Chapter 1

Getting Started

We use LD(n) for the least natural number greater than 1 that divides n.

$$
\begin{aligned}
&\mathit{divides}\ d\ n = \\
&\quad \mathit{rem}\ n\ d \equiv 0
\end{aligned}
$$

It is useful to define LD in terms of a second function that calculates the least divisor starting from a given threshold k, with $k <= n$.

$$
\begin{aligned}
&\mathit{ld}\ n = \mathit{ldf}\ 2\ n \\
&\mathit{ldf}\ k\ n \mid k\ `\mathit{divides}`\ n = k \\
&\qquad\qquad \mid k^2 > n \qquad = n \\
&\qquad\qquad \mid \mathit{otherwise} \quad = \mathit{ldf}\ (k+1)\ n
\end{aligned}
$$

Exercise 4

If *ldf* used $k^2 \geqslant n$ how would that change the function?

Answer of exercise 4

It wouldn't, because otherwise *divides k n* would have been *True*.

$$
\begin{aligned}
&\mathit{prime0}\ n \\
&\quad \mid n > 1 \qquad = \mathit{error}\ \texttt{"not a positive integer"} \\
&\quad \mid n \equiv 1 \qquad = \mathit{False} \\
&\quad \mid \mathit{otherwise} = \mathit{ld}\ n \equiv n
\end{aligned}
$$

Exercise 6

Can you gather from the definition of *divides* what the type declaration for *rem* would look like?

Answer of exercise 6

$$\mathit{rem} :: \mathit{Integer} \to \mathit{Integer} \to \mathit{Integer}$$

Problem 1.9

Exercise 9

Define a function that given the maximum of a list of integers. Use the predefined function *max*.

Answer of exercise 9

```
maxIn :: [Int] → Int
maxIn []       = error "UNDEFINED: empty list"
maxIn [x]      = x
maxIn (x : xs) = max x (maxIn xs)
```

Exercise 10

Define a function `removeFst` that removes the first occurrence of an integer m from a lst of integers. if m does not occur n the list, the list remains unchanged.

Answer of exercise 10

```
removeFst :: Int → [Int] → [Int]
removeFst _ []                  = []
removeFst m (x : xs) | m ≡ x    = xs
                     | otherwise = x : removeFst m xs
```

Exercise 13

Write a function *count* for counting the number of occurrences of a character in a string.

Answer of exercise 13

```
count :: Char → String → Int
count _ [] = 0
count c (y : ys) | c ≡ y      = 1 + count c ys
                 | otherwise = count c ys
```

Exercise 14

Write a function `blowup` such that *blowup*"*bang*!'' should yield `"baannngggg!!!!!"`

Answer of exercise 14

```
blowup :: String → String
blowup = concat ∘ zipWith replicate [1 . .]
```

The above solution is in point-free form because I'm under the impression that point-free form is what Haskellers aim for by default. Like, to write in

point-free form often is an achievement. Check out the slight difference (the readability in particular) in the following definition:

$$blowup\ chrs = concat\ (zipWith\ replicate\ [1\,..]\ chrs)$$

Problem 1.15

Exercise 15

Write a function

$$srtString :: [String] \to [String]$$

that sorts a list of strings in alphabetical order.

Answer of exercise 15

$$\begin{aligned} &srtString\ [\,] = [\,] \\ &srtString\ xs = srtString\ (remove) \end{aligned}$$

I know I probably could have reversed the definition order to be a little more terse, like this:

Problem 1.17

Write a function $substring :: String \to String \to Bool$ that checks whether *str1* is a substring of *str2*.

The `prefix` function was given in Example 1.16:

$$\begin{aligned} &prefix :: String \to String \to Bool \\ &prefix\ [\,]\ \ \ \ \ \ \ \ \_\ \ \ \ \ \ \ \ = True \\ &prefix\ \_\ \ \ \ \ \ \ \ [\,]\ \ \ \ \ \ \ \ = False \\ &prefix\ (x : xs)\ (y : ys) = (x \equiv y) \wedge prefix\ xs\ ys \end{aligned}$$

The actual definition for `substring` is here:

$$\begin{aligned} &substring :: String \to String \to Bool \\ &substring\ str1\ \ str2@(\_ : restOfStr2) \\ &\quad \mid prefix\ str1\ \ str2 = True \\ &\quad \mid otherwise\ \ \ \ \ \ \ \ = prefix\ str1\ \ restOfStr2 \end{aligned}$$

Problem 1.20

Use *map* to write a function *lengths* that takes a list of lists and returns a list of the corresponding lengths.

$$\begin{aligned} &lengths :: [[a]] \to [Int] \\ &lengths = map\ length \end{aligned}$$

Problem 1.21

Use *map* to write a function *sumLengths* that takes a list of lists and returns the sum of their lengths.

```
sumLengths :: [[a]] → Int
sumLengths  = sum ∘ map length
sumLengths' = sum ∘ lengths


factors :: Integer → [Integer]
factors n | n < 1      = error "argument not positive"
          | n ≡ 1      = []
          | otherwise = p : factors (n `div` p)
              where p = ld n
primes0 :: [Integer]
primes0 = filter prime0 [2..]
ldp :: Integer → Integer
ldp n = ldpf primes1 n
ldpf :: [Integer] → Integer → Integer
ldpf (p : ps) n | rem n p ≡ 0 = p
                | p^2 > n     = n
                | otherwise   = ldpf ps n
primes1 :: [Integer]
primes1 = 2 : filter prime [3..]
prime :: Integer → Bool
prime n | n < 1      = error "not a positive integer"
        | n ≡ 1      = False
        | otherwise = ldp n ≡ n
```

Problem 1.24

What happens when you modify the defining equation of `ldp` to `ldp = ldpf primes1`? Nothing. It's just in point-free form.

Chapter 2

Talking about Mathematical Objects

Making Symbolic Form Explicit

Exercise 22

Statement 1. *Between every two rational numbers there is a third one.*

Can you think of an argument showing that 1 is true?

