Compilers I : CS3320 Spring 2019

Programming Assignment 1 : Toy Cool Programs

Instructor : Prof. Ramakrishna Upadrasta
Author : Puneet Mangla (CS17BTECH11029)

Correct Programs :

1. Absoulte difference

Given two numbers find the absolute difference between them.

Cool	MIPS	Correspondance

if (x-y) < 0 then
out_int(y-x)
else
out_int(x-y)
fi
	label1:
	...
	sub $t1 $t1 $t2
	sw $t1 12($a0)
	...
	blt $t1 $t2 label4
	la $a0 bool_const0
	...
		Start of label
		
		Subtracts both numbers
		stores the difference
		
		compare differncewith 0
		prints the abs. Difference
		when it returns from label4

2. Triangle

Given three numbers check whether they form a triangle or not.

Cool	MIPS	Correspondance

if c < a+b then
if a < b+c then
if b < a+c then
out_string("Triangle !\n")
else
out_string("Not triangle !\n")
fi
else
	label2:
	...
	add $t1 $t1 $t2
	...
	blt $t1 $t2 label5
	...
	label5:
	...
	add $t1 $t1 $t2
	...
		Start of label
		
		Add a+b
		
		Compare with a+b with c
		
		Start of next label
		
		Add b+c

out_string("Not triangle !\n")	blt $t1 $t2 label8	Compare b+c with a
	…	
fi	label8:	Start of next label
else	…	
out_string("Not triangle !\n")	add $t1 $t1 $t2	Add a+c
	…	
fi	blt $t1 $t2 label11	Comapre a+c with b
	la $a0 bool_const0	print the output string sfter returning from label11

3. Perpendicular vectors

Check if two vectors are perpendicular or not.

Cool	MIPS	Correspondance
If x1*x2 + y1*y2 = 0 then	label3:	Start of label
out_string("Yes, they are perpendicular vectors !\n")	…	
	mul $t1 $t1 $t2	multiply x coordinates
	…	
else	mul $t1 $t1 $t2	multiply y coordinates
out_string("Oops, they are not perpendicular vectors !\n")	…	
	add $t1 $t1 $t2	add both of them to get dot product
	sw $t1 12($a0)	
fi	…	
	beq $t1 $t2 label6	comapre with 0 and print
	la $a1 bool_const0	prints the ouput string sfter returning from label6

4. Even and Odd number

Find if a given number is even or odd.

Cool	MIPS	Correspondance
if (x - 2*(x/2)) = 0 then	label0:	Start of label
out_string("Even Number\n")	…	
	div $t1 $t1 $t2	Divide the number by 2
else	…	
out_string("Odd Number\n")	mul $t1 $t1 $t2	Multiply the result by 2
	…	
fi	sub $t1 $t1 $t2	Subract the result from

	… beq $t1 $t2 label3 la $a1 bool_const0 …	original number Compare with 0 print the output string after returning from label3

5. Pythogorean Triplet

Check whether a triplet (h,p,b) froms a ptythogorean triplet.

Cool	MIPS	Correspondance

if h*h = p*p + b*b then
out_string("Yes, they are pythogorean triplets !\n")
else
out_string("Oops, they are not pythogorean triplets !\n")
fi
	label2:
	…
	mul $t1 $t1 $t2
	…
	mul $t1 $t1 $t2
	…
	mul $t1 $t1 $t2
	…
	add $t1 $t1 $t2
	…
	beq $t1 $t2 label5
	la $a1 bool_const0
		Start of label
		Square hypotenuse
		Square perpendicular
		Square base
		Add base square and perpendicular square
		Compare with hypotenuse square
		print the output string after returning from label5

Incorrect Programs:

1. Wrong Identifiers (Section 10.1)

Code snippet :

```
let 1x : Int <- in_int() in
    out_int(1x)
```

Error Message :

```
syntax error at or near INT_CONST = 1
syntax error at or near OBJECTID = x
Compilation halted due to lex and parse errors
```

Study :

1x is not a valid identifier in cool. When compiler encounters 1, It assumes that a number should be there but when encounters x throws a parsing error because it is unexpected.

2. Wrong Strings (Section 10.2)

Code snippet :

```
let str : String <- "Hello there ,
                     I am using cool" in
   out_string(str)
```

Error Message :

```
syntax error at or near ERROR = Unterminated string
constant
Compilation halted due to lex and parse errors
```

Study :

According to cool lexical conventions a non-escaped newline character may not appear in a string. But at first line there is a newline which is non-escaped and this is the reason of the error.

3. Wrong Comments (Section 10.3)

Code snippet :

```
let str : String <- "Hello there , I am using cool\n" in -
section 10.3 : line 1
   out_string(str)
```

Error Message :

```
syntax error at or near '-'
Compilation halted due to lex and parse errors
```

Study :

When compiler sees ‘-’ token it assumes that another ‘-’ token should follow to start the comments but throws an error because no other ‘-’ character that is found.

4. Wrong Keywords (Section 10.4)

Code snippet :

```
let class : Int <- 10 in
   out_int(3*class)
```

Error Message :

```
syntax error at or near CLASS
syntax error at or near CLASS
Compilation halted due to lex and parse errors
```

Study :

‘class’ is a reserved keyword in cool for declaring classes. Since it is reserved we can’t use it as a variable name and hence the cool compilers throws the error. It expects a class defination after keyword ‘class’.

5. Wrong whitespaces (Section 10.5)

Code snippet :

```
let xwhitespace: Int <- in_int() in
   out_int(x)
```

Error Message :

```
Undeclared identifier x.
In call of method out_int, type Object of parameter arg
does not conform to declared type Int.
Compilation halted due to lex and parse errors
```

Study :

whitespace should be blank, \n, \r, \t, \v. ‘whitespace’ can’t be used to seperate <x> and <:> . Cool compilers thinks ‘xwhitespace’t to be an identifier and throws an error when it tries to print ‘x’ which is undeclared.