1. Find exact closed-form solutions to the following recurrences in two ways (by unwinding and by homogenization), and verify the result by induction.

   (a) $f(n) = 4f(n-1) + 9$, $f(0) = 1$.

   Solution:

   - Unwinding: We try to guess a solution by unwinding the formula for $f(n)$:

   $$\begin{aligned} f(n) &= 9 + 4f(n-1) \\ &= 9 + 4 \cdot 9 + 4^2 \cdot f(n-2) \\ &= 9 + 4 \cdot 9 + 4^2 \cdot 9 + 4^3 \cdot f(n-3) \end{aligned}$$

   Continuing in this manner suggests the guess

   $$f(n) = 9 + 4 \cdot 9 + 4^2 \cdot 9 + \cdots + 4^{n-1} \cdot 9 + 4^n \cdot f(0).$$

   Since $f(0) = 1$, we can write

   $$\begin{aligned} f(n) &= 9 + 4 \cdot 9 + 4^2 \cdot 9 + \cdots + 4^{n-1} \cdot 9 + 4^n \\ &= 9 \cdot \frac{4^n - 1}{4 - 1} + 4^n \\ &= 3(4^n - 1) + 4^n = 4^{n+1} - 3. \end{aligned}$$

   - Homogenization: By adding 3 to both side, we get

   $$f(n) + 3 = 4f(n-1) + 12 = 4(f(n-1) + 3).$$

   The function $g(n) = f(n) + 3$ then satisfies the recurrence $g(n) = 4g(n-1)$ with initial condition $g(0) = 4$. This gives us $g(n) = 4^{n+1}$. Then $f(n) = g(n) - 3 = 4^{n+1} - 3$.

   - Verification by induction: We prove our guess is correct by induction on n. When $n = 0$, both $f(0)$ and $4^{n+1} - 3$ are 1. Now assume $f(n) = 4^{n+1} - 3$ for some n. Then

   $$f(n+1) = 4f(n) + 9 = 4(4^{n+1} - 3) + 9 = 4^{n+2} - 12 + 9 = 4^{n+2} - 3$$

   as it should be.

   (b) $f(n) = \frac{3}{5}f(n-1) + \frac{4}{5}$, $f(0) = 0$.

   Solution:

   - Unwinding: We try to guess a solution by unwinding the formula for $f(n)$:

   $$\begin{aligned} f(n) &= \frac{4}{5} + \frac{3}{5}f(n-1) \\ &= \frac{4}{5} + \left(\frac{3}{5}\right) \cdot \frac{4}{5} + \left(\frac{3}{5}\right)^2 f(n-1) \\ &= \frac{4}{5} + \left(\frac{3}{5}\right) \cdot \frac{4}{5} + \left(\frac{3}{5}\right)^2 \cdot \frac{4}{5} + \left(\frac{3}{5}\right)^3 f(n-1). \end{aligned}$$

   Continuing in this manner suggests the guess

   $$f(n) = \frac{4}{5} + \left(\frac{3}{5}\right) \cdot \frac{4}{5} + \left(\frac{3}{5}\right)^2 \cdot \frac{4}{5} + \left(\frac{3}{5}\right)^3 \cdot \frac{4}{5} + \cdots + \left(\frac{3}{5}\right)^{n-1} \cdot \frac{4}{5} + \left(\frac{3}{5}\right)^n \cdot f(0)$$

Since $f(0) = 0$, we can write

$$\begin{aligned} f(n) &= \frac{4}{5} + \left(\frac{3}{5}\right) \cdot \frac{4}{5} + \left(\frac{3}{5}\right)^2 \cdot \frac{4}{5} + \left(\frac{3}{5}\right)^3 \cdot \frac{4}{5} + \cdots + \left(\frac{3}{5}\right)^{n-1} \cdot \frac{4}{5} \\ &= \frac{\frac{4}{5}\left(1 - \left(\frac{3}{5}\right)^n\right)}{1 - \left(\frac{3}{5}\right)} \\ &= 2 - 2\left(\frac{3}{5}\right)^n. \end{aligned}$$

- Homogenization: By subtracting 2 from both side, we have

$$f(n) - 2 = \frac{3}{5}f(n-1) - \frac{6}{5} = \frac{3}{5}(f(n-1) - 2).$$

  The function $g(n) = f(n) - 2$ then satisfies the recurrence $g(n) = \frac{3}{5}g(n-1)$ with initial condition $g(0) = -2$. This gives us $g(n) = -2(\frac{3}{5})^n$. Then $f(n) = g(n) + 2 = 2 - 2(\frac{3}{5})^n$.
- Verification by induction: We prove our guess is correct by induction on n. When $n = 0$, both $f(0)$ and $2 - 2\left(\frac{3}{5}\right)^n$ are 0. Now assume $f(n) = 2 - 2\left(\frac{3}{5}\right)^n$ for some n. Then

$$f(n+1) = \frac{4}{5} + \frac{3}{5}f(n) = \frac{4}{5} + \frac{3}{5}\left(2 - 2\left(\frac{3}{5}\right)^n\right) = \frac{4}{5} + \frac{6}{5} - \frac{6}{5}\left(\frac{3}{5}\right)^n = 2 - 2\left(\frac{3}{5}\right)^{n+1}$$

  as it should be.

