Lecture 5

- Booleans
- Helper functions
- General recursion
  +
  Strong induction

A natural number

is either

0, or

1 + k, k nat

A boolean is

- true
- false

→ and that's it!

type bool

Values <u>true</u> <u>false</u>

<u>operation</u>

<u>case</u> b <u>of</u> (b : bool)

true => [e_1]

| false => [e_2]

"Helper functions" → function that helps you write some other function

```
fun laughs(n: int): string =        (*
  case n of                             E.g.
     0 => ""                               laughs(4) = "haha"
   | 1 => "a"                           *)
   | _ =>    laughs(n-2) ^ "ha"
```

base cases

```
fun laughs(n:int): string =
      case n of
          0 => ________
        | _ =>    . . . . . .  laughs(n-1) . . . .
```

```
fun laughs(n:int):string =
    case n of
        0 => ""
      | _ => case evenP(n) of
                 true => "h" ^ laughs(n-1)
               | false => "a" ^ laughs(n-1)
```

laughs(4) should be "haha"

laughs(3) should be "aha" — put an h on the front

laughs(2) should be "ha" — put an a on the front

```
fun test_laughs() =
(tests "l1" (laughs 4) "haha";
 tests "l2" (laughs 5) "ahaha")
```

tests → fn helper

"l2" → name

(laughs 5) → string

"ahaha" → string

tests →

s → string

testi → int

testb → bool

```
test_laughs();
run();
```

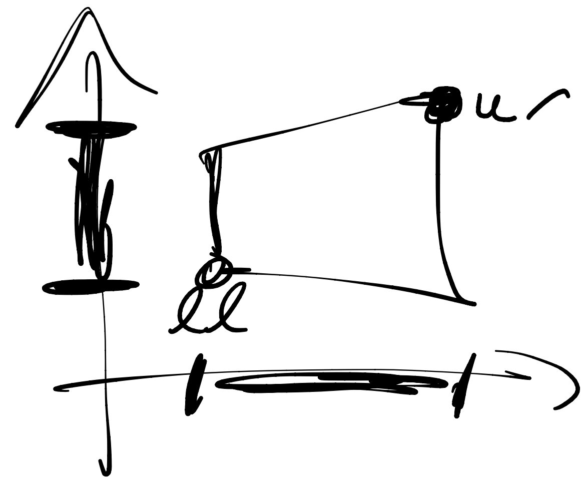
Helper functions

1. ask questions

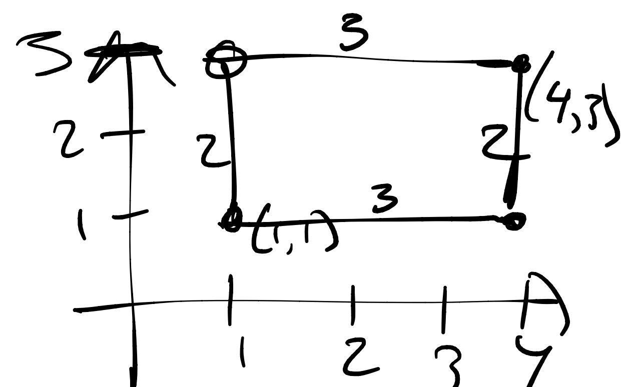
2. avoid repeated code

```
type point = int * int
type rect = point * point

fun area(r:rect):int =
  let
     val ((llx, lly), (urx, ury)) = r
  in
         (urx - llx) * (ury - lly)
  end
```



```
(* Purpose: compute the perimeter of the
   rectangle *)
fun perim(r:rect):int =
  let ((llx, lly), (urx, ury)) = r
  in
     2* (urx-llx) + 2*(ury- lly)
  end
```



```
fun area (r:rect):int =
  let
    val ((llx, lly), (urx, ury)) = r
  in
    (urx-llx) * (ury-lly)
  end
```

```
fun perim (r:rect):int =
  let ((llx, lly), (urx, ury)) = r
  in
    2*(urx-llx) + 2*(ury-lly)
  end
```

```
(* compute the width and height of r *)
fun sides(r: rect): int * int =
    let
        val ((llx, lly), (urx, ury)) = r
    in
            ( urx - llx ,  ury - lly )
               width        height
    end
```

```
fun area (r:rect):int =
  let
    val ( w , h ) = sides(r)
  in
    ( w ) * ( h )
  end
```

```
fun perim(r:rect):int =
  let ( w , h ) = sides(r)
  in
    2*( w ) + 2*( h )
  end
```

Methodology

1. Name + type
2. Purpose
3. Examples
4. Body
   - a. ask a question with a case?
   - b. write a helper
   - c. try a different pattern of recursion
5. Test

① recursion on $n-1$

② recursion on $n-2$

③ recursion on $n-3$

⋮

} general recursion:

define f(n: int) by calling

f(a) for any $a < n$

Euclid's algorithm for

greatest common divisor

d is a common divisor of a and b iff $d \mid a$ and $d \mid b$

"$x \mid y$"

x (nat) divides y (nat) iff

there exists a k (nat) such that

$y = kx$

$3 \mid 6$

$\exists k.\ 6 = k3$ ("exists"; k = 2)

d is the greatest common divisor of a and b

iff

① $d \mid a$ and $d \mid b$ (d is a common divisor of a and b)

and

$\forall d'$ with $d' \mid a$ and $d' \mid b$ any other common divisor

"for all"

$d' \leq d$

$$\gcd(105, 63)$$

$$\mapsto \gcd(63, 42)$$

$$\mapsto \gcd(42, 21)$$

$$\mapsto \gcd(21, 0)$$

$$= 21$$

$$21 \mid 0$$

$$\exists k.\ 0 = k \cdot 21$$

↳ need

$$k = 0 \checkmark$$

$$105 = q\,63 + r$$

↳ quotient (q): 105 div 63

↳ remainder (r): 105 mod 63

$$0 \leq r < 63$$

$$q = 1$$

$$r = 42$$

$$63 = 1 \cdot 42 + 21$$

$$42 = 2 \cdot 21 + 0$$

Assuming $a \geq b$,

```
(* Purpose: gcd(a,b) computes the greatest
            common divisor of a and b *)
fun gcd(a:int, b:int):int =
  case b of
    0 => a
  | _ => gcd(b, a mod b)
```

$a \bmod b < b$

↑ remainder of $\frac{a}{b}$

"Strong Induction":

To prove something for all nats,

1. Prove it for 0
2. for any nonzero k, Prove it for k assuming it for all $l < k$

Theorem: $\gcd(a,b) \mid a$ and $\gcd(a,b) \mid b$

Proof: strong ind on b.

Case for b=0: $\gcd(a,0) \mapsto a$

To show: $a \mid a$ ✓

$a \mid 0$ ✓

Case for $b > 0$:

To show: $\gcd(a, b) \mid a$ (d)

$\gcd(a, b) \mid b$ (d)

want: $d \mid a$ iff $\exists m$ $a = md$

Know

$a = qb + r$

$= q(kd) + ld$

$= d(qk + l)$

$\gcd(a, b)$

$\mapsto \gcd(b, a \bmod b)$ (d)

Inductive hypothesis:

$d \mid b$

$d \mid a \bmod b$

$d \mid b$ means $\exists k.\ b = kd$

$d \mid a \bmod b$ means $\exists l.\ (a \bmod b) = ld$ (r)