

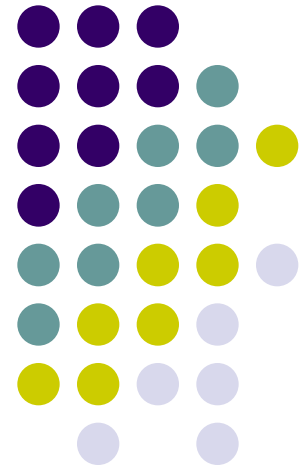
CS 460

Programming Languages

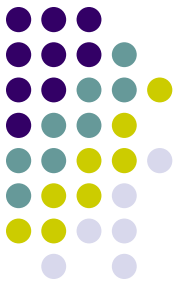
Fall 2023

Dr. Watts

(4 December 2023)

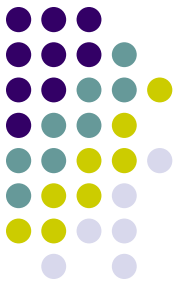


Assignments



- Please make sure you look at the e-mail sent by the script. Avoid the easy to fix errors!
- For example: Exercise 4
 - Spec indicates that the functions should be “mergesort” and “quicksort” – not “merge_sort” and “quick_sort”
- Exercise 5 spec and framework posted
 - New money.h file includes
 - New friend function
 - In cents; // recommended
 - Doxygen website generation

Enumerated types



- Why did we create this?

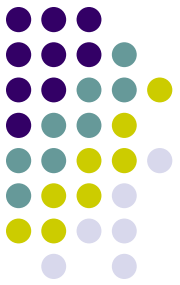
```
enum token_type
```

```
{
```

```
    NONE, IDKEY_T, NUMLIT_T, LISTOP1_T, PLUS_T, MINUS_T, GT_T, LT_T, TRUE_T,  
    FALSE_T, DIV_T, MULT_T, EQUALTO_T, GTE_T, LTE_T, LPAREN_T, RPAREN_T,  
    SQUOTE_T, IDENT_T, IF_T, COND_T, DISPLAY_T, NEWLINE_T, AND_T, OR_T,  
    NOT_T, DEFINE_T, LET_T, LISTOP2_T, NUMBERP_T, LISTP_T, ZEROP_T, NULLP_T,  
    EOFP_T, MODULO_T, ROUND_T, READ_T, ELSE_T, STRLIT_T, ERROR_T, EOF_T,  
    MAX_TOKENS
```

```
};
```

- How do you use this in a program?
- Benefits?
- Drawbacks?



Project 3

- PL460 to C++
- Code generation
- Spec and Framework posted
- Project3Framework contains

makefile

CodeGenerator.cpp

LexicalAnalyzer.h

SyntacticalAnalyzerP2.cpp

Object.h

README.txt

run1

Project3.cpp

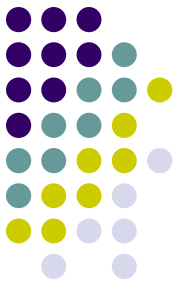
CodeGenerator.h

LexicalAnalyzer.o

SyntacticalAnalyzerP2.h

Object.o

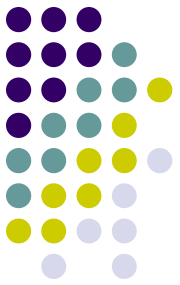
P3Test1.pl460



Project 3

- Project 2 Syntactic Analyzer will make calls to Code Generator to write to .cpp file
- Look at the grammar
 - Insertion of calls to CodeGenerator
 - Where?
 - Driven by grammar
 - Calls to WriteCode in SyntaxAnalyzer
 - What strings should be written?

Project 3

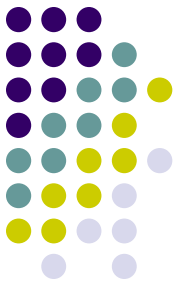


- Sample PL460 program

```
(define (aFunction)
  "Hello world"
)
(define (main)
  (display 0)
  (newline)
  (display (aFunction))
  (newline)
)
(main)
```

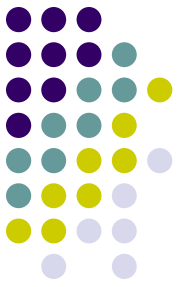
- Corresponding C++ program
- Modifications to generating code in SyntaxAnalyzer?

