Compiler Report

CS 4000 : Compiler Construction

Bryant Nelson

4/30/2010

Table of Contents

Introduction

- This is a completed project.
    - Including all extra credit.
- An interpreter is included in the same executable as the compiler.
- The executable is named “executable.exe.jpg” to fool mail.ttu.

Compiler Information

Selected Features

All of the following are supported by the compiler:

- Assignment between arrays of the same size. This makes a deep copy of one into the other.
- Regular exponentiation.
    - E^E
- Using a foreach loop to index through an inner array.
    - foreach int in arr[1, 2, 1]
- When using a foreach loop, if you modify the index variable it modifies the original array.

Output

My compiler outputs a “.ALT” file which contains the Symbol Table and the Quad Table of the compiled program. My interpreter takes this type of file as input.

Interpreter Information

Selected Features

All of the following are supported by the interpreter:

- Runtime Errors:
    - Array Index Out Of Range
    - Divide By Zero
    - Zero Raised to Negative Exponent
- Optional keyboard input if no input file is provided.
- Optional screen output if no output file is provided.

Modified Grammar

Non-Terminal		Production	Selection Set
P	→	program **D S** end	
D	→	**IL D1**	
	→	begin	
D1	→	array[**CL D**	
	→	integer **D**	
IL	→	id **IL1**	
IL1	→	, **IL**	
	→	:	
CL	→	cons **CL1**	
CL1	→	, **CL**	
	→	]	
ID	→	id **ID1**	
ID1	→	[**EL**	
	→	ε	{ ":=", "+", "-", "*", "/", "^", ")", ",", "<", "=", ">", "]", "and", "or", ";", "end", "fi", "od", "esac", "do", "then", "else" }
EL	→	**E EL1**	
EL1	→	**, EL**	
	→	]	
IDL	→	**ID IDL1**	
IDL1	→	, **IDL**	
	→	)	
E	→	**T E1**	
E1	→	+ **T E1**	
	→	- **T E1**	
	→	ε	{ ")", ",", "<", "=", ">", "]", "and", "or", ";", "end", "fi", "od", "esac", "do", "then", "else" }
T	→	**B T1**	
T1	→	* **B T1**	
	→	/ **B T1**	
	→	ε	{ "+", "-", ")", ",", "<", "=", ">", "]", "and", "or", ";", "end", "fi", "od", "esac", "do", "then", "else" }
B	→	**F B1**	
B1	→	^ **F B1**	
	→	ε	{ "+", "-", "*", "/", ")", ",", "<", "=", ">", "]", "and", "or", ";", "end", "fi", "od", "esac", "do", "then", "else" }
F	→	**ID**	
	→	cons	
	→	exp(**E**,**E**)	
	→	(**E**)	
C	→	**X C1**	

C1	→	or **X C1**	
	→	ε	{ "do", "then", ")", "]" }
X	→	**Y X1**	
X1	→	and **Y X1**	
	→	ε	{ "or", "do", "then", ")", "]" }
Y	→	**E Y1**	
	→	not(**C**)	
	→	[**C**]	
Y1	→	< **E**	
	→	> **E**	
	→	= **E**	
S	→	**ID** := **E S1**	
	→	read(**IDL S1**	
	→	write(**IDL S1**	
	→	readln(**IDL S1**	
	→	writeln(**IDL S1**	
	→	case **C** do **S M S1**	
	→	while **C** do **S** od **S1**	
	→	if **C** then **S S2**	
	→	foreach id in **ID** do **S** od **S1**	
	→	with **D** begin **S** end **S1**	
S1	→	; **S3**	
S2	→	fi **S1**	
	→	else **S** fi **S1**	
S3	→	**S**	
	→	ε	{ "end", ":", "od", "fi", "esac", "else" }
M	→	: **C** do **S M**	
	→	esac	

Grammar Modification Notes

1. The addition of **S3** is to ensure that every line ends in a ';', excluding the final 'end'.
2. The foreach loop has been altered by adding **ID** as the second argument. This allows the programmer to iterate through a sub-array.
3. The original definition for exponentiation is included as well as the revised definition.