Answer of exercise 22

a = i/j b = m/n a and b are rationals i,j,m and n are integers suppose c = (2i - 1) / j suppose d = b + c

avg of a + b = ((i*n)(j*m))/(j*n)*2 a + b / 2 is a rational?

prove: $\forall x, y \in \mathbb{Q}\colon \quad \exists z \in \mathbb{Q} \mid x < z < y$

Exercise 23

Give structure trees of the following formulas:

1. $\forall x\Big(A(x) \implies \big(B(x) \implies C(x)\big)\Big)$
2. $\exists x\big(A(x) \land B(x)\big)$
3. $\exists x A(x) \land \exists x B(x)$

[start=1]

Ex. 24 — $\forall x\Big(A(x) \implies \big(B(x) \implies C(x)\big)\Big)$

Answer (Ex. 24) —

$$\forall x\Big(A(x) \implies \big(B(x) \implies C(x)\big)\Big)$$

$$A(x) \implies \big(B(x) \implies C(x)\big)$$

$$A(x) \quad B(x) \implies C(x)$$

$$B(x) \quad C(x)$$

Ex. 25 — $\exists x\big(A(x) \wedge B(x)\big)$

Answer (Ex. 25) —

$$\exists x \mid \big(A(x) \wedge B(x)\big)$$

$$A(x) \wedge B(x)$$

$$A(x) \quad B(x)$$

Ex. 26 — $\exists x A(x) \wedge \exists x B(x)$

Answer (Ex. 26) —

$$\exists x A(x) \wedge \exists x B(x)$$

$$\exists x A(x) \quad \exists x B(x)$$

$$A(x) \quad B(x)$$

Write as formulas with restricted quantifiers:

Ex. 27 — $\exists x \mid \big(\exists y \mid x \in \mathbb{Q} \wedge y \in \mathbb{Q} \wedge x < y\big)$

Answer (Ex. 27) — $\exists x, y \in \mathbb{Q} \mid x < y$

Ex. 28 — $\forall x\big(x \in \mathbb{Q} \implies \exists y \mid y \in \mathbb{Q} \wedge x < y\big)$

Answer (Ex. 28) — $\forall x \in \mathbb{R}\,(\exists y \in \mathbb{R} \mid x < y)$

Ex. 29 — $\forall x\big(x \in \mathbb{Z} \implies \exists m, n\,(m \in \mathbb{N} \wedge n \in \mathbb{N} \wedge x = m - n)\big)$

Answer (Ex. 29) — $\forall x \in \mathbb{Z}\,(\exists m, n \in \mathbb{N} \mid x = m - n)$

Exercise 31

Translate into formulas, taking care to express the intended meaning:

1. The equation $x^2 + 1 = 0$ has a solution.
2. A largest natural number does not exist.
3. The number 13 is prime (use $d \mid n$ for 'd divides n').
4. The number n is prime.
5. There are infinitely many primes.

Answer (Ex. 31.1) —

$$\exists x \mid x = \pm\sqrt{-1}$$

Answer (Ex. 31.2) —

$$\neg\exists m \in \mathbb{N} \mid \forall n \in \mathbb{N}\colon m \geqslant n$$

Answer (Ex. 31) —

$$\nexists n \in \mathbb{N} \mid n > 1 \ \wedge \ n \backslash 13$$

Exercise 32

Translate into formulas [taking care to express the intended meaning]

1. Everyone loved Diana. (Use the expression $L(x, y)$ for: x loved y, and the name d for Diana)
2. Diana loved everyone.
3. Man is mortal. (Use $M(x)$ for 'x is a man', and $M\prime(x)$ for 'x is mortal'.)
4. Some birds do not fly. (Use $B(x)$ for 'x is a bird' and $F(x)$ for 'x can fly'.)

Exercise 33

Translate into formulas, using appropriate expressions for the predicates:

1. Dogs that bark do not bite.
2. All that glitters is not gold.
3. Friends of Diana's friends are her friends.
4. The limit of $\frac{1}{n}$ as n approaches infinity is zero.

Exercise 34

1. Everyone loved Diana except Charles.
2. Every man adores at least two women.
3. No man is married to more than one woman.

Exercise 35

1. The King is not raging.
2. The King is loved by all his subjects. (use $K(x)$ for 'x is a King', and $S(x, y)$ for 'x is a subject of y').

Exercise 36

1. $\exists x \in \mathbb{Q} \mid x^2 = 5$
2. $\forall n \in \mathbb{N}\colon \exists m \in \mathbb{N} \mid n < m$

3. $$\forall n \in \mathbb{N} \nexists d \in \mathbb{N} \mid 1 < d < (2^n + 1) \ \wedge \ d \backslash (2^n + 1)$$

4. $\forall n \in \mathbb{N} \exists m \in \mathbb{N} \mid (n < m \ \wedge \ \forall p \in \mathbb{N} \, (p \leqslant n \ \vee \ m \leqslant p))$
5. $\forall \varepsilon \in \mathbb{R}^{+} \Big(\exists n \in \mathbb{N} \mid \forall m \in \mathbb{N} \big(m \geqslant n \implies (|a - a_m| \leqslant \varepsilon) \big) \Big)$ (a, a_0, a_1 refer to real numbers)

2.1 Abstract Formulas and Concrete Structures

Exercise 37

Consider the following formulas:

Are these formulas true or false in the following contexts?

Exercise 38

In Exercise 37, delete the first quantifier on x in the five formulas. Determine for which values of x the resulting open formulas are satisfied in each of the structures.

2.2 Logical Handling of the Quantifiers

Exercise 39

(the propositional equivalent of this was in Exercise 2.19) Argue that Φ and Ψ are equivalent iff $\Phi \iff \Psi$ is valid.

Exercise 39

For every sentence Φ in exercise 2.36, consider its negation $\neg\Phi$, and produce a more positive equivalent by working the negation symbol through the quantifiers.

Exercise 46

Does it hold that $\neg \exists x \in A \mid \Phi(x)$ is equivalent to $\exists x \notin A \mid \Phi(x)$? If your answer is 'yes', give a proof; if 'no', then you should show this by giving a simple refutation (an example of formulas and structures where the two formulas have different truth values).

Exercise 47

Is $\exists x \notin A \mid \Phi(x)$ equivalent to $\exists x \in A \mid \neg\Phi(x)$? Give a proof if your answer is 'yes', and a refutation otherwise.

Exercise 48

Produce the version of Theorem 2.40 that employs restricted quantification. Argue that your version is correct.

Exercise 50

That the sequence $a_0, a_1, a_2, \cdots \in \mathbb{R}$ converges to a, i.e., that $lim_{n\to\infty} a_n = a$, means that $forall\delta > 0 \colon \exists n \mid \forall m \geqslant n \colon |a - a_M| < \delta$. Give a possible equivalent for the statement that the sequence $a_0, a_1, a_2, \cdots \in \mathbb{R}$ does not converge.

2.3 Quantifiers as Procedures

$$
\begin{aligned}
&\mathit{every}, \mathit{some} :: [a] \to (a \to \mathit{Bool}) \to \mathit{Bool} \\
&\mathit{every}\ \mathit{xs}\ p = \mathit{all}\ p\ \mathit{xs} \\
&\mathit{some}\ \mathit{xs}\ p = \mathit{any}\ p\ \mathit{xs}
\end{aligned}
$$

Exercise 51

Define a function

$$\mathit{unique} :: (a \to \mathit{Bool}) \to [a] \to \mathit{Bool}$$

that gives `True` for `unique p xs` just in case there is exactly one object among `xs` that satisfies `p`.

