Name: Nicholas Jordan
Date: 1/16/2015

Milestone Report

Handwritten Answers to Milestone Questions:

Attached at bottom

Specification (what do <u>you</u> think the purpose of this milestone is)

I think a core concept was to teach the expression of programming and mathematical logic in a different form than we might be used to, in this case the postfix notation of gforth. This offered a fairly significant understanding of gforth syntax in general. It also gives a look into how a set of operations in the exercises might be decomposed and parsed into something the machine can read.

Processing (how did <u>you and/or your team</u> go about solving the problem)

Understanding how to translate infix expressions to postfix was a key task, namely which operators take precedence and whether they are left or right associative. With that, it was fairly trivial to generate an expression for gforth input for simpler exercises. With more complex exercises, it took a better reading of gforth tutorials and how the stack functions, i.e. how the flow of control works for an IF-ELSE-THEN, and allowing a variable "function" to be able to call itself recursively.

Testing Requirement (how did <u>you and/or your team</u> test for correctness)

Due to the simplicity of the first few exercises, the mathematical exercises were calculated by hand, and checked with the expected result of the gforth code. In more complex cases, like IF-ELSE-THEN operations, testing was done piecemeal, such as finding out what happens on the stack in each piece. Finally, with the operations factorial and fibanacci, several numbers (mostly in the range of 1 to 10, for simplicity's sake) were inputted, followed through the flow of control on the stack, and confirming the correct results. In all cases, the contents of the stack are printed and removed after every operation.

Retrospective (what did you learn in this milestone)

I learned about the separate data and floating point stacks used in gforth, the seperate mathematical operations required, and the "elevation" of integers when a floating point operand operates with it. I learned how to store data (simple numbers or entire functions) to an assigned name. I learned how an If-then works in gforth. I learned how stack operations like dup, swap, and drop work. Overall, I learned a lot about gforth and it's syntax.

Team Evaluation (what is the percentage of time contributed by each team member)

N/A

Name: Nicholas Jordan
Date: 1/16/2015

N-ary tree data structure:

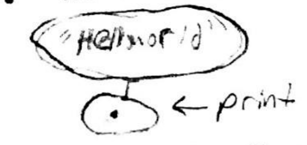
```
Class tree
{
        node root
        int depth, size, etc...?
}

Class node
{
        Bool isleaf
        node parent
        node[] children
        String contents
}

PostOrderTraverse(node)
{
        For each child_node in node.children //left to right
        {
                If child_node isleaf
                {
                        Print child_node contents
                }
                Else
                {
                        PostOrderTraverse(child_node);
                        Print child_node contents
                }
}
```

Usage: call PostOrderTraverse(tree.root) to traverse entire tree, printing node contents in post-order.

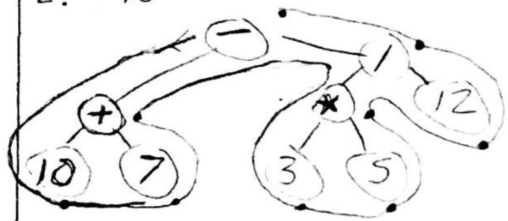
Name: Nicholas Jordan
Date: 1/16/2015

Nicholas Jordan	CS480	Milestone 1

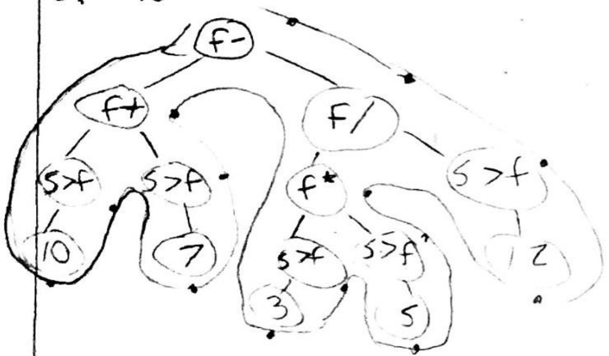
1. ." Hello world"



