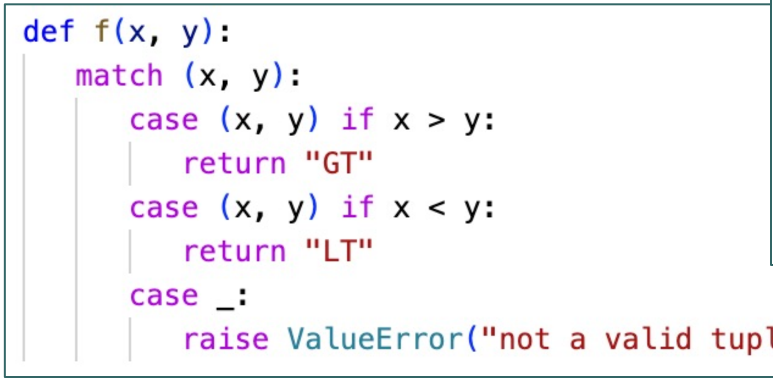
Patterns

- In most modern programming languages patterns are “baked into” the syntax of pattern match statement such as ‘match’ statements/expressions
  - That is, patterns are not standalone structures/values in those languages
- This is true for Asteroid as well
  - But…

Patterns

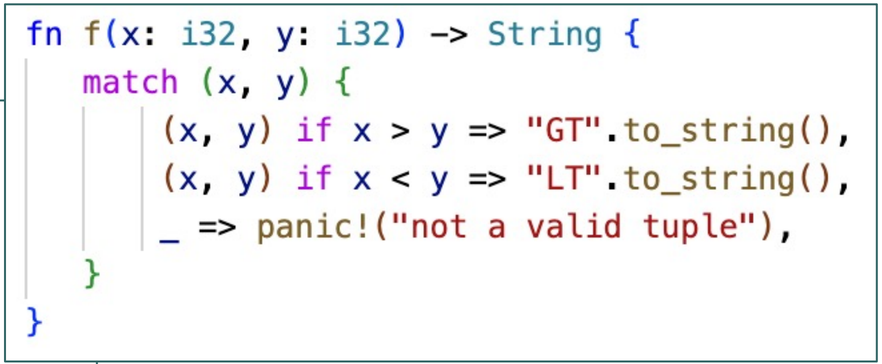
Python



```
def f(x, y):
    match (x, y):
        case (x, y) if x > y:
            return "GT"
        case (x, y) if x < y:
            return "LT"
        case _:
            raise ValueError("not a valid tuple")
```

ln014/match.py

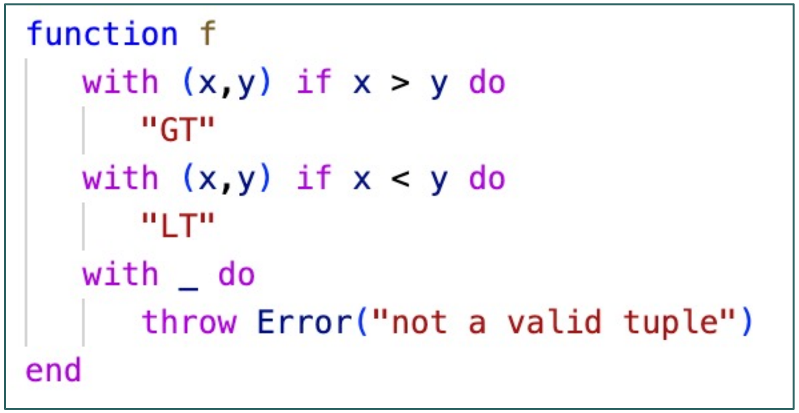
Rust



```
fn f(x: i32, y: i32) -> String {
    match (x, y) {
        (x, y) if x > y => "GT".to_string(),
        (x, y) if x < y => "LT".to_string(),
        _ => panic!("not a valid tuple"),
    }
}
```