Answer (Ex. 51) — $\mathit{unique}\ p\ \mathit{xs} = \mathit{length}\ (\mathit{filter}\ p\ \mathit{xs}) \equiv 1$

$$
\begin{aligned}
&\mathit{none} :: [a] \to (a \to \mathit{Bool}) \to \mathit{Bool} \\
&\mathit{none}\ \mathit{xs}\ p = \neg\ (\mathit{some}\ \mathit{xs}\ p) \\
&\mathit{unique}' :: (a \to \mathit{Bool}) \to [a] \to \mathit{Bool} \\
&\mathit{unique}'\ p\ [\,] \quad\ = \mathit{false} \\
&\mathit{unique}'\ p\ (x : \mathit{xs}) \\
&\quad \mid p\ x \qquad\ = \mathit{none}\ \mathit{xs}\ p \\
&\quad \mid \mathit{otherwise} = \mathit{unique}'\ p\ \mathit{xs}
\end{aligned}
$$

Exercise 52

Define a function

$$\mathit{parity} :: [\mathit{Bool}] \to \mathit{Bool}$$

that gives `True` for `parity xs` just in case an even number of the `xs`s equals `True`.

Exercise 53

Define a function

$$evenNR :: (a \to Bool) \to [a] \to Bool$$

that gives `True` for `evenNR p xs` just in case an even number of the `xs`s have property `p`. (Use the `parity` function from the previous exercise.)

Chapter 3

The Use of Logic: Proof

3.1 Proof Style

3.2 Proof Recipes

3.3 Rules for the Connectives

Exercise 2

Apply both implication rules to prove $P \implies R$ from the givens $P \implies Q$, $P \implies (Q \implies R)$.

Answer (Ex. 2) —

•Prove that $P \implies R$ when:

(1)$P \implies Q$

(2)$P \implies (Q \implies R)$

$\Vdash$\{ let us assume that P is true. If we can then also show that R is true, then the implication holds. \}

•Prove that $Q \implies R$ is true when:

(3)P is true

[4]\{ *modus ponens* with (2) and (3) \}
$Q \implies R$

[5]\{ *modus ponens* with (1) and (3) \}
Q

[6]\{ *modus ponens* with [5] and [4] \}
R

Exercise 4

Assume that $n, m \in \mathbb{N}$.
Show: (m is odd $\wedge$ n is odd) $\implies$ $m + n$ is even.

Answer (Ex. 4) —

•Prove that (m is odd $\wedge$ n is odd) $\implies$ $m+n$ is even when:

•Prove that $m+n$ is necessarily even if m and n are both odd.

$-n \in \mathbb{N}$

$-m \in \mathbb{N}$

$\Vdash$\{ If the antecedent isn't true, then the implication is *vacuously true*. So in order to prove the implication let's assume the antecedent is true and then demonstrate that the consequent is true. \}

•Prove that $m+n$ is even when:

(1)m is odd

(2)n is odd

[3]\{ definition of odd-ness \}

$\exists x \in \mathbb{Z} \mid m = 2x+1$

[4]\{ definition of odd-ness \}

$\exists y \in \mathbb{Z} \mid n = 2y+1$

$\Vdash$\{ formalizing the definition of "even" to make it amenable to proof \}

•Prove that $\exists i \in \mathbb{Z} \mid m+n = 2i$

$m+n$

$(2x+1)+(2y+1)$

$2x+2y+2$

$2(x+y+1)$

•Prove that $x+y+1$ is an integer, and is therefore a witness of i (??)

Exercise 5

Show:

1. From $P \iff Q$ it follows that $(P \implies R) \iff (Q \implies R)$,
2. From $P \iff Q$ it follows that $(R \implies P) \iff (R \implies Q)$.

Answer (Ex. 5.1) —

•Prove that $(P \implies R) \iff (Q \implies R)$ when:

$-P \iff Q$

$\Vdash$\{ show that each implies the other \}

•Prove $Q \implies R$ when:

$-P \implies R$

$\Vdash$\{ deduction rule \}

•Prove R when:

$-Q$

$+$\{ *modus ponens* \}

P

$+\{$ *modus ponens* $\}$
R

□

•Prove $P \implies R$ when:
$-Q \implies R$
$\Vdash\{$ deduction rule $\}$
•Prove R when:
$-P$
$+\{$ *modus ponens* $\}$
Q
$+\{$ *modus ponens* $\}$
R

□

Answer (Ex. 5.2) —

•Prove $(R \implies P) \iff (R \implies Q)$ when:
$(1) P \iff Q$
$\Vdash\{$ show that each implies the other to demonstrate equivalence $\}$
•Prove that $R \implies P$ when:
$-R \implies Q$
$\Vdash\{$ deduction rule $\}$
•Prove that P when:
$-R$
$+\{$ *modus ponens* $\}$
Q
$+\{$ *modus ponens* / Elimination Rule $\}$
P
•Prove that $R \implies Q$ when:
$-R \implies P$
$\Vdash\{$ deduction rule $\}$
•Prove Q when:
$-R$
$+\{$ *modus ponens* $\}$
P
$+\{$ *modus ponens* $\}$
Q

Exercise 7

Produce proofs for:

1. Given: $P \implies Q$. To show: $\neg Q \implies \neg P$,
2. Given: $P \iff Q$. To show: $\neg P \iff \neg Q$.

Answer (Ex. 7.1) —

•Prove that $\neg Q \implies \neg P$ when:

(1)$P \implies Q$

⊩{ deduction rule }

•Prove $\neg P$ when:

(2)$\neg Q$

⊩{ assume the opposite and then derive $\perp$ }

•Prove ?? ## Q when:

(3)P

[4]{ *modus ponens* }

Q

[5]{ Q and $\neg Q$ }

$\perp$

$\ldots \neg P$

Answer (Ex. 7.2) —

•Prove that $\neg P \iff \neg Q$ when:

(1)$P \iff Q$

⊩{ division into cases. Show that the antecedent/consequent relationship runs both ways to prove equivalence }

•Prove that $\neg P \implies \neg Q$

⊩{ deduction rule }

•Prove $\neg Q$ provided that:

$-\neg P$

⊩{ assume the opposite and then derive $\perp$ }

•Prove P when:

$-Q$

+{ *modus ponens* with root assumption }

P

contradiction!

$\ldots \neg Q$

•Prove that $\neg Q \implies \neg P$

⊩{ deduction rule }

•Prove $\neg P$ when:

$-\neg Q$

⊩{ assume the opposite and then derive something false }

•Prove Q supposing:

$-P$

+{ *modus ponens* with root assumption }

Q

$\ldots \neg P$

* Exercise 9

Show that from $(P \implies Q) \implies P$ it follows that P.

Hint: Apply Proof by Contradiction. (The implication rules do not suffice for this admittedly exotic example.)

Answer (Ex. 9) —

•Prove P when:

$-(P \implies Q) \implies P$

$\Vdash${ contradiction }

•Prove $\bot$ when:

$-\neg P$

[1]{ the antecedent is false so the implication is vacuously true }

$P \implies Q$

[2]{ *modus ponens* }

P

[3]{ law of the excluded middle; $P \wedge \neg P$ }

$\bot$

Theorem 1. *From $P \vee Q$, $\neg P$ it follows that Q.*

Exercise 11

Assume that A, B, C, and D are statements.

