Compilers-1
PA-0

Revanth.R
CS18BTECH11040

Correct COOL Programs

trivial-1.cl :-

- This program takes 2 positive integer inputs (n,i) and finds n(mod i) or n % i (in C).
- It employs recursive method.

```
Main.modulo  :
   addiu    $sp  $sp  -16
       sw    $fp  16($sp)
       sw    $s0  12($sp)
       sw    $ra  8($sp)
   addiu    $fp  $sp  4
   move     $s0  $a0
       lw    $s1  16($fp)
       lw    $a0  20($fp)
       lw    $t1  12($s1)
       lw    $t2  12($a0)
       la     $a0  bool_const1
       ble   $t1  $t2  label7
       la     $a0  bool_const0
```

trivial-2.cl :-

- This program takes a positive integer input to find whether it is odd or even.

- It uses the modulo function to do so.

```
Main.odd :
   addiu    $sp  $sp  -16
       sw   $fp  16($sp)
       sw   $s0  12($sp)
       sw   $ra  8($sp)
   addiu    $fp  $sp  4
   move     $s0  $a0
       lw   $a0  16($fp)
       sw   $a0  0($sp)
   addiu    $sp  $sp  -4
       la   $a0  int_const0
       sw   $a0  0($sp)
   addiu    $sp  $sp  -4
   move     $a0  $s0
       bne  $a0  $zero  label11
       la   $a0  str_const3
       li   $t1  1
       jal  _dispatch_abort
```

trivial-3.cl :-

- This program takes a positive integer input and prints its factorial.
- It is done by recursive method.

```
Main.fact:
   addiu    $sp  $sp  -24
       sw   $fp  24($sp)
       sw   $s0  20($sp)
       sw   $ra  16($sp)
   addiu    $fp  $sp  4
   move     $s0  $a0
       la   $s1  int_const0
```

```
        la     $s2  int_const1
        lw     $a0  24($fp)
        lw     $t1  12($s2)
        lw     $t2  12($a0)
        la     $a0  bool_const1
        blt    $t1  $t2  label6
        la     $a0  bool_const0
```

trivial-4.cl :-

- This program takes a positive integer input and prints its factorial.
- It is done by iterative method.

```
Main.fact:
   addiu    $sp  $sp  -20
      sw    $fp  20($sp)
      sw    $s0  16($sp)
      sw    $ra  12($sp)
   addiu    $fp  $sp  4
   move     $s0  $a0
      la    $s1  int_const0
```

trivial-5.cl :-

- This program takes a positive integer input and prints its corresponding fibonacci number.
- It is done by recursive method.
- A class Or (|| in C) is defined as there is no such thing as ‘or’ in COOL.
- In this program, fib(0) and fib(1) are taken to be 0.

```
Or.or:
  addiu   $sp  $sp  -12
      sw   $fp  12($sp)
      sw   $s0  8($sp)
      sw   $ra  4($sp)
  addiu   $fp  $sp  4
  move    $s0  $a0
      lw   $a0  16($fp)
      lw   $t1  12($a0)
      beqz $t1  label0
      lw   $a0  16($fp)
      b    label1

Main.fib:
  addiu   $sp  $sp  -20
      sw   $fp  20($sp)
      sw   $s0  16($sp)
      sw   $ra  12($sp)
  addiu   $fp  $sp  4
  move    $s0  $a0
      lw   $s1  20($fp)
      la   $t2  int_const0
  move    $t1  $s1
      la   $a0  bool_const1
      beq  $t1  $t2  label8
      la   $a1  bool_const0
      jal  equality_test
```

Nontrivial-1.cl :-

- This program takes 2 positive integer inputs (n,i) and prints n^i.
- It is done by recursive means. In this algorithm, the power i.e ‘i’ is repeated cut into half and multiplied as in the program. It takes approx. O(log i) time complexity.

- The modulo function is also used here.

```
Main.power:
   addiu   $sp $sp -20
      sw   $fp 20($sp)
      sw   $s0 16($sp)
      sw   $ra 12($sp)
   addiu   $fp $sp 4
   move    $s0 $a0
      la   $s1 int_const0
      lw   $s2 20($fp)
      la   $t2 int_const0
   move    $t1 $s2
      la   $a0 bool_const1
      beq  $t1 $t2 label7
      la   $a1 bool_const0
      jal  equality_test
```

Nontrivial-2.cl :-

- This program takes a positive integer input and prints whether it is an armstrong number or not.
- Armstrong number: Sum of cube of digits of number is equal to the number itself. Eg:- 371
- Here modulo function is also used.

```
Main.arm:
   addiu   $sp $sp -32
      sw   $fp 32($sp)
      sw   $s0 28($sp)
      sw   $ra 24($sp)
   addiu   $fp $sp 4
   move    $s0 $a0
      la   $s1 int_const0
      la   $s2 int_const0
```

```
    lw    $s3  32($fp)
```