(c) $f(n) = 3f(n/2) + n$, $f(1) = 1$, where n is a power of 2.

Solution:

- Unwinding: We try to guess a solution by unwinding the formula for $f(n)$:

$$\begin{aligned} f(n) &= 3f\left(\frac{n}{2}\right) + n \\ &= 3\left(3f\left(\frac{n}{2^2}\right) + \frac{n}{2}\right) + n = 3^2 f\left(\frac{n}{2^2}\right) + 3 \cdot \frac{n}{2} + n \\ &= 3^2\left(3f\left(\frac{n}{2^3}\right) + \frac{n}{2^2}\right) + 3 \cdot \frac{n}{2} + n = 3^3 f\left(\frac{n}{2^3}\right) + 3^2 \cdot \frac{n}{2^2} + 3 \cdot \frac{n}{2} + n \end{aligned}$$

  Continuing in this manner for $\log n$ steps suggests the guess

$$f(n) = 3^{\log n} f(1) + 3^{\log n} \cdot \frac{n}{2^{\log n}} + 3^{\log n - 1} \cdot \frac{n}{2^{\log n - 1}} + \cdots + n.$$

  By $f(1) = 1$ and geometric sum formula, we can write

$$f(n) = 3^{\log n} + \frac{(3/2)^{\log n} - 1}{(3/2) - 1} \cdot n = 3 \cdot 3^{\log n} - 2n = 3 \cdot n^{\log 3} - 2n.$$

- Homogenization: We try the homogenization $f(n) = f'(n) + an$. Then $f'(n) + an$ must equal $3(f'(n/2) + an/2) + n$, so $f'(n) - 3f'(n/2) = (a/2 + 1)n$. This is a homogeneous recurrence $f'(n) = 3f'(n/2)$ when $a = -2$. The initial condition is then $f'(1) = f(1) + 2 = 3$. $f'(n)$ solves to $f'(n) = 3f'(n/2) = 3^2 f'(n/2^2) = \cdots = 3^{\log n} f'(1) = 3 \cdot 3^{\log n} = 3 \cdot n^{\log 3}$. Then $f(n) = f'(n) - 2n = 3 \cdot n^{\log 3} - 2n$.
- Verification by induction: If we write $m = \log n$, then $f(2^m) = 3^{m+1} - 2^{m+1}$. We prove this is correct by induction on m. When $m = 0$, both $f(2^m)$ and $3^{m+1} - 2^{m+1}$ are one. Now assume $f(2^m) = 3^{m+1} - 2^{m+1}$ for some m. Then

$$f(2^{m+1}) = 3f(2^m) + 2^{m+1} = 3 \cdot (3^{m+1} - 2^{m+1}) + 2^{m+1} = 3^{m+2} - 2^{m+2}$$

  as it should be.

2. Find exact closed-form solutions to the following recurrences.

   (a) $f(n) = 8f(n-1) - 15f(n-2)$, $f(0) = 0$, $f(1) = 1$

   Solution: This is a homogeneous linear recurrence, so we guess a solution of the form $f(n) = x^n$ for some nonzero x. If our guess is correct, x^n must equal $8x^{n-1} - 15x^{n-2}$ for all n, from where $x^2 = 8x - 15$. This quadratic equation has the two solutions $x_1 = 3$ and $x_2 = 5$. Any linear combination of x_1^n and x_2^n also satisfies the recurrence. We look for a a linear combination $f(n) = sx_1^n + tx_2^n$ that satisfies the additional requirements $f(0) = 0$ and $f(1) = 1$:

   $$\begin{aligned} 0 &= f(0) = s + t \\ 1 &= f(1) = sx_1 + tx_2 = 3s + 5t. \end{aligned}$$

   The unique solution to this system is $s = -1/2$ and $t = 1/2$. Therefore

   $$f(n) = -\frac{1}{2} \cdot 3^n + \frac{1}{2} \cdot 5^n = \frac{5^n - 3^n}{2}$$

   is the solution to our recurrence.

