Caleb Klenda

Homework 2

2/25/2025

# Problem 1

This first model I call the “forever tree” since it looks like a laid down tree cut in half. It leverages the use of brackets to create a branching fractal. The definition is as follows:

* **Alphabet**: {F, X, +, -}
* **Axiom**: "X"
* **Rules**:
  + X → F+[[X]-X]-F[-FX]+X
  + F → FF

“X” operates as a placeholder for the branching expansions. The brackets allow for recursive branching, where:

* “[“pushes the position and angle onto a stack.
* “]” pops the last position and angle, returning to that point.

A screenshot of a computer

AI-generated content may be incorrect.

Running the system with an angle 25°:

A black and white image of a tree

AI-generated content may be incorrect.

With 35°:

A black and white image of a tree

AI-generated content may be incorrect.

With 45°:

A screenshot of a computer screen

AI-generated content may be incorrect.

The shallow angles make the image more tree-like whereas the larger ones makes the image more crystalline.

Bumping back down the angle to 15 and lowering the iterations results in a tree-like structure that is less dense. Changing up the initial angle allows for a different placement of the base stem of the branch.

With initial angle 0°, 6 iterations, 15° turn:

A screen shot of a computer screen

AI-generated content may be incorrect.

With initial angle 15°, 6 iterations, 15° turn:

A screenshot of a computer

AI-generated content may be incorrect.

Other notes from experiment are the complex production rules do not always generate complex and beautiful fractals. It took quite a bit of experimentation to get something branch like without devolving into a chaotic mess.

# Problem 2

2.1)

- The non-terminal symbols are: E, T, & F

- The terminal symbols are: +, -, \*, /, (, ), ident, & number

2.1) Derving id-id-id with G2 (left-most)

**A diagram of a flowchart

AI-generated content may be incorrect.Start with E**  
→ E - T (using E → E - T)

**Expand the leftmost E**  
→ (E - T) - T (using E → E - T)

**Expand the leftmost E**  
→ (T - T) - T (using E → T)

**Expand the leftmost T**  
→ (F - T) - T (using T → F)

**Expand the leftmost F**  
→ (id - T) - T (using F → id)

**Expand the next T**  
→ (id - F) - T (using T → F)

**Expand the next F**  
→ (id - id) - T (using F → id)

**Expand the last T**  
→ (id - id) - F (using T → F)

**Expand the last F**  
→ (id - id) - id (using F → id)

This results in the parse tree on the right:

2.3)

A diagram of a tree

AI-generated content may be incorrect.**Start with E**  
→ E - T (using E → E - T)

**Expand the leftmost E**  
→ (T - T) (using E → T)

**Expand the leftmost T**  
→ (F - T) (using T → F)

**Expand the leftmost F**  
→ (id - T) (using F → id)

**Expand the next T**  
→ (id - T \* F) (using T → T \* F)

**Expand the leftmost T in T \* F**  
→ (id - F \* F) (using T → F)

**Expand the first F in F \* F**  
→ (id - id \* F) (using F → id)

**Expand the last F**  
→ (id - id \* id) (using F → id)

2.4)

For “id-id-id”, the subtraction operator results in the following order of evaluation: Compute id - id first; Then, take the result and subtract id. This doesn’t matter much since subtraction is commutative with itself. However, for “id-id\*id” multiplication and subtraction are not communicative with one another and so the order of operations matters. In this case, G2 will enforce such an order by ensuring that id\*id happens before id – (id\*id).

# Problem 3

3.1) Since it’s ambiguous, multiple trees can exist; a total of 6 are possible.

A diagram of a diagram

AI-generated content may be incorrect.A diagram of a diagram

AI-generated content may be incorrect.

A diagram of a diagram

AI-generated content may be incorrect.A diagram of a diagram

AI-generated content may be incorrect.

A diagram of a diagram

AI-generated content may be incorrect.

A diagram of a flowchart

AI-generated content may be incorrect.

3.2) Only one tree can be constructed since it is fully left associative with the operators. A diagram of a diagram

AI-generated content may be incorrect.

3.3)

**Start with E**  
→ E - T { ( ‘+’ | ‘-‘ ) T } (using E → T { ( ‘+’ | ‘-‘ ) T })

**Expand T**

→ E – { F { ( ‘\*’ | ‘/’ ) F } (using E → F { ( ‘\*’ | ‘/‘ ) F })