Commonly seen MIPS instructions

- addiu → Add immediate unsigned (no overflow)
- sw → Store word
- move → move value stored at address to registers
- bne → Branch on not equal
- str_const(I) sections contains all the string literals in our code section.
- Function calls corresponds to using branch instructions in the assembly

Incorrect COOL Programs

miss-1.cl :-

Wrong program as the type identifier must start with capital letter in line 14 (Object).

miss-2.cl :-

Wrong program as single quotes are used instead of double in line 16.

miss-3.cl :-

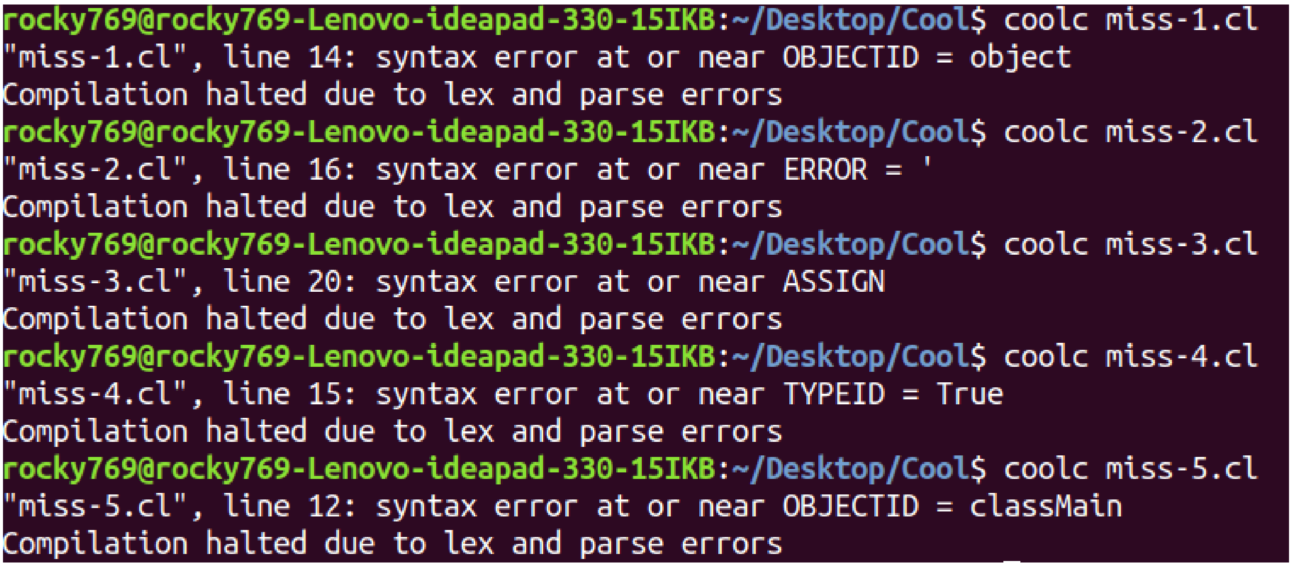
Wrong program as the comment symbol is missing in the last line.

miss-4.cl :-

Wrong program as in line 15, the first letter in true should be small.

miss-5.cl :-

Wrong program as a white space is missing between class and Main.



```
rocky769@rocky769-Lenovo-ideapad-330-15IKB:~/Desktop/Cool$ coolc miss-1.cl
"miss-1.cl", line 14: syntax error at or near OBJECTID = object
Compilation halted due to lex and parse errors
rocky769@rocky769-Lenovo-ideapad-330-15IKB:~/Desktop/Cool$ coolc miss-2.cl
"miss-2.cl", line 16: syntax error at or near ERROR = '
Compilation halted due to lex and parse errors
rocky769@rocky769-Lenovo-ideapad-330-15IKB:~/Desktop/Cool$ coolc miss-3.cl
"miss-3.cl", line 20: syntax error at or near ASSIGN
Compilation halted due to lex and parse errors
rocky769@rocky769-Lenovo-ideapad-330-15IKB:~/Desktop/Cool$ coolc miss-4.cl
"miss-4.cl", line 15: syntax error at or near TYPEID = True
Compilation halted due to lex and parse errors
rocky769@rocky769-Lenovo-ideapad-330-15IKB:~/Desktop/Cool$ coolc miss-5.cl
"miss-5.cl", line 12: syntax error at or near OBJECTID = classMain
Compilation halted due to lex and parse errors
```

NOTE :-

DO NOT GIVE NEGATIVE INTEGER INPUTS TO THESE PROGRAMS AS THEY MAY GIVE WRONG ANSWER.

-----------------THE END-----------------