SML

```
PolyML.print_depth 5;
infix --;
fun (i -- n) = if (i > n) then [ ] else i::(i + 1 -- n);
fun isTautology([ ]) = true
  | isTautology(x::xs) = x andalso isTautology(xs);
```

```
> val it = (): unit
> infix 0 --
> val -- = fn: int * int -> int list
> # val isTautology = fn: bool list -> bool
```

SML functions

- (i – n) creates a list of number that starts at i and ends at n
- isTautology checks the validity

CS113 LAB 5 - Example 9.1

SML

```
val n = 42; val a = 1.0; val r = 0.42;
fun f(i) = (i * (i + 1.0)) / 2.0;
val lhs = real(foldl op+ 0 (1 -- n));
val rhs = f(real(n));
Real.== (lhs, rhs);
```

```
> val f = fn: real -> real
> val lhs = 903.0: real
> val rhs = 903.0: real
> val it = true: bool
```

- Use the technique of mathematical induction to show that

$$\sum 1 + 2 + 3 \cdots + n = \frac{n(n+1)}{2}, \ n \geq 1$$

SML

```
"Geometric Progression";
val n = 42; val a = 1.0; val r = 0.5;
fun f(i) = a * Math.pow(r, i);
val lhs = foldl op+ 0.0 (map f(map real(0 -- n)));
val rhs = (a * (1.0 - Math.pow(r, real(n) + 1.0))) / (1.0 - r);
Real.== (lhs, rhs);
```

```
> val n = 42: int
val a = 1.0: real
val r = 0.5: real
> val f = fn: real -> real
> val lhs = 2.0: real
> val rhs = 2.0: real
> val it = true: bool
```

- Use induction to show

$$P(n) : \sum_{i=0}^{n} ar^i = \frac{a(1-r^{n+1})}{1-r},\ n \geq 0 \ where\ r \neq 1$$

CS113 LAB 5 - Example 9.3

SML

```
"Arithmetic progression";
val n = 42; val a = 1; val r = 42;
fun f1(i) = Real.fromInt(( a + (i - 1) * r));
fun f2(k) = (real(k) / 2.0) * (2.0 * real(a) + (real(k) - 1.0) * real(r));
val lhs = foldl op+ 0.0 (map f1(a -- n));
val rhs = f2(n);
Real.== (lhs, rhs);
```

```
> val n = 42: int
val a = 1: int
val r = 42: int
> val f1 = fn: int -> real
> val f2 = fn: int -> real
> val lhs = 36204.0: real
> val rhs = 36204.0: real
> val it = true: bool
```

- Use induction to show that

$$P(n) : \sum_{i=1}^{n}(a + (i-1)r) = \frac{n}{2}[2a + (n-1)r],\ n \geq 1$$

CS113 LAB 5 - Example 9.4a

SML

```
val n = 42;
fun f(k) = IntInf.pow(2, k);
val lhs = (0 -- n);
val rhs = map f(0 -- n);
isTautology(map op<(ListPair.zip(lhs, rhs)));
```

```
> val n = 42: int
> val f = fn: int -> int
> val lhs = [0, 1, 2, 3, 4, ...]: int list
> val rhs = [1, 2, 4, 8, 16, ...]: int list
> val it = true: bool
```

- Use induction to prove that

$$n < 2^n \text{ for all non-negative integers } n$$

CS113 LAB 5 - Problem 9.6

SML

```
val n = 1000;
fun f(i) = ((i * (i + 1.0)) / 2.0);
val prove = f(real(n)) - 3.0;
```

```
> val n = 1000: int
> val f = fn: real -> real
> val prove = 500497.0: real
```

- Use the formula

$$1 + 2 + \cdots + n = \frac{n(n+1)}{2}$$

- to find the value of the sum

$$3 + 4 + \cdots + 1,000$$

CS113 LAB 5 - Problem 9.9

SML

```
val n = 42;
fun f(i) = IntInf.pow(4, i) - 1;
fun divides m n = (n mod m = 0);
isTautology(map (divides 3) (map f(1 -- n)));
```

```
> val n = 42: int
> val f = fn: int -> int
> val divides = fn: int -> int -> bool
> val it = true: bool
```

- For each positive integer n

$$\text{let } P(n) \text{ be the proposition } 4^n - 1 \text{ is divisible by } 3$$

CS113 LAB 5 - Problem 9.10

SML

```
val n = 42;
fun f(i) = IntInf.pow(2, 3 * i) - 1;
fun divides m n = n mod m = 0;
isTautology(map (divides 7) (map f(1 -- n)));
```

```
> val n = 42: int
> val f = fn: int -> int
> val divides = fn: int -> int -> bool
> val it = true: bool
```

- For each positive integer n

$$\text{let } P(n) \text{ be the proposition } 2^{3n} - 1 \text{ is divisible by } 7$$

CS113 LAB 5 - Problem 9.13

SML

```
val n = 42;
fun fl(i, k, n) = if (i = 1) then 3::fl(i + 1, k, n)
                  else if (i > n) then [ ]
                  else 7 * k::fl(i + 1, 7 * k, n);
fun fr(i) = 3 * IntInf.pow(7, i - 1);
val lhs = fl(1, 3, n);
val rhs = map fr(1 -- n);
isTautology(map op=(ListPair.zip(lhs, rhs)));
```

```
> val n = 42: int
> # # val fl = fn: int * int * int -> int list
> val fr = fn: int -> int
> val lhs = [3, 21, 147, 1029, 7203, ...]: int list
> val rhs = [3, 21, 147, 1029, 7203, ...]: int list
> val it = true: bool
```

- A sequence $a_1,\ a_2 \ldots$ is defined recursively by $a_1 = 3$ and $a_n = 7a_n - 1$ for $n \geq 2$. Show that

$$a_n = 3 \times 7^{n-1} \text{ for all integers } n \geq 1$$

CS113 LAB 5 - Problem 9.15

SML

```
val n = 42;
fun fl(i, k, n) = if (i = 1) then 2::fl(i + 1, k, n)
                  else if (i > n) then [ ]
                  else 5 * k::fl(i + 1, 5 * k, n);
fun fr(i) = 2 * IntInf.pow(5, i - 1);
val lhs = fl(1, 2, n);
val rhs = map fr(1 -- n);
isTautology(map op=(ListPair.zip(lhs, rhs)));
```

```
> val n = 42: int
> # # val fl = fn: int * int * int -> int list
> val fr = fn: int -> int
> val lhs = [2, 10, 50, 250, 1250, ...]: int list
> val rhs = [2, 10, 50, 250, 1250, ...]: int list
> val it = true: bool
```

- Define the following sequence of numbers: $a_1 = 2$ and for $n \geq 2$, $a_n = 5a_n - 1$.
- Find a formula for

$$a_n$$

SML

```
val n = 42;
fun f(i) = i * 2 - 1;
val lhs = foldl op+ 0 (map f(1 -- n));
val rhs = n * n;
isTautology([op=(lhs, rhs)]);
```

```
> val n = 42: int
> val f = fn: int -> int
> val lhs = 1764: int
> val rhs = 1764: int
> val it = true: bool
```

- Use mathematical induction to show that

$$\text{the sum of the first } n \text{ odd positive integers is equal to } n^2$$