ln014/match.rs

Asteroid

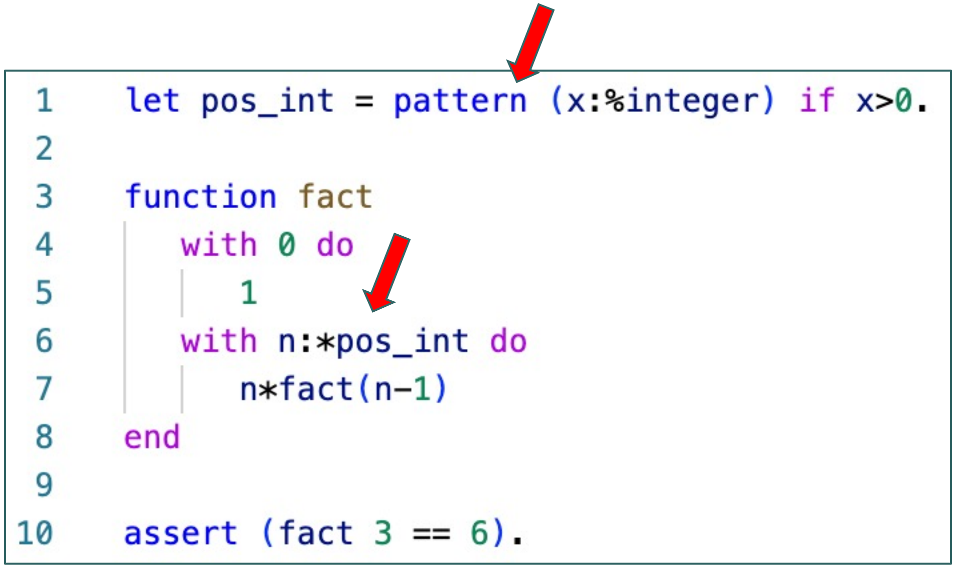


```
function f
    with (x,y) if x > y do
        "GT"
    with (x,y) if x < y do
        "LT"
    with _ do
        throw Error("not a valid tuple")
end
```

ln014/match.ast

First-Class Patterns

- But, Asteroid allows the user to store patterns in variables which can then be dereferenced when needed



```
let pos_int = pattern (x:%integer) if x>0.

function fact
   with 0 do
      1
   with n:*pos_int do
      n*fact(n-1)
end

assert (fact 3 == 6).
```

ln014/int_match2.ast

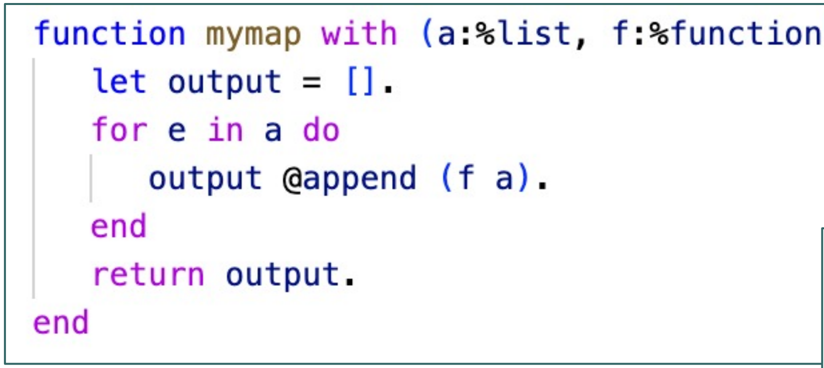
An interesting consequence of first-class patterns is that programs become much more readable.

First-Class Patterns

- Promoting a language feature to first-class status does not increase the computational power of a language (they all are Turing-Complete) but it does **increase its expressiveness** usually perceived as **more readable** programs!

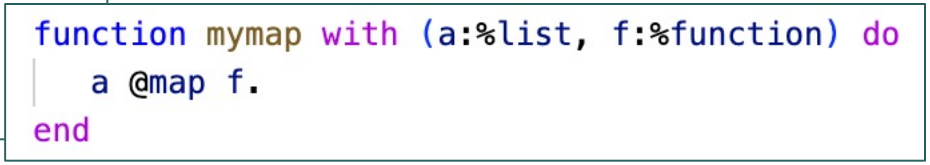
First-Class Features

- We have observed this with functions,
  - Promoting functions to first-class status enables higher-order programming
  - Higher-order programming enables features such as the ‘map’ function



```
function mymap with (a:%list, f:%function) do
    let output = [].
    for e in a do
        output @append (f a).
    end
    return output.
end
```