   (b) $f(n) = f(n-1) + f(n-2) + 1$, $f(0) = 0$, $f(1) = 1$
   (**Hint:** Try homogenizing with $f(n) = g(n) + c$ for some constant c.)

   Solution: We first homogenize the recurrence. By adding one to both sides we can write

   $$f(n) + 1 = (f(n-1) + 1) + (f(n-2) + 1).$$

   The function $g(n) = f(n) + 1$ then satisfies the recurrence $g(n) = g(n-1) + g(n-2)$ with initial conditions $g(0) = 1$ and $g(1) = 2$. This is the same as the recurrence from Section 4 of Lecture 7, but with different initial conditions. We can solve it using the same method. Alternatively we can reason like this. If we define $g(-1) = 1$ then the recurrence is still satisfied for all $n \geq 1$. Then the function $h(n) = g(n-1)$ satisfies the same recurrence but with initial conditions $h(0) = g(-1) = 1$ and $h(1) = g(0) = 1$. This is exactly the same as the recurrence in Lecture 7, so

   $$h(n) = \frac{1}{\sqrt{5}} \cdot \Big(\frac{1+\sqrt{5}}{2}\Big)^{n+1} - \frac{1}{\sqrt{5}} \cdot \Big(\frac{1+\sqrt{5}}{2}\Big)^{n+1},$$

   from where

   $$g(n) = h(n-1) = \frac{1}{\sqrt{5}} \cdot \Big(\frac{1+\sqrt{5}}{2}\Big)^{n+2} - \frac{1}{\sqrt{5}} \cdot \Big(\frac{1+\sqrt{5}}{2}\Big)^{n+2},$$

   and finally

   $$f(n) = g(n) - 1 = \frac{1}{\sqrt{5}} \cdot \Big(\frac{1+\sqrt{5}}{2}\Big)^{n+2} - \frac{1}{\sqrt{5}} \cdot \Big(\frac{1+\sqrt{5}}{2}\Big)^{n+2} - 1.$$

3. Recall that a *saddle* in a table of numbers is an entry that is largest in its column and smallest in its row. In Lecture 2 we showed that every table can have at most one saddle. Here is an algorithm for finding it (if it exists):

Input: A $n \times n$ table T. Assume n is a power of two and all entries of T are distinct.
Algorithm Saddle(T)**:**
- If $n = 1$, output the (unique) entry in T.
- Otherwise,
  - Recursively run **Saddle**(T_i) on each of the four quadrants T_1, T_2, T_3, T_4 of T.
  - Let s_i be the output of **Saddle**(T_i).
  - Test if s_i is a saddle of T by comparing it to
    - all numbers in its row and column *except* those in T_i.
  - If one of s_1, s_2, s_3, or s_4 passes the test, output it.

(a) Show a sample run of **Saddle** on the following input T:

12	2	5	10
16	7	13	4
15	8	14	9
6	1	11	3

Solution: The algorithm recursively splits the instance into four quadrants

$$T_1 = \begin{matrix} 12 & 2 \\ 16 & 7 \end{matrix} \qquad T_2 = \begin{matrix} 5 & 10 \\ 13 & 4 \end{matrix} \qquad T_3 = \begin{matrix} 15 & 8 \\ 6 & 1 \end{matrix} \qquad T_4 = \begin{matrix} 14 & 9 \\ 11 & 3 \end{matrix}.$$

In the next level of recursion each of T_1, T_2, T_3, T_4 is split into its four entries and each of them is output as a candidate saddle. For example, 12, 2, 16, 7 are all considered as candidate saddles in T_1 and compared to the other entry in their row and column. Only 7 survives and is output as the saddle of T_1. Similarly, 8 and 9 are output as the saddles of T_3 and T_4, respectively. T_2 does not have a saddle so the recursive call **Saddle**(T_2) does not produce one. At this stage the candidate saddles in the four quadrants are

12	2	5	10
16	$\underline{7}$	13	4
15	$\underline{8}$	14	$\underline{9}$
6	1	11	3

Now each candidate saddle is compared to the other number in its row and column except the ones in the table it is already in. Thus 7 is compared to 8, 1, 13, and 4, and so on. 7 does not survive because it is larger than 4 (and smaller than 8). Similarly 9 does not survive because it is smaller than 10. Only 8 survives and is the output produced by **Saddle**(T).