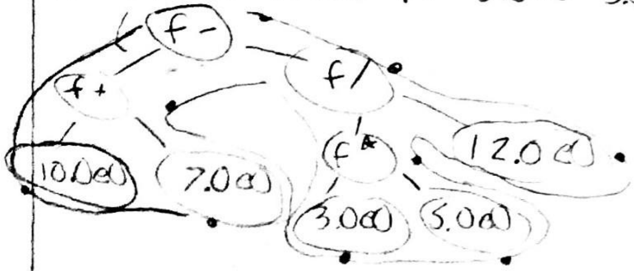
2. 10 7 + 3 5 * 12 / -



3. 10 s>f 7 s>f f+ 3 s>f 5 s>f f* 12 s>f f/ f-

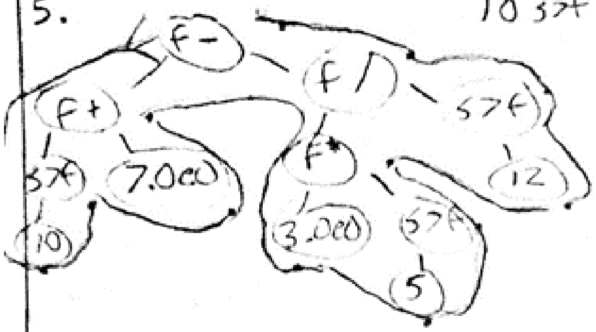


4. 10.0e0 7.0e0 f+ 3.0e0 5.0e0 f* 12.0e0 f/ f-



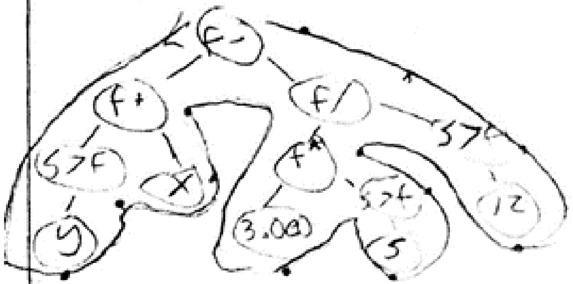
Name: Nicholas Jordan
Date: 1/16/2015

5. 10 s>f 7.0e0 f+ 3.0e0 5 s>f f* 12 s>f f/ f-

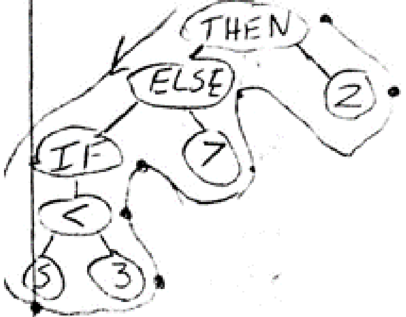


6. : y 10 ;
: x 7.0e0 ;

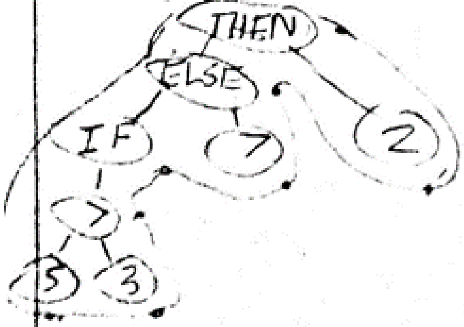
y s>f x f+ 3.0e0 5 s>f f* 12 s>f f/ f-



7. 5 3 < IF 7 ELSE 2 THEN

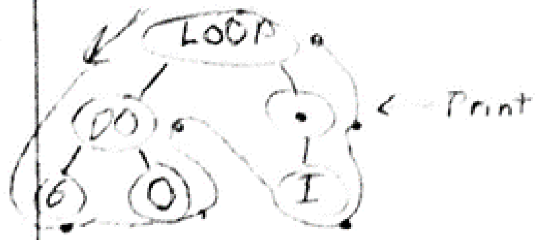


8. 5 3 7 IF 7 ELSE 2 THEN

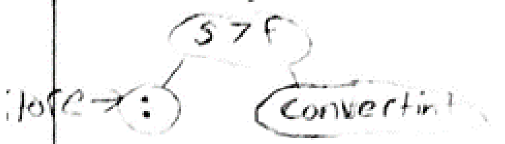


Name: Nicholas Jordan
Date: 1/16/2015

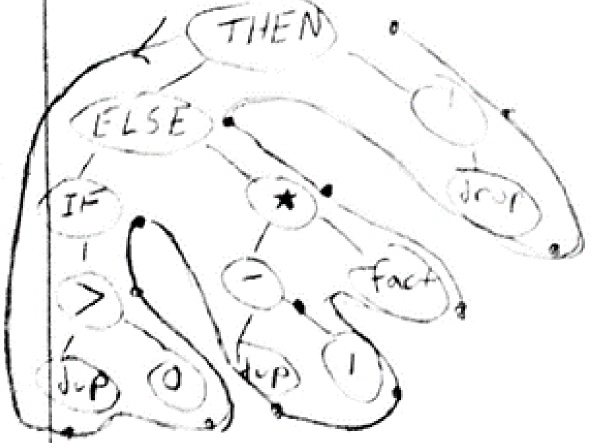
9. 6 0 DO I . LOOP



10. : convertint s>f ;

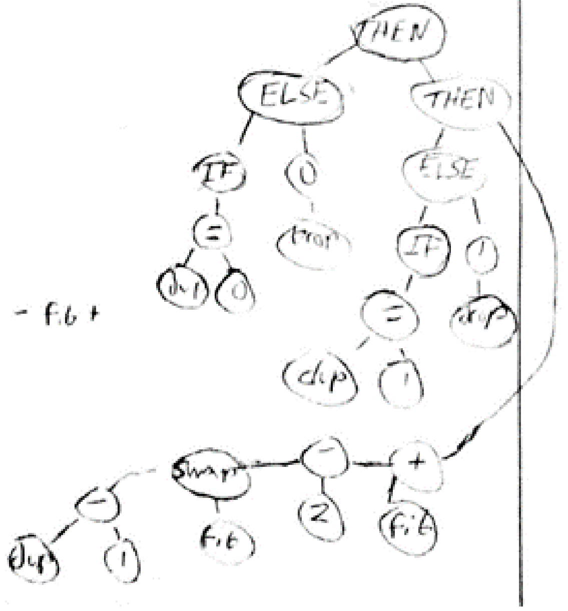


11. : fact recursive dup 0 > IF dup 1 - fact * ELSE drop 1 THEN ;



12. : fib recursive

```
    dup 0 = IF
        drop 0
    ELSE
        dup 1 = IF
            drop 1
        ELSE
            dup 1 - fib swap 2 - fib +
        THEN
    THEN
```



THEN
ELSE
THEN
IF
0
ELSE
=
drop
IF
1
dup
0
=
drop
dup
1
swap
-
+
-
fib
2
fib
dup
1