Semantic Notes

1. My "readln()" function expects each identifier argument to be on a different line.
2. My "writeln()" function writes each identifier argument on a different line.

Test Cases

Program 1

Compilation / Execution

```
E:\Compiler Construction\Compiler Testing\CS 4000 Compiler.exe
0-------------------------------------------0
|                                           |
|              CS4000 Compiler              |
|             By: Bryant Nelson             |
|                                           |
0-------------------------------------------0

Please select from the following menu:
        1. Compile Program
        2. Run Program
        3. Exit
        :1

        Enter the file name of your program: 1.prog
        Enter the name of your program: 1

        Compilation Completed Successfully!

Please select from the following menu:
        1. Compile Program
        2. Run Program
        3. Exit
        :2

        Enter the name of your program: 1
        Enter the Input Source, for keyboard leave blank: 1.dat
        Enter the Output Destination, for screen leave blank: 1.out


Program Completed!


Output:
5
```

Program Alterations

- Added a semicolon to the line before ‘end’. My grammar requires it, as noted above.

Program 2

Compilation / Execution

```
E:\Compiler Construction\Compiler Testing\CS 4000 Compiler.exe
0-----------------------------------------0
|                                         |
|              CS4000 Compiler            |
|             By: Bryant Nelson           |
|                                         |
0-----------------------------------------0

Please select from the following menu:
        1. Compile Program
        2. Run Program
        3. Exit
        :1

        Enter the file name of your program: 2.prog
        Enter the name of your program: 2

        Compilation Completed Successfully!

Please select from the following menu:
        1. Compile Program
        2. Run Program
        3. Exit
        :2

        Enter the name of your program: 2
        Enter the Input Source, for keyboard leave blank: 2.dat
        Enter the Output Destination, for screen leave blank: 2.out

Awaiting input: 2
Awaiting input: 7
Awaiting input: 5
Awaiting input: 20

Program Completed!


Output:
3
```

Program & Data Alterations

- Added a semicolon to the line before ‘end’. My grammar requires it, as noted above.
- Changed the data file so that each value was on a different line. As stated above, my semantics for “readln()” expect it this way.

Program 3

Compilation / Execution

```
E:\Compiler Construction\Compiler Testing\CS 4000 Compiler.exe
0-----------------------------------------0
|                                         |
|              CS4000 Compiler            |
|             By: Bryant Nelson           |
|                                         |
0-----------------------------------------0

Please select from the following menu:
        1. Compile Program
        2. Run Program
        3. Exit
        :1

        Enter the file name of your program: 3.prog
        Enter the name of your program: 3

        Syntax Error: Expecting closing structure, found 'od' at line 10 (CODE 0
29)
        Compile Error!
```

Program Alterations

- Added a semicolon to the line before 'end'. My grammar requires it, as noted above.
- Since my grammar expects every line to end with a semicolon, the line that was originally supposed to cause an error will not. However, I removed the semicolon on that line in order to intentionally create an error.

Program 4

Compilation / Execution

```
E:\Compiler Construction\Compiler Testing\CS 4000 Compiler.exe
0-----------------------------------------0
|                                         |
|              CS4000 Compiler            |
|             By: Bryant Nelson           |
|                                         |
0-----------------------------------------0

Please select from the following menu:
        1. Compile Program
        2. Run Program
        3. Exit
        :1

        Enter the file name of your program: 4.prog
        Enter the name of your program: 4

        Compilation Completed Successfully!

Please select from the following menu:
        1. Compile Program
        2. Run Program
        3. Exit
        :2

        Enter the name of your program: 4
        Enter the Input Source, for keyboard leave blank: 4.dat
        Enter the Output Destination, for screen leave blank: 4.out

Awaiting input: 7

Program Completed!


Output:
5040
```

Program Alterations

- Added a semicolon to the line before ‘end’. My grammar requires it, as noted above.
- I also had to add a semicolon to the last line of the while loop.