In014/map1.ast

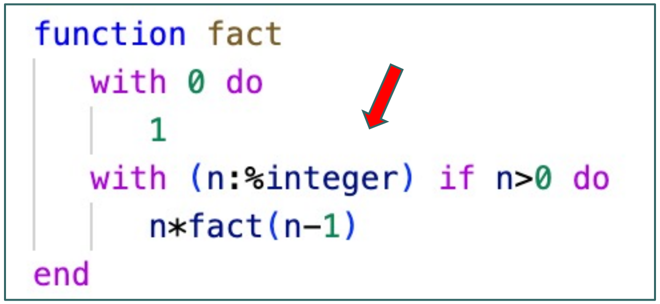


```
function mymap with (a:%list, f:%function) do
    a @map f.
end
```

In014/map2.ast

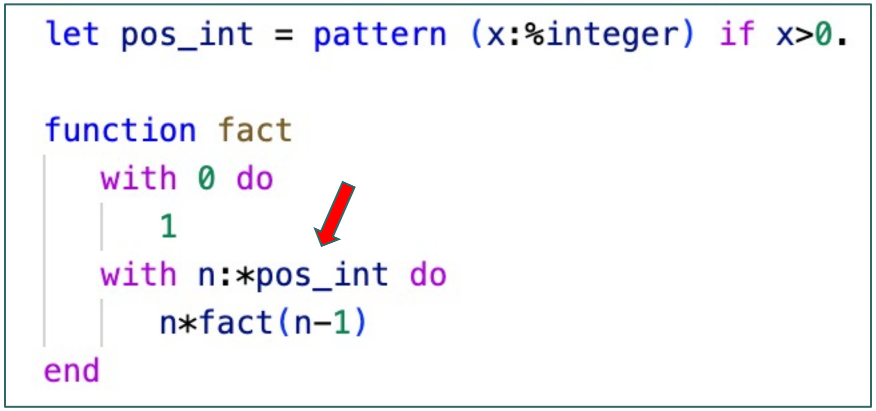
First-Class Patterns

- We can observe the same phenomenon with first-class patterns
  - Programs written with first-class patterns tend to be easier to read and understand



```
function fact
   with 0 do
      1
   with (n:%integer) if n>0 do
      n*fact(n-1)
end
```

In014/int_match1.ast



```
let pos_int = pattern (x:%integer) if x>0.

function fact
   with 0 do
      1
   with n:*pos_int do
      n*fact(n-1)
end
```

In014/int_match2.ast

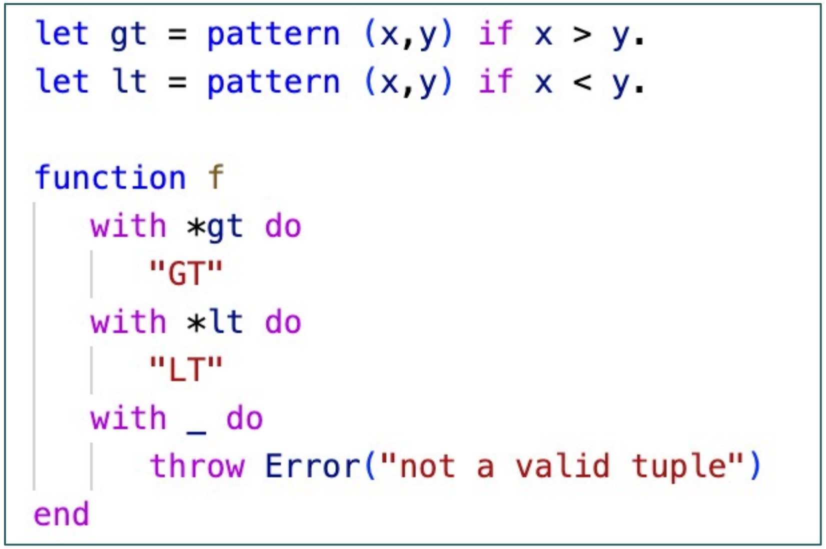
Observation: first-class patterns tend to behave like types – more on that later

First-Class Patterns

- Just like in higher-order programming where any function can be stored in a variable or passed/returned to/from a function…
- …we can do the same with first-class patterns
  - Any pattern can be stored in a variable
  - Any pattern can be passed/returned to/from a function

First-Class Patterns

- Any pattern can be stored in a variable.

