Problem 1

(a) $\sum_{i=1}^{n} i(i + 1) = n(n+1)(n+2)/3$.

Proof by induction. The base case is $n = 0$, for which both sides evaluate to 0. For the induction step from $n$ to $n+1$, assume that the claim has already been proved for $n$. We can now write

$$\sum_{i=1}^{n+1} i(i + 1) = \sum_{i=1}^{n} i(i + 1) + (n + 1)(n + 2)$$

$$= n(n+1)(n+2)/3 + (n + 1)(n + 2)$$

$$= n(n+1)(n+2) + 3(n+1)(n+2)$$

$$= (n+3)(n+1)(n+2)/3.$$ 

The crucial step is the one marked IH: here, we used the induction hypothesis for the sum of the first $n$ terms. The rest is simple arithmetic.

(b) $\sum_{i=1}^{n} ia^i = \frac{na^{n+2}-(n+1)a^{n+1}+a}{(a-1)^2}$.

Proof by induction. The base case is $n = 0$, for which both sides evaluate to 0. For the induction step from $n$ to $n+1$, assume that the claim has already been proved for $n$. We can now write

$$\sum_{i=1}^{n+1} ia^i = \sum_{i=1}^{n} ia^i + (n + 1)a^{n+1}$$

$$= \frac{na^{n+2}-(n+1)a^{n+1}+a}{(a-1)^2} + (n + 1)a^{n+1}$$

$$= \frac{na^{n+2}-(n+1)a^{n+1}+a + (n+1)a^{n+1}(a^2-2a+1)}{(a-1)^2}$$

$$= \frac{(n+1)a^{n+3}-(2n+2-n)a^{n+2}+a}{(a-1)^2}$$

$$= \frac{(n+1)a^{n+3}+(n+2)a^{n+2}+a}{(a-1)^2}$$

The crucial step is again the one marked IH: here, we used the induction hypothesis for the sum of the first $n$ terms. After that, we simply cancel terms after multiplying out.

Problem 2

Prove that for each $n$, there is a graph $G$ with $n$ nodes that has at least $(n-1)!$ different spanning trees.

Here, we use induction on $n$ to show that the complete graph with $n$ vertices has at least $(n-1)!$ different spanning trees. The complete graph has edges between every pair of nodes.

The base case is $n = 1$. A graph with one node has no edges, and one spanning tree, namely the node itself. Since $(1-1)! = 0! = 1$, the base case holds.

For the induction step from $n$ to $n+1$, we assume that the claim has already been shown for the complete graph of $n$ nodes. Now we consider the complete graph with $n+1$ nodes. It consists of a complete graph
with \( n \) nodes — call it \( G \) — and one more node — call it \( u \) — which is connected to all nodes of \( G \). By the induction hypothesis, \( G \) as the complete graph on \( n \) nodes contains at least \((n-1)!\) spanning trees. For each such spanning tree — call it \( T \) —, there are \( n \) ways of connecting \( u \) to a node of \( T \) and obtain a new spanning tree. All spanning trees obtained this way are different from each other. For each of the \((n-1)!\) spanning trees of \( G \), we get \( n \) different ones, for a total of \( n! \), completing the proof by induction.

(In case you are curious: \((n-1)!\) is really an underestimate, because we could leave out some edges of \( G \), and connect \( u \) to two or more nodes of \( G \), to obtain yet more different spanning trees.)

Problem 3

Prove formally that for each sequence of breaks, it will take exactly \( mn - 1 \) breaks.

Let \( k \) be a counter over the number of breaks performed. By induction on \( k \), we will prove that never mind what the exact breaks are, after \( k \) breaks, there are exactly \( k + 1 \) pieces of chocolate. When the chocolate bar is completely broken into \( mn \) pieces, therefore, exactly \( mn - 1 \) breaks must have happened.

The base case is \( k = 0 \). After 0 breaks, there is exactly the one original entire chocolate bar. For the induction step from \( k \) to \( k + 1 \), assume (by induction) that we have a sequence of breaks, and after \( k \) breaks, there are exactly \( k + 1 \) pieces. The next break, by the rules of the game, picks up one current piece, and breaks it into two new pieces, without altering the other \( k \) pieces. Thus, the number of pieces increases by exactly 1, and there are \( k + 2 \) pieces afterwards. This completes the inductive proof.

Problem 4

Prove formally that the following program correctly computes the sum of the first \( n \) numbers in the array \( a \).

```plaintext
sum = 0;
for (i = 0; i < n; i++)
    sum = sum + a[i];
```

The key insight is that right before the iteration with value \( i \), \( \text{sum} \) contains the sum of the first \( i \) numbers, i.e., \( \sum_{j=0}^{i-1} a[j] \). We will prove this by induction on \( i \).

For the base case \( i = 0 \), right before the iteration of the loop with \( i = 0 \), by the initialization, we have \( \text{sum} = \sum_{j=0}^{i-1} a[j] \). For the induction step, consider iteration \( i + 1 \). By induction hypothesis, right before the program entered the loop for the \( i \)th time, the value was \( \text{sum} = \sum_{j=0}^{i-1} a[j] \). In the \( i \)th iteration, the value \( a[i] \) was added to \( \text{sum} \), so before entering iteration \( i + 1 \), the value is exactly \( \text{sum} = \sum_{j=0}^{i-1} a[j] + a[i] = \sum_{j=0}^{i} a[j] \). This completes the inductive proof.

Thus, when the loop terminates with value \( i = n \), the value of \( \text{sum} \) is \( \text{sum} = \sum_{j=0}^{n-1} a[j] + a[i] \), i.e., the sum of all array entries.