Program 5

Compilation / Execution

```
E:\Compiler Construction\Compiler Testing\CS 4000 Compiler.exe
O-----------------------------------------O
|                                         |
|              CS4000 Compiler            |
|             By: Bryant Nelson           |
|                                         |
O-----------------------------------------O

Please select from the following menu:
        1. Compile Program
        2. Run Program
        3. Exit
        :1

        Enter the file name of your program: 5.prog
        Enter the name of your program: 5

        Semantic Error: Variable cannot be referenced before it is declared! Lin
e 11 (CODE 055)
        Compile Error!
```

Program Alterations

- Added a semicolon to the line before ‘end’. My grammar requires it, as noted above.
- I also had to add a semicolon to the last line of the while loop.

Program 6

Compilation / Execution

```
E:\Compiler Construction\Compiler Testing\CS 4000 Compiler.exe
O-------------------------------------------O
|                                           |
|              CS4000 Compiler              |
|             By: Bryant Nelson             |
|                                           |
O-------------------------------------------O

Please select from the following menu:
        1. Compile Program
        2. Run Program
        3. Exit
        :1

        Enter the file name of your program: 6.prog
        Enter the name of your program: 6

        Compilation Completed Successfully!

Please select from the following menu:
        1. Compile Program
        2. Run Program
        3. Exit
        :2

        Enter the name of your program: 6
        Enter the Input Source, for keyboard leave blank: 6.dat
        Enter the Output Destination, for screen leave blank: 6.out

Awaiting input: 7
Awaiting input: 4
Awaiting input: 8
Awaiting input: 1
Awaiting input: 3
Awaiting input: 2
Awaiting input: 6
Awaiting input: 9
Awaiting input: 5
Awaiting input: 10

Program Completed!


Output:
7
4
8
1
3
2
6
9
5
10
```

Program Alterations

- Added a semicolon to the line before ‘end’. My grammar requires it, as noted above.
- Added a semicolon to the last line of the while loop, and to the last line of the foreach loop.
- Changed “i := 0;” to “i := 1;” since my indices start at 0.
- Removed “or i = 10” from the while condition since my indices start at 0.

Program 7

Compilation / Execution

```
E:\Compiler Construction\Compiler Testing\CS 4000 Compiler.exe
O-------------------------------------------O
|                                           |
|              CS4000 Compiler              |
|             By: Bryant Nelson             |
|                                           |
O-------------------------------------------O

Please select from the following menu:
        1. Compile Program
        2. Run Program
        3. Exit
        :1

        Enter the file name of your program: 7.prog
        Enter the name of your program: 7

        Compilation Completed Successfully!

Please select from the following menu:
        1. Compile Program
        2. Run Program
        3. Exit
        :2

        Enter the name of your program: 7
        Enter the Input Source, for keyboard leave blank: 7.dat
        Enter the Output Destination, for screen leave blank: 7.out

Awaiting input: 7
Awaiting input: 4
Awaiting input: 8
Awaiting input: 1
Awaiting input: 3
Awaiting input: 2
Awaiting input: 6
Awaiting input: 9
Awaiting input: 5
Awaiting input: 10

Program Completed!


Output:
14
8
16
2
6
4
12
18
10
20
```

Program Alterations

- Added a semicolon to the line before ‘end’. My grammar requires it, as noted above.
- Added a semicolon to the last line of both while loops and the foreach loop.
- Added a semicolon after the foreach loop’s closing ‘od’.
- Changed “i := 0;” to “i := 1;”, in both instances, since my indices start at 0.
- Removed “or i = 10” from the while conditions since my indices start at 0.