(b) Let $C(n)$ be the worst-case number of comparisons **Saddle** performs on an $n \times n$ input. Explain why

$$C(n) \leq 4C(n/2) + 4n. \tag{1}$$

Solution: The number of comparisons $C(n)$ on a $n \times n$ table equals is the number of comparisons made by the four recursive calls, each of which is at most $C(n/2)$, plus the extra comparisons done in the test step. In general there can be as many as four candidate saddles coming in from the recursion. Each of them is compared to at most $n/2$ numbers in its row and $n/2$ numbers in its column. (Not all these comparisions are necessarily distinct; for example in part (a) 7 is compared to 8 twice, separately when each is considered as a possible candidate saddle.) So the number of extra comparisons is at most $4 \cdot (n/2 + n/2) = 4n$.

(c) Apply Theorem 6 from Lecture 7 to calculate the big-Oh asymptotic growth of $C(n)$.

Solution: We instantiate the Master Theorem with $a = 4$, $b = 2$, and $g(n) = 4n$. Then $c = \log_b a = 2$, and $g(n) = O(n) = O(n^{c-\varepsilon})$ with $\varepsilon = 1$. So we have $C(n) = O(n^c) = O(n^2)$.

(d) Obtain an exact formula for $C(n)$ assuming the inequality in (**??**) is an equality. Argue that your solution is an *upper bound* on the number of comparisons performed by **Saddle**.

Solution: We solve the recurrence $C(n) = 4C(n/2) + 4n$ with $C(1) = 0$. The solution to (**??**) can only be smaller by strong induction on n. We unwind the recurrence to obtain

$$\begin{aligned}
C(n) &= 4C(n/2) + 4n \\
&= 4(4C(n/2^2) + 4(n/2)) + 4n = 4^2C(n/2^2) + 4^2n/2 + 4n \\
&= 4^2(4C(n/2^3) + 4(n/2^2)) + 4^2n/2 + 4n = 4^3C(n/2^3) + 4^3n/2^2 + 4^2n/2 + 4n \\
&= \cdots \\
&= 4^{\log n}C(1) + 4 \cdot (n + 2n + 2^2n + \cdots + 2^{\log n - 1}n) \\
&= 4(2^{\log n} - 1)n \\
&= 4n(n-1).
\end{aligned}$$

Assuming the equality that $C(n) = 4C(n/2) + 4n$ then $C(n) = \Theta(n^2)$ by the Master Theorem, Lecture 7. At the return of each recursive call to **Saddle**, for an input of size larger than $n = 1$, there is between 0 and 4 candidate saddles to consider. Hence the expression for $C(n)$ is a worst case scenario when at the return of each recursive call, the relevant comparisons for 4 saddles are made, and so the expression for $C(n)$ is an upper bound.

4. DNA (Deoxyribonucleic acid) is a molecule that carries the genetic instructions for all known organisms and many viruses. It consists of a chain of bases. In DNA chain, there are four types of bases: **A**, **C**, **G**, **T**. For example, a DNA chain of length 10 can be **ACGTACGTAT**.

   (a) Let $g(n)$ be the number of configurations of a DNA chain of length n in which the pairs **TT** and **TG** never appear. Write a recurrence for $g(n)$. (**Hint:** Is the first base a **T**?)

   Solution: A sequence of length n will either start with a **T** or not. If it starts with a **T**, then the next base must be either **C** or **A**. Then, any valid sequence of length $n-2$ can be appended. Thus there are $2 \cdot g(n-2)$ valid sequences of length n that start with **T**. If we do not start with **T**, then we have 3 choices and can append any valid sequence of length $n-1$. Thus there are $3 \cdot g(n-1)$ choices that do not start with **T**. Together we have the recurrence: $g(n) = 3g(n-1) + 2g(n-2)$. As base cases, we have $g(1) = 4$ and $g(2) = 14$.

   (b) Solve the recurrence from part (a).

   Solution: This is a homogeneous linear recurrence, so we guess a solution of the form $g(n) = x^n$ for some nonzero x. If our guess is correct, x^n must equal $3x^{n-1} + 2x^{n-2}$ for all n, from where $x^2 = 3x + 2$. This quadratic equation has the two solutions $x_1 = \frac{3+\sqrt{17}}{2}$ and $x_2 = \frac{3-\sqrt{17}}{2}$. Any linear combination of x_1^n and x_2^n also satisfies the recurrence. We look for a a linear combination $g(n) = sx_1^n + tx_2^n$ that satisfies the additional requirements $g(0) = 1$ and $g(1) = 4$:

$$1 = g(0) = s + t$$

$$4 = g(1) = sx_1 + tx_2 = s \cdot \frac{3+\sqrt{17}}{2} + t \cdot \frac{3-\sqrt{17}}{2}.$$

   The unique solution to this system is $s = \frac{17+5\sqrt{17}}{34}$ and $t = \frac{17-5\sqrt{17}}{34}$. Therefore

$$g(n) = \frac{17+5\sqrt{17}}{34}\left(\frac{3+\sqrt{17}}{2}\right)^n + \frac{17-5\sqrt{17}}{34}\left(\frac{3-\sqrt{17}}{2}\right)^n$$

   is the solution to our recurrence.

