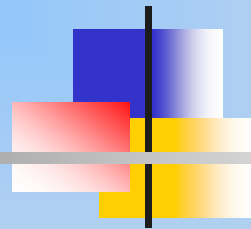


CSE302: Compiler Design



Instructor: Dr. Liang Cheng
Department of Computer Science and Engineering
P.C. Rossin College of Engineering & Applied Science
Lehigh University

April 17, 2007



Outline

- Recap
- Intermediate code generation
- Things related to project part II
- Summary



Translation of Control Flows

- $S \rightarrow \text{if} (B) S1$
- $S \rightarrow \text{if} (B) S1 \text{ else } S2$
- $S \rightarrow \text{while} (B) S1$



Translation of Boolean Expressions

- $B \rightarrow B1 \ || \ B2$
- $B \rightarrow B1 \ \&\& \ B2$
- $B \rightarrow !B1$
- $B \rightarrow \text{true}$
- $B \rightarrow \text{false}$
- $B \rightarrow E1 \ \text{rel} \ E2$



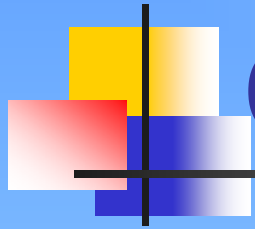
Three-address Code of Switch

```
■ switch ( E ) {  
    case V1 : S1  
    case V2 : S2  
    ...  
    case Vn-1 : Sn-1  
    default : Sn  
}
```



Three-address Code of Procedures

- $n = f(a[i]);$
 - $t1 = i * 4$
 - $t2 = a[t1]$
 - param t2
 - $t3 = \text{call } f, 1$
 - $n = t3$

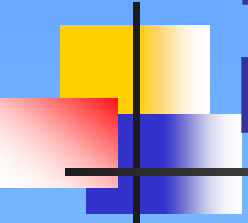


Outline

- Recap
- Intermediate code generation
- Things related to project part II
- Summary

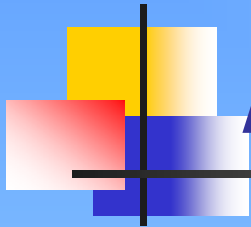


- Take a specification file (grammar) and produce an output file for the parser
 - Input: <filename>.y
 - {definitions}
 - %%
 - {productions/rules}
 - %%
 - {auxiliary routines}
 - Output: y.tab.c
 - LALR parser



How to Specify Associativity and Precedence in Yacc?

- `%left` and `%right`
 - All tokens in the same line have the same precedence level and associativity
 - The lines are listed in order of increasing precedence
 - `%left '+' '-'`
 - `%left '*' '/'`
 - `%right '**'`
- `%nonassoc`
 - `A .LT. B .LT. C`



An Example

- %right '='
- %left '+' '-'
- %left '*' '/'

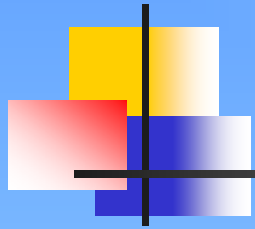
- %%
- expr : expr '=' expr
- | expr '+' expr
- | expr '-' expr
- | expr '*' expr
- | expr '/' expr
- | **id**;



Precedence of Unary Operators

- %right '='
- %left '+' '-'
- %left '*' '/'

- %%
- expr : expr '+' expr
- | expr '-' expr
- | expr '*' expr
- | expr '/' expr
- | '-' expr %prec '*'
- | **id**;



If-Else Statements

- Yacc would report a shift/reduce conflict. Yacc will "solve" this conflict by associating the else with the nearest if.



Support for Arbitrary Value Types in Yacc

- A parse stack and a value stack
- Floating-point values
 - `#define YYSTYPE double`
 - `#ifndef YYSTYPE`
 - `typedef int YYSTYPE`
 - `#endif`
 - `YYSTYPE yylval`



Integrate with Lexer

- **calc.l**

```
%{
%}
number    [0-9]+(\\.[0-9]+)?
%%
[ ]       { /* skip blank */ }
{number} {
    sscanf(yytext, "%lf", &yyval);
    return NUMBER;
}
\\n|.     {return yytext[0];}
```

- flex calc.l

- yacc calc.y

- gcc y.tab.c -ly -ll

Instructor: Dr. Liang Cheng

- **calc.y**

```
%{#include <stdio.h>
#define YYSTYPE double
%}
%token NUMBER
%%
lines : lines expr '\\n' {printf("%g\\n",$2);} | lines '\\n' | ;
expr  : expr '+' term   {$$ = $1 + $3;}
      | expr '-' term   {$$ = $1 - $3;}
      | term             {$$ = $1;}
term  : term '*' factor  {$$ = $1 * $3;}
      | term '/' factor  {$$ = $1 / $3;}
      | factor           {$$ = $1;}
factor : '(' expr ')' {$$ = $2;} | NUMBER {$$ = $1;}
%%
```

- **#include "lex.yy.c"**

CSE302: Compiler Design

04/17/07



Support for Arbitrary Value Types

- **YYSTYPE** may be a union type

```
%union {  
    body of union ...  
}
```

- This declares the Yacc value stack, and the external variables `yylval` and `yyval`, to have type equal to this union.
- `-d`: the union declaration is copied onto the `y.tab.h`
- The union member names are associated with terminals and nonterminals
 - `%token <name> _TOKEN`
 - `%type <name> _nonterminal`



Outline

- Recap
- Intermediate code generation
- Things related to project part II
- **Summary**