1. From the given $A \implies (B \vee C)$ and $B \implies \neg A$, derive that $A \implies C$.

2. From the given $(A \vee B) \implies (C \vee D)$, $C \implies A$, and $B \implies \neg A$, derive that $B \implies D$.

Answer (Ex. 11.1) —

•Derive $A \implies C$ when:

$-B \implies \neg A$

$-A \implies (B \vee C)$

$\Vdash${ deduction rule }

•Prove C when:

$-A$

+{ *modus ponens* }

$B \vee C$

+{ show that B cannot be true }

•Derive $\bot$ when:

$-B$

+{ *modus ponens* }

$\neg A$

+{ law of the excluded middle; $A \wedge \neg A$ }

$\bot$

$\ldots \neg B$

$+\{$? by example 3.10 $\}$

C

Answer (Ex. 11.2) —

•Derive that $B \implies D$ when:

(1)$A \vee B \implies C \vee D$

(2)$C \implies A$

(3)$B \implies \neg A$

$\Vdash\{$ deduction rule $\}$

•Prove D, assuming:

$-B$

[4]{ A disjunction follows from each of its disjuncts. }

$A \vee B$

[5]{ *modus ponens* }

$C \vee D$

[6]{ *modus ponens* }

$\neg A$

[7]{ *modus tollens* }

$\neg C$

[8]{ ? by 3.10 }

D

Exercise 15

Show that for any $n \in \mathbb{N}$, division of n^2 by 4 gives a remainder 0 or 1.

Answer (Ex. 15) —

•Prove that $\frac{n^2}{4}$ gives 0 or 1 as a remainder when:

$-n \in \mathbb{N}$

$+\{$ simple experimentation $\}$

$9 \equiv 1 \pmod 4$

$+\{$ simple experimentation $\}$

$16 \equiv 0 \pmod 4$

$+\{$ simple experimentation $\}$

$25 \equiv 1 \pmod 4$

$\Vdash\{$ division into cases; every natural number is either even or odd $\}$

•Prove that $\frac{n^2}{4}$ gives 0^* as a remainder when:

$-n$ is even

$+\{$ definition of what it means for a number to be even $\}$

*yes, yes, it's more proper to define the task as aiming for 0 *or* 1, but this makes the *intent* of the goal more clear. Besides, proving one of the conjuncts makes the entire expression true. *boo-yah!*

$\exists k \in \mathbb{Z} \mid n = 2k$
+{ figure out n^2 in terms of k }
•n^2
={ substitute based on observation }
$(2k)^2$
={ ? }
$4k^2$
$\ldots n^2 = 4k^2$
now prove that 4 divides 4kˆ2.
Then prove that 4 nearly divides 4kˆ2 + 4k + 1

Exercise 17

Prove the remaining items of THEOREM 2.10 (p. 45). To prove $\Phi \equiv \Psi$ means [something similar to proving set equivalence].

3.4 Rules for the Quantifiers

Exercise 18

Show, using $\forall$-introduction and Deduction Rule: if from Γ, $P(c)$ it follows that $Q(c)$ (where c satisfies P, but is otherwise "arbitrary"), then from Γ it follows that $\forall x\colon P(x) \implies Q(x)$.

Answer (Ex. 18) —

•Prove $\Gamma \implies \big(\forall x\colon P(x) \implies Q(x)\big)$ provided that:
(1)$\big(P(c) \wedge \Gamma\big) \implies Q(c)$
⊩{ deduction rule; suppose Γ (because it's the antecedent) }
•Prove $\forall x\colon P(x) \implies Q(x)$ when:
(2)Γ
⊩{ deduction rule }
•Prove $Q(x)$ when:
(3)$\exists x \mid P(x)$
[4]{ A conjunction follows from its two conjuncts taken together. }
$\Gamma \wedge P(x)$
[5]{ *modus ponens* }
$Q(x)$
$\ldots \forall x\colon P(x) \implies Q(x)$

3.5 Summary of the Proof Recipes

3.6 Some Strategic Guidelines

Exercise 25

Show:

1. from $\forall x\big(P(x) \implies Q(x)\big)$, $\forall x\colon P(x)$ it follows that $\forall x\colon Q(x)$,
2. from $\exists x\big(P(x) \implies Q(x)\big)$, $\forall x\colon P(x)$ it follows that $\exists x \mid Q(x)$.

Answer (Ex. 25.1) —

•Prove that $\forall x\colon Q(x)$ when:

$-\forall x\colon P(x) \implies Q(x)$

$-\forall x\colon P(x)$

$\Vdash\{$?? $\}$

basically modus ponens but with quantifiers.

3.7 Reasoning and Computation with Primes

Exercise 34

Part I

Let $A = \{4n+3 \mid n \in \mathbb{N}\}$. Show that A contains infinitely many prime numbers.

Hint: any prime ¿ 2 is odd, hence of the form $4n+1$ or $4n+3$. Assume that there are only finietly many primes of the form $4n+3$, say $p_1, \ldots p_m$. Consider the number $N = 4p_1 \ldots p_m - 1 = 4(p_1 \ldots p_m - 1) + 3$. Argue that N must contain a factor $4q+3$, using the fact that $(4a+1)(4b+1)$ is of the form $4c+1$.

Part II

Use `filter prime [ 4*n + 3 | n <- [0..] ]` to generate the primes of this form.

Exercise 36

It is not very difficult to show that if n is composite, $M_n = 2^n - 1$ is composite too. Show this. Hint: Assume that $n = ab$ and probe that $xy = 2^n - 1$ for the numbers $x = 2^b - 1$ and $y = 1 + 2^b + 2^{2b} + \cdots + 2^{(a-1)b}$.

Exercise 38

A slightly faster way to generate the primes is by starting out from the odd numbers. The stepping and marking will work as before, for you count k positions in the odd numbers starting from any odd number $a = 2n + 1$, you will move on o number $(2n+1)+2k$, and if a is a multiile of k, then so is $a+2k$. Implement a function `fasterprimes :: [Integer]` using this idea. The odd natural numbers, staring from 3, can be generated as follows:

$oddsFrom3 :: [Integer]$
$oddsFrom3 = 3 : map\ (+2)\ oddsFrom3$

Exercise 39

Write a Haskell program to refute the following statement about prime numbers: "if $p_1, \ldots, p_k$ are all the primes $< n$, then $(p_1 \times \cdots \times p_k) + 1$ is a prime."

