IIT Bombay

10 + 20 +

eof

Grammar and the associated actions

The process of parsing and attribute evaluation

10

10 \$\$ Parsing Value Stack Stack

IIT Bomba

eof

10

20

NUM

eof

• Grammar and the associated actions

• The process of parsing and attribute evaluation

$$top 
ightarrow egin{array}{c|c} + & & & & \\ \hline Expr & & 10 \\ Parsing & Value \\ Stack & Stack \\ \end{array}$$

IIT Bombay

Uday Khedker

## 7.11 Greatien et 71001.bate Etalaatien

10 + 20

flexc++.bisonc++:

Grammar and the associated actions

• The process of parsing and attribute evaluation

$top \to$	NUM	20	\$1
	+ Expr Parsing	10 Value	
	Stack	Stack	

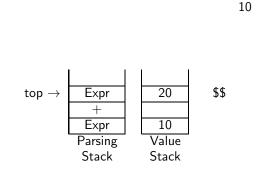
IIT Bombay

### 711 Overview of 71th Bate Evaluation

flexc++.bisonc++:

• Grammar and the associated actions

• The process of parsing and attribute evaluation



IIT Bombay

20

### 7th Sterview of Attribute Evaluation

10 + 20

flexc++.bisonc++:

• Grammar and the associated actions

• The process of parsing and attribute evaluation

ĺ	İ	l	1
$top \to$	Expr	20	\$3
·	+		
	Expr	10	\$1
•	Parsing	Value	
	Stack	Stack	

IIT Bombay

Grammar and the associated actions

CS 306

• The process of parsing and attribute evaluation

 $top \rightarrow \begin{array}{|c|c|c|c|}\hline Expr & 30 & \$\$\\ \hline Parsing & Value & \\ Stack & Stack & \\ \hline \end{array}$ 

Uday Khedker IIT Bombay