# CS480

# Compilers

# Chapter 1, 2 (and section 7.6)

Pages 1-51, 60 - 62, 429 - 440

February 4, 2013

# Compiler

- Definition

- Difference between compiler and
  - Eclipse
  - MS Visual Studio
  - Parts of each?

- Other source code analysis tools?
- gcc options?

`-g` `-pg` `-E` `-O#` {*0..3*} `-Os`

```
//              The code below has been seeded with memory errors to
//              demonstrate the use of Valgrind.
//*****************************************************************************

#include <stdio.h>
#include <stdlib.h>

void printer (char * str)
{
  printf ("%s", str);
}

int main (int argc, char** argv)
{
  char *pLetter;

  pLetter = (char*) malloc (sizeof(char));

  printer ("HELLO WORLD");
  printer ("\n");
  printer ("GOODBYE WORLD");
  printer("New Commit!");
  printer ("\n");

  *pLetter = 'P';
  printf ("%c\n", *pLetter);
  free (pLetter);
```

```
39  int readInfo( fstream & in, playerInfo info[], const int SIZE);
40  void printDisplay(ostream & out, string title);
41  void printPlayer(ostream & out, playerInfo info[], int place);
42  void printTrue(ostream & out, playerInfo info[], int size, int place);
43  void printAll(ostream & out, playerInfo info[], int size);
44  void printHisto(ostream & out, playerInfo info[], int place);
45
46  int main() { ... }
103 /*****************************************************************
104 Function:     readInfo
105 Description:  Reads the infomation of a team from a file and
106               stores it into an array of structs
107 Parameters:   in a fstream, info[]-an array of the struct
108               playerInfo, SIZE a const int the highest possible
109               number of players
110 Returned:     numbPlayers, the real number of players
111 *****************************************************************/
112 int readInfo( fstream & in, playerInfo info[], const int SIZE)
113 {
114   int numbPlayers=0;
115   cout >> "READ INFO" >> endl;
116   for( int i=0; (i<SIZE) && (in >> info[i].number); i++)
117   {
118     in >> info[i].nameLast >> info[i].nameFirst;
119     in >> info[i].minutes >> info[i].goals;
120     in >> info[i].shots>> info[i].shotOG;
121
122     info[i].shotPrecent[0] = info[i].goals/(info[i].shots*1.0);
123     info[i].shotPrecent[1] = info[i].shotOG/(info[i].shots*1.0);
```

# A case against syntax highlighting

- Syntax highlighting in English

Alice was not a bit hurt, and she jumped up on to her feet in a moment: she looked up, but it was all dark overhead; before her was another long passage, and the White Rabbit was still in sight, hurrying down it. There was not a moment to be lost: away went Alice like the wind, and was just in time to hear it say, as it turned a corner, `Oh my ears and whiskers, how late it's getting!' She was close behind it when she turned the corner, but the Rabbit was no longer to be seen: she found herself in a long, low hall, which was lit up by a row of lamps hanging from the roof.

http://www.linusakesson.net/programming/syntaxhighlighting/index.php

# Two Jobs

- Analysis

- Synthesis/Generation

# Compiler, in detail

- Lexical Analysis `taxes = salary * 0.30 + 500;`
- Syntactic Analysis
  - AST
- Semantic Analysis

# Compiler, in detail

- Intermediate code generator

```
taxes = salary * 0.30 + 500;
```

- Code generation
  - Optimization

- What's the difference between assembly lang and machine lang?

# Example?

What does the compiler need to do here?

```
 1#include <stdio.h>
 2#if SYS == RH5.1
 3#define LIB  "rh51.h"
 4#elif SYS == RH6.0
 5#define LIB "rh60.h"
 6#else
 7#define LIB "rh62.h"
 8#endif
 9#include LIB
10
11#define MAX(x,y) ((x) > (y) ? (x) : (y))
12
13main ()
14{
15   int i = 3, j = 7;
16   printf ("%d", MAX(i,j));
17}
```

# Compiler Terms

- Front End/Back End

- Passes

# Linker/Loader

- What's a link error?

# Ch 2: One-Pass Compiler

- Convert infix to postfix (RPN) expressions
  - why is RPN useful?

- Infix: 5 + 2 / 4 + 1

- Postfix:

# Language

- Context Free Grammar
  - components?
- Can you give a grammar for postfix notation?
  - Assume single digit ints

- Parse Tree?
  - Components?
  - Parse tree for 1 2 + 2 -

# Grammars

```
bin -> bin + bin
bin -> bin - bin
bin -> 0 | 1
```

- Ambiguous?
- Associativity?
  - Is + left or right associative?

# Grammars

```
expr -> expr + term | expr - term | term
term -> term * factor | term / factor | factor
factor -> digit | ( expr )
```

- Ambiguous?

- Left or Right associative

  \+ - * /

- **9 + 2 * 5**

# Syntax Directed Translation

- Grammar for infix
- Grammar of postfix
- ???
- Translation

# Annotated Parse Trees

- Semantic rules
- Synthesized Attributes
- Traversal
- Emit Translation

# Parsing!

- Top Down

- Bottom Up

# Symbol Tables (p 60-62, 429-440)

- Lexeme
- Runtime location
  - Where could a variable be stored at runtime?
- Type
- Level/Scope
- Array dimension/Number of parameters

# Symbol Table

- Cross Reference Information

- Data structures
  - Unordered linear list
  - Ordered linear list
  - Binary search tree
  - AVL tree

  - advantages/disadvantages
  - Insert/Delete/Search O(??)

# Makefile – find the errors

list.h

```
struct List
{
   int data;
   struct List *psNext;
};
```

```
all: main

main: main.o list.o
   gcc -o main main.o list.o -g

main.o: main.c
   gcc -o main.o -c main.c -g

list.o: list.c
   gcc -o list.o -c list.c -g

run_valgrind:
   valgrind -v --leak-check=y....
```

list.c

```
#include “list.h”

int listInsert(struct List **psHead, int data)
{
  struct List *psNewData;

  psNewData = (struct List*) malloc(sizeof(struct List));
  psNewData->data = data;
  psNewData->psNext = NULL;
```