Exercise 41

How would you go about yourself to prove the fact Euclid proved? (if $2^n - 1$ is prime, then $2^{n-1}(2^n - 1)$ is perfect). Here is a hint: if $2^n - 1$ is prime, then the proper divisors of $2^{n-1}(2^n - 1)$ are

$$1, 2, 2^2, \ldots, 2^{n-1}, 2^n - 1, 2(2^n - 1), 2^2(2^n - 1), \ldots, 2^{n-2}(2^n - 1).$$

Exercise 42

Exercise 43

Chapter 4

Sets, Types and Lists

4.1 Let's Talk About Sets

Theorem 2. *For all sets A, B, and C, we have that:*

1. *"reflexivity":* $A \subseteq A$
2. *"antisymmetry":* $A \subseteq B \ \wedge \ B \subseteq A \implies A = B$
3. *"transitivity":* $A \subseteq B \ \wedge \ B \subseteq C \implies A \subseteq C$

Exercise 2

Show that the superset relation also has the properties of Theorem 2. I.e., show that $\supseteq$ is reflexive, antisymmetric and transitive.
Answer (Ex. 2) — $A \subseteq B \equiv B \supseteq A$

Exercise 4

Show that $\big\{\{1,2\},\{0\},\{2,1\}\big\} = \big\{\{0\},\{1,2\}\big\}$.
Answer (Ex. 4) —

- •Prove that $\big\{\{1,2\},\{0\},\{2,1\}\big\} = \big\{\{0\},\{1,2\}\big\}$
- +{ order and repetition in this notation are irrelevant }
  $\{1,2\} \equiv \{2,1\}$
- ⊩{ demonstrating that each is a subset of the other }
  - •Prove that $\big\{\{1,2\},\{0\},\{2,1\}\big\} \subseteq \big\{\{0\},\{1,2\}\big\}$.
  - •Prove that $\big\{\{1,2\},\{0\},\{2,1\}\big\} \supseteq \big\{\{0\},\{1,2\}\big\}$.

* Exercise 7

Assume that A is a set of sets. Show that $\{x \in A \mid x \notin x\} \notin A$.

4.2 Paradoxes, Types and Type Classes

$$\begin{aligned} &\mathit{funny}\ x \mid \mathit{halts}\ x\ x\ &= \bot \\ &\qquad\quad \mid \mathit{otherwise} &= \mathit{True} \end{aligned}$$

So, does `funny funny` diverge or halt? IT'S A PARADOX!!

$$\begin{aligned} &\mathit{halts}\ f\ x = f \not\equiv g \\ &\quad \textbf{where}\ g\ y \mid y \equiv x\ &= \bot \\ &\qquad\qquad\quad \mid \mathit{otherwise} &= f\ y \end{aligned}$$

$$\begin{aligned} &\mathit{collatz} :: \mathit{Integer} \to [\mathit{Integer}] \\ &\mathit{collatz}\ n \mid n < 1\ &= \mathit{error}\ \texttt{"argument not positive"} \\ &\qquad\quad \mid n \equiv 1\ &= [1] \\ &\qquad\quad \mid \mathit{even}\ n &= n : \mathit{collatz}\ (n\ `\mathit{div}`\ 2) \\ &\qquad\quad \mid \mathit{odd}\ n\ &= n : \mathit{collatz}\ (3 * n + 1) \end{aligned}$$

Exercise 8

Explain the following error message:

```
Prelude> elem 1 1
ERROR: [a] is not an instance of class "Num"
Prelude>
```

Answer (Ex. 8) — An intuitive explanation is that you cannot test anything for being a "member" of a number.
The function *elem* has type $\mathit{Eq}\ a \Rightarrow a \to [a] \to \mathit{Bool}$. However, `(elem 1) :: Num a => [a] -> Bool`. This is because when `elem` is partially applied to 1, Haskell determines that `elem` will be operating on some *instance* of the `Num` typeclass[1]. As such, the type changes. The function produced by applying `elem` to `1` can only be sensibly applied to a list of `Num`s. What the function received was *not* a list of some type a such that a was an instance of `Num`, but rather another `1`. I think that the error is, in effect, saying that "my expected parameter is not compatible with what you've supplied me".

4.3 Special Sets

Exercise 10

Show:

1. $\{a\} = \{b\}$ iff $a = b$,

[1] however, since 1 can be an `Int` *or* an `Integer`, it must stick with the more general `Num`

2. $\{a_1, a_2\} = \{b_1, b_2\}$ iff: $a_1 = b_1 \ \wedge \ a_2 = b_2$, or $a_1 = b_2 \ \wedge \ a_2 = b_1$.
   NOTE: these *can* be formally answered!

Answer (Ex. 10.1) — Because $\{a\}$ and $\{b\}$ are both *singletons*, in order to test if the two sets are the same (by Extensionality and such) we only need to compare the element from one to the element from the other.

Exercise 11

Explain that $\varnothing \neq \{\varnothing\}$. And that $\{\varnothing\} \neq \big\{\{\varnothing\}\big\}$.

Answer (Ex. 11) — First, remember the *Principle of Extensionality*: that a set is completely defined by its elements, and as such is equal to another set with the same elements.

I'll start with showing why $\varnothing \neq \{\varnothing\}$. What are the contents of $\varnothing$? Nothing—which can also be seen in the fact that $|\varnothing| = 0$. In contrast, $|\{\varnothing\}| = 1$ because it contains one element: the empty set!

The same reasoning can be applied to the other two, because the first pair happen to be the respective contents of the second pair. Since sets are totally defined by their elements, and we've shown that the elements are not equal, then the respective enclosing sets are also not equal.

4.4 Algebra of Sets

Exercise 13

What are the types of the set difference operator $-$ and of the inclusion operator $\subseteq$?

Answer (Ex. 13) — $\subseteq :: s \to s \to t$
$- :: s \to s \to s$

Exercise 14

Give the types of the following expressions:

1. $x \in \{x \mid E(x)\}$.
2. $\{x \mid E(x)\}$.
3. $(A \cap B) \subseteq C$.
4. $(A \cup B) \cap C$.
5. $\forall x \colon x \in A \implies x \in B$.
6. $A = B$.
7. $a \in A \iff a \in B$.

Answer (Ex. 14) — t, s, t, s, t, t, t

Exercise 17

A, B and C are sets. Show:

1. $A \nsubseteq B \iff A - B \neq \varnothing$.

2. $A \cap B = A - (A - B)$.

Answer (Ex. 17.1) —

•Show that $A \nsubseteq B$ when:

(1)$A - B \neq \varnothing$

•$A - B \neq \varnothing$

$\iff$ { Rewrite the set difference notation (?) }

$\{x \mid x \in A \ \wedge \ x \notin B\} \neq \varnothing$

$\iff$ { Rewrite the ... }

$\exists x \in A \mid x \notin B$

[2]{ $A - B$ can be rewritten as: }

Answer (Ex. 17.2) —

•Prove that $A \cap B = A - (A - B)$

$\Vdash${ Demonstrate that the RHS is a subset of the LHS and vice-versa }

•Prove that $A \cap B \subseteq A - (A - B)$

$\Vdash${ ?? }

•Prove that $x = y$ when:

(1)$\exists x \mid x \in (A \cap B)$

[2]{ definition of subset & intersection of sets }

$\exists x \mid x \in A \ \wedge \ x \in B$

(3)$\exists y \mid y \in (A - (A - B))$

[4]{ ?? }

$\exists y \in A \mid y \notin A - B$

•Prove that $A - (A - B) \subseteq A \cap B$

Exercise 19

Express $(A \cup B) \cap (C \cup D)$ as the union of four intersections.