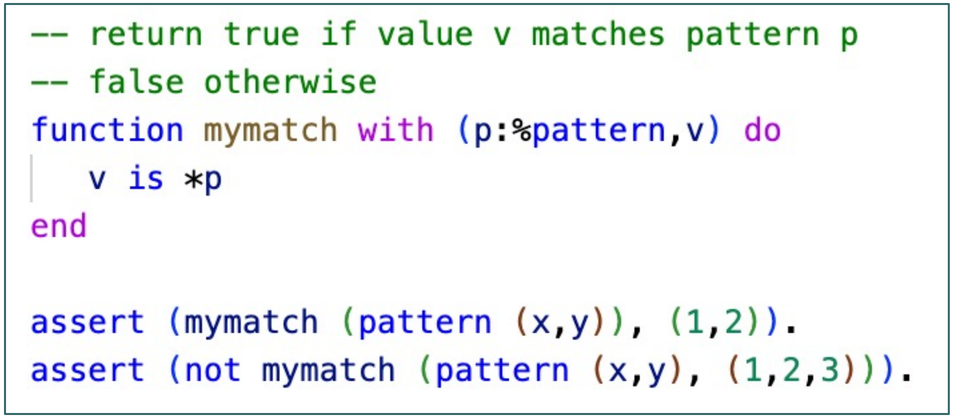


```
let gt = pattern (x,y) if x > y.
let lt = pattern (x,y) if x < y.

function f
    with *gt do
        "GT"
    with *lt do
        "LT"
    with _ do
        throw Error("not a valid tuple")
end
```

First-Class Patterns

- We can pass patterns to functions.

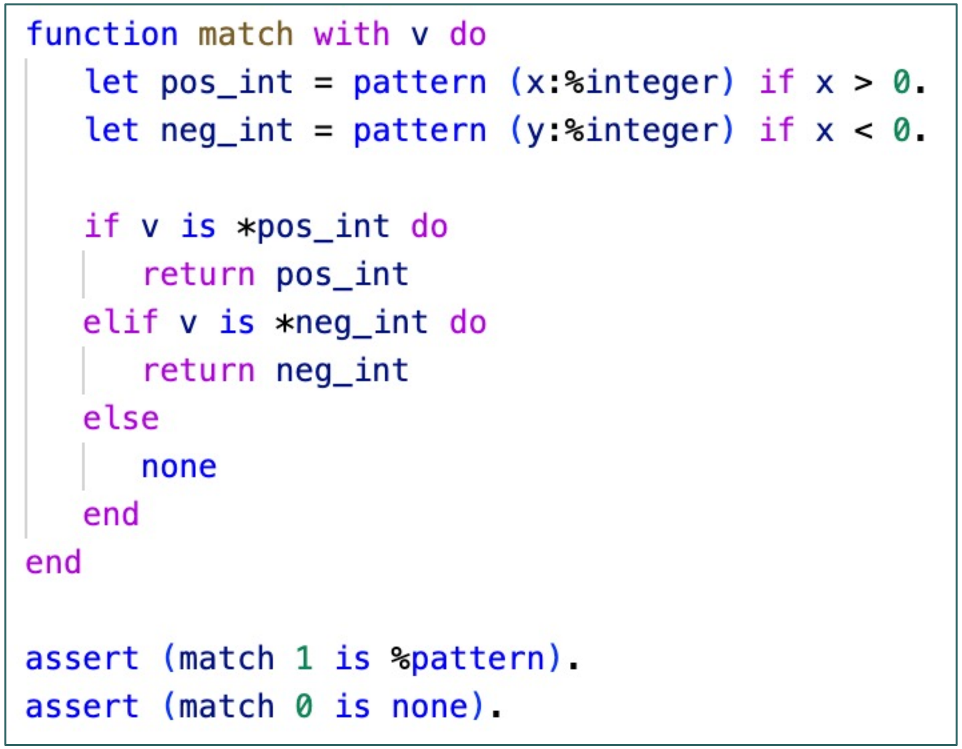


```
-- return true if value v matches pattern p
-- false otherwise
function mymatch with (p:%pattern,v) do
    v is *p
end

assert (mymatch (pattern (x,y)), (1,2)).
assert (not mymatch (pattern (x,y), (1,2,3))).
```

First-Class Patterns

- Returning patterns from functions.



```
function match with v do
    let pos_int = pattern (x:%integer) if x > 0.
    let neg_int = pattern (y:%integer) if x < 0.

    if v is *pos_int do
        return pos_int
    elif v is *neg_int do
        return neg_int
    else
        none
    end
end

assert (match 1 is %pattern).
assert (match 0 is none).
```

Reading

- asteroid-lang.readthedocs.io/en/latest/User%20Guide.html#patterns-as-first-class-citizens