Project 3



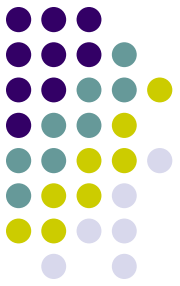
- Blue grammar rules
- Table of corresponding code snippets
- Questions?

Expressions and Assignment Statements (Chapter 7)

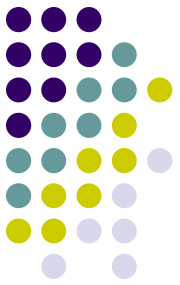


- Arithmetic Expressions
- Overloaded Operators
- Type Conversions
- Relational and Boolean Expressions
- Short-Circuit Evaluation
- Assignment Statements
- Mixed-Mode Assignment

Relational and Boolean Expressions



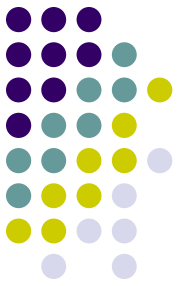
- `if (a == b)`
- `cout << a == b << endl;`
- Counting applications



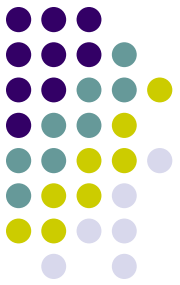
Short-Circuit Evaluation

- `if (a == b and c < d)`
- `if (a == b or c < d)`
- `if (function1 (a, b) and function2 (b, c))`
- `if (function1 (a, b) or function2 (b, c))`
- Side effects
- `if (letter == 'a' || 'e' || 'i' || 'o' || 'u')`
- C++ vs Java

Assignment Statements

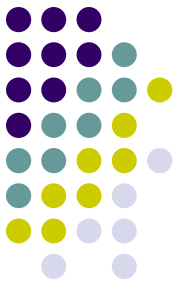


- As independent statements
- As part of an expression
- Return value



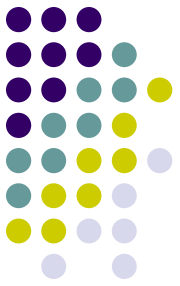
Type Conversions

- `int a;`
- `float b;`
- `char c;`
- `Float (a);`
- `(unsigned short) c;`



Mixed-Mode Assignment

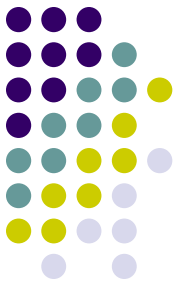
- Coalescing / coercion
- In FORTRAN, C, and C++, any numeric value can be assigned to any numeric scalar variable; whatever conversion is necessary is done
- In Pascal, integers can be assigned to reals, but reals cannot be assigned to integers (the programmer must specify whether the conversion from real to integer is truncated or rounded)
- In Java, only widening assignment coercions are done
- In Ada, there is no assignment coercion



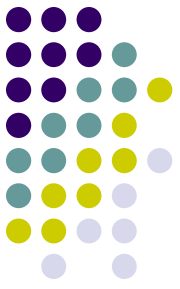
Control Structures

- Structured programming defines 3 types
 - Sequential
 - Decision (also known as selection)
 - Looping (also known as iterative)

Sequential Control Structures

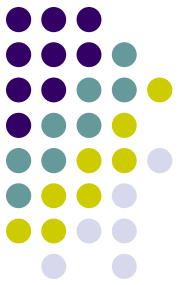


- C++
- PL460



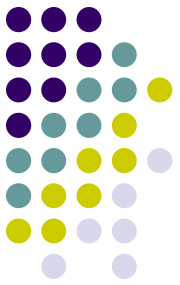
Selection Control Structures

- C++
- PL460
- If statements – single vs dual branches
- Selection entry point vs selection point
- Switch vs cond
- Switch vs if-else chain



Iterative Control Structures

- Test at Top vs Test at Bottom
- C++
- PL460



Flow control disrupters

- return
- exit
- break
- continue