Answer (Ex. 19) —

•$(A \cup B) \cap (C \cup D)$

$\equiv${ distributivity }

$\big(A \cap (C \cup D)\big) \cup \big(B \cap (C \cup D)\big)$

$\equiv${ distributivity }

$\big((A \cap C) \cup (A \cap D)\big) \cup \big(B \cap (C \cup D)\big)$

$\equiv${ distributivity }

$\big((A \cap C) \cup (A \cap D)\big) \cup \big((B \cap C) \cup (B \cap D)\big)$

$\equiv${ trivial rewrite }

$\bigcup\big((A \cap C), (A \cap D), (B \cap C), (B \cap D)\big)$

Exercise 21

Show that $A \oplus B = (A - B) \cup (B - A) = (A \cup B) - (A \cap B)$.
($A \oplus B$ is the set of all objects that are in A or B but *not both*.)

Answer (Ex. 21) —

•see above (I'm getting lazy)

[1]{ ?? }

$A \oplus B \equiv \{x \,|\, x \in A \oplus x \in B\}$

Exercise 23

Let X be a set with at least two elements. Then by Theorem 2, the relation $\subseteq$ on $\mathscr{P}(X)$ has the properties of reflexivity, antisymmetry, and transitivity. The relation $\leqslant$ on $\mathbb{R}$ also has these properties. The relation $\leqslant$ on $\mathbb{R}$ has the further property of *linearity*: for all $x, y \in \mathbb{R}$, either $x \leqslant y$ or $y \leqslant x$. Show that $\subseteq$ on $\mathscr{P}(X)$ lacks this property.

Exercise 26

Give a logical translation of $\bigcap \mathcal{F} \subseteq \bigcup \mathcal{G}$ using only the relation $\in$.

Exercise 27

Let $\mathcal{F}$ be a family of sets. Show that there is a set A with the following properties:

1. $\mathcal{F} \subseteq \mathscr{P}(A)$.
2. For all sets B: if $\mathcal{F} \subseteq \mathscr{P}(B)$ then $A \subseteq B$.

Exercise 29

fill in and prove Theorem 4.20

Exercise 30

Answer as many of the following questions as you can.

1. Determine: $\mathscr{P}(\varnothing)$, $\mathscr{P}(\mathscr{P}(\varnothing))$, $\mathscr{P}(\mathscr{P}(\mathscr{P}(\varnothing)))$
2. How many elements has $\mathscr{P}^5(\varnothing)$?
3. How many elements has $\mathscr{P}(A)$, given that A as n elements?

Exercise 31

Check whether the following is true: if two sets have the same subsets, then they are equal. I.e.: if $\mathscr{P}(A) = \mathscr{P}(B)$, then $A = B$. Give a proof or a refutation by means of a counterexample.

Exercise 32

Is it true that for all sets A and B:

1. $\mathscr{P}(A \cap B) = \mathscr{P}(A) \cap \mathscr{P}(B)$?
2. $\mathscr{P}(A \cup B) = \mathscr{P}(A) \cup \mathscr{P}(B)$?

Provide either a proof or a refutation by counter-example.

* Exercise 33

Show:

1. $B \cap \left(\bigcup_{i \in I} A_i\right) = \bigcup_{i \in I} (B \cap A_i)$
2. $B \cup \left(\bigcap_{i \in I} A_i\right) = \bigcap_{i \in I} (B \cup A_i)$
3. $\left(\bigcup_{i \in I} A_i\right)^c = \bigcap_{i \in I} A_i^c$, assuming that $\forall i \in I \colon A_i \subseteq X$
4. $\left(\bigcap_{i \in I} A_i\right)^c = \bigcup_{i \in I} A_i^c$, assuming that $\forall \in I \colon A_i \subseteq X$

* Exercise 34

Assume that you are given a certain set A_0. Suppose you are assigned the task of finding sets A_1, A_2, A_3, $\ldots$, such that $\mathscr{P}(A_1) \subseteq A_0$, $\mathscr{P}(A_2) \subseteq A_1$, $\mathscr{P}(A_3) \subseteq A_2$, $\ldots$ Show that no matter how hard you try, you will eventually fail, that is: hit a set A_n for which no A_{n+1} exists such that $\mathscr{P}(A_{n+1}) \subseteq A_n$. (I.e., $\varnothing \notin A_n$.)

Suppose you can go on forever. Show this would entail $\mathscr{P}\left(\bigcap_{i \in \mathbb{N}} A_i\right) \subseteq \bigcap_{i \in \mathbb{N}} A_i$. Apply exercise 4.7.

* Exercise 35

Suppose that the collection $\mathcal{K}$ of sets satisfies the following condition:

$$\forall A \in \mathcal{K} \colon A = \varnothing \ \vee \ \exists B \in \mathcal{K} \mid A = \mathscr{P}(B)$$

Show that every element of $\mathcal{K}$ has the form $\mathscr{P}^n(\varnothing)$ for some $n \in \mathbb{N}$. (N.B.: $\mathscr{P}^0(\varnothing) = \varnothing$)

4.5 Ordered Pairs and Products

Exercise 39

fill out and prove the other items of Theorem 4.38

Exercise 40

1. Assume that A and B are non-empty and that $A \times B = B \times A$. Show that $A = B$.
2. Show by means of an example that the condition of non-emptiness in 1 (the previous part/Question) is necessary. (Did you use this in your proof of 1?)

* Exercise 41

To show that defining (a, b) as $\big\{\{a\}, \{a, b\}\big\}$ works, prove that:

1. $\{a, b\} = \{a, c\} \quad \Longrightarrow \quad b = c.$
2. $\big\{\{a\}, \{a, b\}\big\} = \big\{\{x\}, \{x, y\}\big\} \quad \Longrightarrow \quad a = x \ \wedge \ b = y.$

4.6 Lists and List Operations

Exercise 43

How does it follow from this definition that lists of different length are unequal?

Exercise 44

Another ordering of lists is as follows: shorter lists come before longer ones, and for lists of the same length we compare their first elements, and if these are the same, the remainder lists. Give a formal definition of this ordering. How would you implement it in Haskell?

Exercise 45

Which operation on lists is specified by the Haskell definition in the frame below?

$$
\begin{array}{l}
\mathit{init} :: [a] \to [a] \\
\mathit{init}\ [x] \quad\ = [\,] \\
\mathit{init}\ (x : xs) = x : \mathit{init}\ xs
\end{array}
$$

Answer (Ex. 45) — It's just like Clojure's/STk's `butlast` function.

Exercise 46

Write your own definition of a Haskell operation `reverse` that reverses a list.