Program 8

Compilation / Execution

```
E:\Compiler Construction\Compiler Testing\CS 4000 Compiler.exe
O-------------------------------------------O
|                                           |
|              CS4000 Compiler              |
|             By: Bryant Nelson             |
|                                           |
O-------------------------------------------O

Please select from the following menu:
        1. Compile Program
        2. Run Program
        3. Exit
        :1

        Enter the file name of your program: 8.prog
        Enter the name of your program: 8

        Compilation Completed Successfully!

Please select from the following menu:
        1. Compile Program
        2. Run Program
        3. Exit
        :2

        Enter the name of your program: 8
        Enter the Input Source, for keyboard leave blank: 8.dat
        Enter the Output Destination, for screen leave blank: 8.out

Awaiting input: 7
Awaiting input: 4
Awaiting input: 8
Awaiting input: 1
Awaiting input: 3
Awaiting input: 2
Awaiting input: 6
Awaiting input: 9
Awaiting input: 5
Awaiting input: 10

Program Completed!


Output:
10
9
8
7
6
5
4
3
2
1
```

Program Alterations

- Added a semicolon to the line before ‘end’. My grammar requires it, as noted above.
- Added a semicolon to the last line of all while loops .
- Changed “i := 0;” to “i := 1;”, in all instances, since my indices start at 0, same for ‘j’.
- Removed “or i = 10” from the while conditions since my indices start at 0, same for ‘j’.

Program 9

Compilation / Execution

```
E:\Compiler Construction\Compiler Testing\CS 4000 Compiler.exe
O-------------------------------------------O
|                                           |
|              CS4000 Compiler              |
|             By: Bryant Nelson             |
|                                           |
O-------------------------------------------O

Please select from the following menu:
        1. Compile Program
        2. Run Program
        3. Exit
        :1

        Enter the file name of your program: 9.prog
        Enter the name of your program: 9

        Compilation Completed Successfully!

Please select from the following menu:
        1. Compile Program
        2. Run Program
        3. Exit
        :2

        Enter the name of your program: 9
        Enter the Input Source, for keyboard leave blank: 9.dat
        Enter the Output Destination, for screen leave blank: 9.out

Awaiting input: 1
Awaiting input: 2
Awaiting input: 3
Awaiting input: 4
Awaiting input: 5
Awaiting input: 6
Awaiting input: 7
Awaiting input: 8
Awaiting input: 9
Awaiting input: 10
Awaiting input: 11
Awaiting input: 12
Awaiting input: 13
Awaiting input: 14
Awaiting input: 15
Awaiting input: 16
Awaiting input: 17
Awaiting input: 18
Awaiting input: 19
Awaiting input: 20
Awaiting input: 21
Awaiting input: 22
Awaiting input: 23
Awaiting input: 24
Awaiting input: 25

Program Completed!


Output:
1
4
9
16
25
36
49
64
81
100
121
144
169
196
225
256
289
324
361
400
441
484
529
576
625
```

Program Alterations

- Added a semicolon everywhere that my altered grammar requires.
- Adjusted all indices and index counters to account for my zero indexed arrays.

Program 10

Compilation / Execution

```
E:\Compiler Construction\Compiler Testing\CS 4000 Compiler.exe
O-----------------------------------------O
|                                         |
|              CS4000 Compiler            |
|             By: Bryant Nelson           |
|                                         |
O-----------------------------------------O

Please select from the following menu:
        1. Compile Program
        2. Run Program
        3. Exit
        :1

        Enter the file name of your program: 10.prog
        Enter the name of your program: 10

        Compilation Completed Successfully!

Please select from the following menu:
        1. Compile Program
        2. Run Program
        3. Exit
        :2

        Enter the name of your program: 10
        Enter the Input Source, for keyboard leave blank: 10.dat
        Enter the Output Destination, for screen leave blank: 10.out


Program Completed!


Output:
1
0
0
0
0
0
1
0
0
0
0
0
1
0
0
0
0
0
1
0
0
0
0
0
1
```

Program Alterations

- Added a semicolon everywhere that my altered grammar requires.
- Adjusted all indices and index counters to account for my zero indexed arrays.

