Related reading: Workbook chapters D.1 through D.6.

Exercises:

1. For a positive integer n, use $\sum$ notation to write an expression for the first n terms of the sum
$$2 + 6 + 18 + 54 + 162 + \dots$$
Show that the first n terms of this sum is equal to $3^n - 1$.

2. For what positive integers n is it true that $n! > 2^n$? Use induction to prove that it is true for these values of n.

3. The *Fibonacci sequence* is the sequence $\{F_n\}$ defined by
$$F_0 = 0; \quad F_1 = 1; \quad \forall n \geq 2,\ F_n = F_{n-1} + F_{n-2}.$$
Let $\phi = \frac{1+\sqrt{5}}{2}$. It can be shown that $\phi \approx 1.618$ and that $\phi^2 = \phi + 1$. Use these facts to prove that $F_n < \phi^n$ for all $n \geq 0$.

4. [1] There must be something wrong with the following proof. What is it?

   Theorem 1. *Let a be any positive number. For all integers $n \geq 0$, we have $a^n = 1$.*

   Proof. If $n = 0$, $a^n = a^0 = 1$, so the base case holds. If the theorem is true for all of $0, 1, \dots, n-1, n$ (i.e. $a^0 = 1, a^1 = 1, \dots, a^{n-1} = 1, a^n = 1$), then we have
   $$a^{n+1} = \frac{a^n \cdot a^n}{a^{n-1}} = \frac{1 \cdot 1}{1} = 1,$$
   so the theorem is true for $n+1$ as well. By the principle of strong induction, $a^n = 1$ for all integers $n \geq 0$. □

5. Define a sequence $\{a_n\}$ as follows: $a_1 = 5$, and for $n \geq 1$, $a_n = a_{n-1} + 2n + 1$. Guess a closed form (i.e. not recursive) expression for a_n. Use induction to prove the guess is correct.

Additional exercises:

6. Show that
$$\sum_{k=1}^{n} 3 \cdot 4^{k-1} = 4^n - 1.$$
Compare this equation to Exercise 1 above, and Example D.2.4 (p.96 of workbook), and suggest a generalisation. Prove this generalisation is true. Use this to factorise the polynomial $x^n - 1$.

7. Prove that the sum of the cubes of the first n positive integers is equal to the square of the sum of the first n positive integers. That is, for all integers $n \geq 1$, prove that
$$1^3 + 2^3 + 3^3 + \dots + n^3 = (1 + 2 + \dots + n)^2.$$
(Hint: Use the result from Example D.2.1)

[1] This question is from Donald Knuth's The Art of Computer Programming (3rd ed.), §1.2.1, Q.2

Challenge exercises: (i.e., if you can do any of these questions, then you're going *way* above the course expectations. But hey, maybe you like this stuff, or maybe you want to see some applications to computer science.)

C1. The following algorithm takes a non-negative integer n as a parameter. Prove that the algorithm returns the value of n^2. (Related reading: Epp, 4th ed. §5.5)

```
1: function SQUARE(n)
Require: n ∈ ℤ, n ≥ 0.
Ensure: Return value is n².
2:     m ← 0
3:     k ← 0
4:     while k < n do
5:         m ← m + 2k + 1
6:         k ← k + 1
7:     end while
8:     return m
9: end function
```

C2. Let d be a positive integer. For any integer $n \geq 0$, prove that there exist integers q, r with $0 \leq r < d$ such that $n = dq + r$.

Hint: Use the well-ordering principle in a proof by contradiction.

Remark: This is a part of a familiar theorem, which is stated on p.58 of the workbook.

C3. Prove the result in Exercise C2 also holds for negative n. (Hint: if n is a negative integer, then $-n$ is a positive integer, and we've already proved it for the positive integers.) Why couldn't we easily use the well-ordering principle to prove it for all integers n, instead of just the non-negative ones? There's one more part of the theorem on p.58 we haven't yet proved, what is it? Can you prove it?

C4. (Related reading: Workbook D.7) We can define a set of arithmetic expressions over the real numbers as follows:

I. BASE: Each real number r is an arithmetic expression.

II. RECURSION: If u and v are arithmetic expressions, then $(u + v)$ and $(u - v)$ are arithmetic expressions.

III. RESTRICTION: There are no arithmetic expressions over the real numbers other than those obtained from I and II.

Design and implement a computer program which takes as input an arithmetic expression as defined above, and computes the result of applying the additions and subtractions. Extend your program to also detect for erroneous input (i.e. input which is not an arithmetic expression).[2]

[2] If you're interested in this sort of thing, consider taking COMP3506 in the future.