Answer (Ex. 46) — $\mathit{reverse} :: [a] \to [a]$

$$
\begin{array}{l}
\mathit{reverse}\ [\,] = [\,] \\
\mathit{reverse}\ l \ = \mathit{squoosh}\ l\ [\,] \\
\quad \textbf{where}\ \mathit{squoosh}\ (x : xs)\ \mathit{ans} = \mathit{squoosh}\ xs\ (x : \mathit{ans}) \\
\qquad\qquad \mathit{squoosh}\ [\,] \qquad \mathit{ans} = \mathit{ans}
\end{array}
$$

Exercise 47

Write a function `splitList` that gives all the ways to split a list of at least two elements in two non-empty parts. The type declaration is:

$$\mathit{splitList} :: [a] \to [([a], [a])]$$

the call `splitList [1..4]` should give `[([1], [2,3,4]),([1,2],[3,4]),([1,2,3],[4])]`.

Answer (Ex. 47) — $\mathit{splitList}\ xs = [\mathit{splitAt}\ i\ xs \mid i \leftarrow [1\,..\,n]]$
$\quad \textbf{where}\ n = (\mathit{length}\ xs) - 1$

4.7 List Comprehension and Database Query

$$
\begin{array}{l}
\textbf{import } DB\ (db) \\
characters = nub\ [x \mid [\texttt{"play"}, \_, \_, x] \leftarrow db] \\
movies \quad\ = \quad\ [x \mid [\texttt{"release"}, x, \_] \leftarrow db] \\
actors \quad\ = nub\ [x \mid [\texttt{"play"}, x, \_, \_] \leftarrow db] \\
directors = nub\ [x \mid [\texttt{"direct"}, x, \_] \leftarrow db] \\
dates \quad\ = nub\ [x \mid [\texttt{"release"}, \_, x] \leftarrow db] \\
universe = nub\ (characters +\!\!+ actors +\!\!+ directors +\!\!+ movies +\!\!+ dates)
\end{array}
$$

Next, define lists of tuples, again by list comprehension:
(I tried to name the variables mnemonically)

$$
\begin{array}{lll}
direct & = [(d, m) & \mid [\texttt{"direct"}, d, m] \leftarrow db] \\
act & = [(a, m) & \mid [\texttt{"play"}, a, m, \_] \leftarrow db] \\
play & = [(a, m, c) & \mid [\texttt{"play"}, a, m, c] \leftarrow db] \\
release & = [(m, y) & \mid [\texttt{"release"}, m, y] \leftarrow db]
\end{array}
$$

Finally, define one placed, two placed and three placed predicates by means of lambda abstraction.

$$
\begin{array}{ll}
characterP & = \lambda x \rightarrow x \in characters \\
actorP & = \lambda x \rightarrow x \in actors \\
movieP & = \lambda x \rightarrow x \in movies \\
directorP & = \lambda x \rightarrow x \in directors \\
dateP & = \lambda x \rightarrow x \in dates \\
actP & = \lambda (x, y) \rightarrow (x, y) \in act \\
releaseP & = \lambda (x, y) \rightarrow (x, y) \in release \\
directP & = \lambda (x, y) \rightarrow (x, y) \in direct \\
playP & = \lambda (x, y, z) \rightarrow (x, y, z) \in play
\end{array}
$$

Translate the following into queries:

Ex. 48 — "Give me the films in which Robert De Niro or Kevin Spacey acted."

Ex. 49 — "Give me all films with Quentin Tarantino as actor or director that appeared in 1994."

Ex. 50 — "Give me all films released after 1997 in which William Hurt did *not* act." (emphasis mine)

4.8 Using Lists to Represent Sets

$$
\begin{array}{l}
delete :: Eq\ a \Rightarrow a \rightarrow [a] \rightarrow [a] \\
delete\ x\ [\,] \quad\ = [\,] \\
delete\ x\ (y : ys)
\end{array}
$$

```
    | x ≡ y      = ys
    | otherwise = y : delete x ys


elem :: Eq a ⇒ a → [a] → Bool
x ∈ []          = False
x ∈ (y : ys)
    | x ≡ y      = True
    | otherwise = elem x ys
union :: Eq a ⇒ [a] → [a] → [a]
[]       `union` ys = ys
(x : xs) `union` ys = x : xs `union` (delete x ys)
intersect :: Eq a ⇒ [a] → [a] → [a]
intersect [] s  = []
intersect (x : xs) s
    | x ∈ s      = x : xs `intersect` s
    | otherwise =     xs `intersect` s
```

Exercise 51

The Haskell operation for list difference is defined as \\ in `List.hs`. Write your own version of this.

Exercise 53

Write functions *genUnion* and *genIntersect* for generalized list union and list intersection. The functions should be of type $[[a]] \to [a]$. They take a list of lists as input and produce a list as output.

Note that *genIntersect* is undefined on the empty list of lists (compare the remark about the presuposition of generalized intersection on page ONE HUNDRED THIRTY-FOUR).

4.9 A Data Type for Sets

Exercise 54

Give implementations of the operations

Exercise 55

In an implementation of sets as lists without duplicates, the implementation of *insertSet* has to be changed. How?

Exercise 56

What would have to change in the module `SetEq.hs` to get a representation of the empty set as 0?

* Exercise 57

1. How many pairs of curly braces {} occur in the expanded notation for $\mathscr{P}^5(\varnothing)$, in the representation where $\varnothing$ appears as {}?
2. How many copies of 0 occur in the expanded notation for $\mathscr{P}^5(\varnothing)$, in the representation where $\varnothing$ appears as 0?
3. How many pairs of curly braces occur in the expanded notation for $\mathscr{P}^5(\varnothing)$, in the representation where $\varnothing$ appears as 0?

Chapter 5

Relations

5.1 The Notion of a Relation

Exercise 13

Show that $\forall x, y \colon \exists R \mid xRy$. ("Between every two things there exist some relation".)

Answer (Ex. 13) — Well, suppose we have some arbitrary x and y—well, there's nothing stopping us from creating an ordered pair (x, y). Now observe that $\{(x, y)\}$ is a valid set. Thus, there is at least one set that contains an ordered pair comprising x and y as parts. Finally, note that $\{(x, y)\}$ satisfies the description of a relation. It's a set of ordered pairs. (The set's being a *singleton* is irrelevant in this circumstance.)

5.2 Properties of Relations

Exercise 17

Show that a relation R on A is irreflexive iff $\Delta_A \cap R = \varnothing$.

