prog_func

- Higher order functions (HOF)
- tips and tricks

Higher order functions (HOF)

function that takes as an argument another function or that returns another function. currying

```
def sum(f: Int => Int)(a: Int, b: Int): Int =
    if (a > b) 0 else f(a) + sum(f)(a + 1, b)
// sum of squares between a and b
sum(x => x * x)(2,3)
// sum of cubes between a and b
sum(x => x * x * x)(2,3)
```

tips and tricks

- use `require` like python `assert`
- there is only one true constructor

```
class Rational(x: Int, y: Int)
    require(y > 0, "denominator must be positive")
    val num = x / gcd(abs(x), y);
    val den = y / gcd(abs(x), y);
    def this(x: Int) = this(x,1)
```

- be careful when overloading that ends with `:`

```
class C(val x: Int) {
def *(that: C): C = new C(this.x * that.x)
def /:(that: C): C = new C(this.x / that.x)
def @:(that: C): C = new C(this.x / that.x)
def &:(that: C): C = new C(this.x / that.x)
override def toString: String = "(" + x.toString + ")"
}
val uno= new C(1) // C = (1)
val due = new C(2) // C = (2)
val tre = new C(3) // C = (3)
val sei = due * tre // C = (6)
val uno2 = due /: tre // C = (1) - in quanto uguale a tre./:(due)
val zero = tre @: due // C = (0) - in quanto uguale a due.@:(tre)
val zero2 = tre &: due // C = (0) - in quanto uguale a due.&:(tre)
```