AM Language Notation

This document defines the core notation for the AM programming language.

1. Expressions

- Arithmetic: `+ - * / ^`
- Equality: `=`, `≠` (`!=`)
- Inequalities: `< ≤ > ≥`
- Logical: `∧ (and)`, `∨ (or)`, `¬ (not)`

2. Numbers

- Integers: `42`
- Rationals: `1/2`
- Floats: `3.14` (optional, not default)
- Special constants: `π`, `e`, `∞`, `NaN`

ASCII fallback:

- `pi -> π`
- `inf -> ∞`
- `NaN` stays

3. Variables and Let

```
let x = 2
let y = x^2 + 1
```

4. Strings

```
"Hello {name}, the result is {2+3}"
```

5. Case

```
case x of
0 => "zero"
_ => "nonzero"
end
```

6. Algorithms

Define algorithms with `@Name`:

```
@Add(a, b) = a + b
```

Call algorithms:

```
Add(2, 3)
```

Algorithms are first-class:

```
let f = @Add
f(1,2)
```

7. Editor/IDE Guidance

- Auto-replace ASCII -> Unicode (`pi` → `π`, `->` → `→`)
- Toggle view: "Plain ASCII" vs "Math Unicode"
- Goal: readable on paper AND in code.

8. Sample

```
@SafeDiv(a,b) =
case
b ≠ 0 => a / b
b = 0 ∧ a > 0 => ∞
b = 0 ∧ a < 0 => -∞
_ => NaN
end

SafeDiv(1,2) # = 0.5
```

This is the baseline notation. Future: types, structs, access modifiers.