Answer (Ex. 17) —

•Show that $\Delta_A \cap R = \varnothing \iff R$ is irreflexive when:

(1)$R \subseteq A^2$

$\Vdash${ Derive each side of the equivalence and prove the other to prove the equivalence itself. }

•Show that $\Delta_A \cap R = \varnothing$ when:

(2)R is irreflexive

[3]{ simple substitution based on definition of *irreflexive* }

$\forall x \in A \colon \neg(x \, R \, x)$

[4]{ reformulation, again based on definition }

$\nexists x \in A \mid (x, x) \in R$

•Show that R is irreflexive when:

(2)$\Delta_A \cap R = \varnothing$

[3]{ ?? }

•$\Delta_A \cap R = \varnothing$

≡{ Substitute by definition }

$\{(x, x) \mid x \in A\} \cap R = \varnothing$

≡{ ?? }

Exercise 19

Show the following:

1. A relation R on a set A is symmetric iff $\forall x, y \in A\colon xRy \iff yRx$.
2. A relation R is symmetric iff $R \subseteq R^{-1}$, iff $R = R^{-1}$.

Exercise 20

Show that every asymmetric relation is irreflexive.

Exercise 22

Show from the definitions that an asymmetric relation always is antisymmetric.

Exercise 23

Show that a relation R on a set A is transitive iff

$$\forall x, z \in A\colon\ (\exists y \in A \mid a\ R\ y\ \wedge\ y\ R\ z) \implies x\ R\ z$$

.

Exercise 28

Show that every strict partial order is asymmetric.

Exercise 29

Show that every relation which is transitive and asymmetric is a strict partial order.

Exercise 30

Show that if R is a strict partial order on A, then $R \cup \Delta_A$ is a partial order on A. (So every strict partial order is contained in a partial order.)

Exercise 31

Show that the inverse of a partial order is again a partial order.

Exercise 32

Let S be a reflexive and symmetric relation on a set A.

A path is a finite sequence $a_1, \ldots, a_n$ of elements of A such that for every i, $1 \leq i < n$, we have that $a_i \; S \; a_{a+1}$. Such a path connects a_1 with a_n.

Assume that for all a, $b \in A$ there is exactly one path connecting a with b.

Fix $r \in A$. Define the relation $\leq$ on A by: $a \leq b$ iff a is one of the elements in the path connecting r with b.

Show the following:

1. $\leq$ is reflexive
2. $\leq$ is antisymmetric
3. $\leq$ is transitive
4. for all $a \in A$, $r \leq a$
5. for every $a \in A$, the set $X_a = \{x \in A \,|\, x \leq a\}$ is finite and if $b, c \in X_a$ then $b \leq c$ or $c \leq b$.
   (A structure $(A, \leq, r)$ with these five properties is called a tree with root r. The directory-structure of a computer account is an example of a tree. Another example is the structure tree of a formula (see Section 2.3 above). These are example of finite trees, but if A is infinite, then the tree $(A, \leq, r)$ will have at least one infinite branch.)

Exercise 33

Consider the following relations on the natural numbers. Check their properties (some answers can be found in the text above). The *successor* relation is the relation given by $\{(n, m) \,|\, n + 1 = m\}$. The divisor relation is $\{(n, m) \,|\, n \text{ divides } m\}$. The *coprime* relation C on $\mathbb{N}$ is given by $nCm :\equiv \text{GCD}(n, m) = 1$, i.e., the only factor of n that divides m is 1, and vice versa.

. . . (table) . . .

Exercise 35

Suppose that R is a relation on A.

1. Show that $R \cup \Delta_A$ is the reflexive closure of R.
2. Show that $R \cup R^{-1}$ is the symmetric closure of R.

Exercise 36

Let R be a transitive binary relation on A. Does it follow from the transitivity of R that its symmetric reflexive closure $R \cup R^{-1} \cup \Delta_A$ is also transitive? Give a proof if your answer is yes, a counterexample otherwise.

Exercise 38

Determine the composition of the relation "father of" with itself. Determine the composition of the relations "brother of" and "parent of" Give an example showing that $R \circ S = S \circ R$ can be false.

Exercise 39

Consider the relation

$$R = \{(0,2),(0,3),(1,0),(1,3),(2,0),(2,3)\}$$

on the set $A = \{0,1,2,3,4\}$.

1. Determine R^2, R^3 and R^4.
2. Give a relation S on A such that $R \cup (S \circ R) = S$.

Exercise 40

Verify:

1. A relation R on A is transitive iff $R \circ R \subseteq R$.
2. Give an example of a transitive relation R for which $R \circ R = R$ is false.

Exercise 41

Verify:

1. $Q \circ (R \circ S) = (Q \circ R) \circ S$. (Thus, the notation $Q \circ R \circ S$ is unambiguous.)
2. $(R \circ S)^{-1} = S^{-1} \circ R^{-1}$.

Exercise 45

Show that the relation $<$ on $\mathbb{N}$ is the transitive closure of the relation $R = \{(n, n+1) \mid n \in \mathbb{N}\}$.

Exercise 46

Let R be a relation on A. Show that $R^+ \cup \Delta_A$ is the reflexive transitive closure of R.

Exercise 47

Give the reflexive transitive closure of the following relation:

$$R = \{(n, n+1) \mid n \in \mathbb{N}\}$$

.

* Exercise 48

1. Show that an intersection of *arbitrarily many* transitive relations is transitive.
2. Suppose that R is a relation on A. Note that A^2 is one example of a transitive relation on A that extends R. Conclude that the intersection of *all* transitive relations extending R is the *least* transative relation extending R. In other words, R^+ equals the intersection of *all* transitive relations extending R.

* Exercise 49

1. Show that $(R^*)^{-1} = (R^{-1})^*$.
2. Show by means of a counter-example that $(R \cup R^{-1})^* = R^* \cup R^{-1*}$ may be false.
3. Prove: if $S \circ R \subseteq R \circ S$, then $(R \circ S)^* \subseteq R^* \circ S^*$.

Exercise 50

Suppose that R and S are reflexive relations on the set A. Then $\Delta_A \subseteq R$ and $\Delta_A \subseteq S$, so $\Delta_A \subseteq R \cap S$, i.e., $R \cap S$ is reflexive as well. We say: reflexivity is preserved under intersection. Similarly, if R and S are reflexive, then $\Delta_A \subseteq R \cup S$, so $R \cup S$ is reflexive. Reflexivity is preserved under union. If R is reflexive, then $\Delta_A \subseteq R^{-1}$, so R^{-1} is reflexive. Reflexivity is preserved under inverse. If R and S are reflexive, $\Delta_A \subseteq R \circ S$, so $R \circ S$ is reflexive. Reflexivity is preserved under composition. Finally, if R on A is reflexive, then the complement of R, i.e., the relation $A^2 - R$, is irreflexive. So reflexivity is not preserved under complement. These closure properties of reflexive relations are listed in the table below.

We can ask the same questions for other relational properties. Suppose R and S are symmetric relations on A. Does it follow that $R \cap S$ is symmetric? That $R \cup S$ is symmetric? That R^{-1} is symmetric? That $A^2 - R$ is symmetric? That $R \circ S$ is symmetric? Similarly for the property of transitivity. These questions are summarized in the table below. Complete the table by putting 'yes' or 'no' in the appropriate places.

. . . (table) . . .