Program 11

Compilation / Execution

```
E:\Compiler Construction\Compiler Testing\CS 4000 Compiler.exe
0-----------------------------------------0
|                                         |
|              CS4000 Compiler            |
|             By: Bryant Nelson           |
|                                         |
0-----------------------------------------0

Please select from the following menu:
        1. Compile Program
        2. Run Program
        3. Exit
        :1

        Enter the file name of your program: 11.prog
        Enter the name of your program: 11

        Compilation Completed Successfully!

Please select from the following menu:
        1. Compile Program
        2. Run Program
        3. Exit
        :2

        Enter the name of your program: 11
        Enter the Input Source, for keyboard leave blank: 11.dat
        Enter the Output Destination, for screen leave blank: 11.out


Program Completed!


Output:
10
100
20
100
1
100
```

Program Alterations

- Added a semicolon everywhere that my altered grammar requires.
- Adjusted all indices and index counters to account for my zero indexed arrays.

Altered Programs

```
program 1:

program
m : integer
begin
m:= (40-10)/(9-3);
writeln(m);
end

data 1:

================================
program 2:

program
x1,x2,y1,y2,m : integer
begin
readln(x1,x2,y1,y2);
m:= (y2-y1)/(x2-x1);
writeln(m);
end

data 2:
2
7
5
20

================================
program 3:   should generate an error

program
n,f,i : integer
begin
f:=1;
i:=1;
read(n);
while i < n or i=n do
       f := f*i;
       i := i+1;
od;
write(f);
end

data 3:
7

================================
```

```
program 4:

program
n,f,i : integer
begin
f:=1;
i:=1;
read(n);
while i < n or i=n do
       f := f*i;
       i := i+1;
od;
write(f);
end

data 4:
7

================================
program 5: should generate an error

program
n,f,i : integer
begin
f:=1;
i:=1;
read(n);
while i < n or i=n do
       f := f*i;
       i := i+1;
od;
x:=f;
write(x);
end

data 5:
7

================================
```

```
program 6:
(If your indices begin with zero change the program below to account for
that)

program
n:array[10]
x, i: integer
begin
i:=0;
while i < 10 do
      read(n[i]);
      i:=i+1;
od;

foreach x in n do
       writeln(x);
od;
end

data 6:
7 4 8 1 3 2 6 9 5 10

================================
program 7:
(If your indices begin with zero change the program below to account for
that)

program
n:array[10]
x,i: integer
begin
i:=0;
while i < 10 do
      read(n[i]);
      i:=i+1;
od;

foreach x in n do
       x:=x*2;
od;

i:=0;
while i < 10 do
      writeln(n[i]);
      i:=i+1;
od;
end

data 7:
7 4 8 1 3 2 6 9 5 10

================================
```

program 8:
(If your indices begin with zero change the program below to account for that)

```
program
n:array[10]
i,j,t: integer
begin

i:=0;
while i < 10 do
      read(n[i]);
      i:=i+1;
od;

i:=0;
while i < 10 do
      j:=0;
      while j < 10  do
            if n[i] > n[j] then
                  t:=n[i];
                  n[i]:=n[j];
                  n[j]:=t;
            fi;
            j:=j+1;
      od;
i:=i+1;
od;

i:=0;
while i < 10 do
      writeln(n[i]);
      i:=i+1;
od;
end
```

data 8:
7 4 8 1 3 2 6 9 5 10

==========

program 9:
(If your indices begin with zero change the program below to account for that)

```
program
n:array[5,5]
i,j: integer
begin

i:=0;
while i < 5 do
      j:=0;
      while j < 5 do
            readln(n[i,j]);
            j:=j+1;
      od;
i:=i+1;
od;

i:=0;
while i < 5 do
      j:=0;
      while j < 5 do
            n[i,j] := exp(n[i,j],2);
      j:=j+1;
      od;
i:=i+1;
od;

i:=0;
while i < 5 do
      j:=0;
      while j < 5 do
            writeln(n[i,j]);
            j:=j+1;
      od;
i:=i+1;
od;
end
```

data 9:

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
```

```
16
17
18
19
20
21
22
23
24
25

================================

program 10:
(If your indices begin with zero change the program below to account for
that)

program
n:array[5,5]
i,j: integer
begin

i:=0;
while i < 5 do
      j:=0;
      while j < 5 do
      case i=j do
            n[i,j]:=1;
      :not(j=i) do
            n[i,j]:=0;
      esac;
      j:=j+1;
      od;
i:=i+1;
od;

i:=0;
while i < 5 do
      j:=0;
      while j < 5 do
            writeln(n[i,j]);
            j:=j+1;
      od;
i:=i+1;
od;
end

data 10:


================================
```

```
program 11:
(extra credit)


program
x ,y : integer
begin
x:=1;
y:=100;
with x:integer begin
      x:=10;
      writeln(x,y);
      with x:integer begin
            x:=20;
            writeln(x,y);
      end;
end;
writeln(x,y);
end

data 11:
```

Custom Test Cases

Exponentiation Testing

Compilation / Execution

```
E:\Compiler Construction\Compiler Testing\CS 4000 Compiler.exe
O---------------------------------------------O
|                                             |
|              CS4000 Compiler                |
|             By: Bryant Nelson               |
|                                             |
O---------------------------------------------O

Please select from the following menu:
        1. Compile Program
        2. Run Program
        3. Exit
        :1

        Enter the file name of your program: 12.prog
        Enter the name of your program: 12

        Compilation Completed Successfully!

Please select from the following menu:
        1. Compile Program
        2. Run Program
        3. Exit
        :2

        Enter the name of your program: 12
        Enter the Input Source, for keyboard leave blank:
        Enter the Output Destination, for screen leave blank:

Awaiting input: 2
Awaiting input: 3
Awaiting input: 2
512
512

Program Completed!
```

Comments

This program uses both means of exponentiation to show that they are in fact equal.

Factorial Example

Compilation / Execution

```
E:\Compiler Construction\Compiler Testing\CS 4000 Compiler.exe
0-------------------------------------------0
|                                           |
|              CS4000 Compiler              |
|             By: Bryant Nelson             |
|                                           |
0-------------------------------------------0

Please select from the following menu:
        1. Compile Program
        2. Run Program
        3. Exit
        :1

        Enter the file name of your program: 13.prog
        Enter the name of your program: 13

        Compilation Completed Successfully!

Please select from the following menu:
        1. Compile Program
        2. Run Program
        3. Exit
        :2

        Enter the name of your program: 13
        Enter the Input Source, for keyboard leave blank:
        Enter the Output Destination, for screen leave blank:

Awaiting input: -5
Awaiting input: 6
720
Awaiting input: 9
362880
Awaiting input: 5
120
Awaiting input: 7
5040
Awaiting input: 4
24
Awaiting input: -5

Program Completed!
```

Comments

This program reads in the first input, and uses it as the sentinel value to stop the program. It then continues to read in values and output the factorial of those values until the sentinel value is encountered.

With & ForEach Example

Compilation / Execution

```
E:\Compiler Construction\Compiler Testing\CS 4000 Compiler.exe
0-------------------------------------------0
¦                                           ¦
¦              CS4000 Compiler              ¦
¦             By: Bryant Nelson             ¦
¦                                           ¦
0-------------------------------------------0

Please select from the following menu:
        1. Compile Program
        2. Run Program
        3. Exit
        :1

        Enter the file name of your program: 14.prog
        Enter the name of your program: 14

        Compilation Completed Successfully!

Please select from the following menu:
        1. Compile Program
        2. Run Program
        3. Exit
        :2

        Enter the name of your program: 14
        Enter the Input Source, for keyboard leave blank:
        Enter the Output Destination, for screen leave blank:

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24

Program Completed!
```

Comments

This program firmly illustrates the ability of the “with” statement. As you can see, the variable names are redefined as arrays of different sizes, and even as integers, nested within other “with” bodies. The size and type of each variable has to be tracked throughout both Semantic analysis and Execution.