   (c) Which one of the alternatives $g(n) = o(3^n)$, $g(n) = \Theta(3^n)$, or $3^n = o(g(n))$ is correct?

   Solution: The last one. $(3-\sqrt{17})/2$ is about -0.562 so $\left(\frac{3-\sqrt{17}}{2}\right)^n$ tends to 0 as n gets large. Therefore $g(n)$ is $\Theta\left(\left(\frac{3+\sqrt{17}}{2}\right)^n\right)$. As $(3+\sqrt{17})/2 > (3+\sqrt{9})/2 = 3$, 3^n is $o(\left(\left(\frac{3+\sqrt{17}}{2}\right)^n\right)$ so it must be $o(g(n))$ as well.

5. **(Optional)** You want to move the Towers of Hanoi, but now you have four poles. The rules are the same: n disks are initially stacked by size and the objective is to move them to another pole one by one so that at no point does a larger disk cover a smaller one.

   Consider the following strategy: If $n \leq 10$, ignore one of the poles and apply the solution from class for three poles. If $n > 10$, recursively move the top $n-10$ disks to the second pole, stack up the bottom 10 disks onto the last pole using the other three poles only, and then recursively move the $n-10$ remaining disks from the second pole to the last pole.

   Let $T(n)$ be the number of steps that it takes to move the whole stack of n disks.

(a) Write a recurrence for $T(n)$. Explain why your recurrence is correct.

Solution: The number of moves the strategy makes for n disks and 4 poles equals twice the number of moves for $n-10$ disks and 4 poles, plus the number of moves for 10 disks and 3 poles which equals $2^{10}-1 = 1023$. Therefore the recurrence is

$$T(n) = 2T(n-10) + 1023$$

for $n > 10$ and $T(n) = O(1)$ for $n \leq 10$.

(b) Show that the recurrence from part (a) satisfies $T(n) = O(2^{n/10})$.

Solution: For n sufficiently large,

$$T(n) = 2T(n-10) + 1023 = 2^2T(n-2\cdot 10) + 2\cdot 1023 + 1023 = 2^3T(n-3\cdot 10) + (1+2+2^2)1023.$$

After $\lfloor n/10 \rfloor$ steps, we get

$$T(n) = 2^{\lfloor n/10\rfloor}T(n - \lfloor n/10\rfloor \cdot 10) + (1+2+\cdots+2^{\lfloor n/10\rfloor})\cdot 1023 \leq 2^{n/10}T(k) + (2^{\lfloor n/10\rfloor+1}-1)\cdot 1023$$

for some k between 1 and 10. The last expression is $O(2^{n/10})$.

(c) Can you come up with a different strategy in which $2^{O(\sqrt{n})}$ moves are sufficient?

Solution: When $n \geq 2$, recursively move the top $n - \lfloor\sqrt{n}\rfloor$ disks to the second pole, stack up the bottom $\lceil\sqrt{n}\rceil$ disks onto the last pole using the other three poles only, and then recursively move the $n - \lceil\sqrt{n}\rceil$ remaining disks from the second pole to the last pole. This gives the recurrence

$$T(n) = 2T(n - \lceil\sqrt{n}\rceil) + 2^{\lceil\sqrt{n}\rceil} - 1$$

for the number of moves. Since T is an increasing function for every k in the range $n/2 \leq k \leq n$, we can write

$$T(k) = 2T(k - \lceil\sqrt{k}\rceil) + 2^{\lceil\sqrt{k}\rceil} - 1 \leq 2T(k - \lceil\sqrt{n/2}\rceil) + 2^{\lceil\sqrt{n}\rceil}.$$

Therefore for n sufficiently large

$$\begin{aligned} T(n) &\leq 2T(n - \lceil\sqrt{n/2}\rceil) + 2^{\lceil\sqrt{n}\rceil} \\ &\leq 2^2T(n - 2\lceil\sqrt{n/2}\rceil) + (1+2)2^{\lceil\sqrt{n}\rceil} \\ &\leq 2^3T(n - 3\lceil\sqrt{n/2}\rceil) + (1+2+2^2)2^{\lceil\sqrt{n}\rceil}. \end{aligned}$$

Continuing for just enough steps t so that the argument of $T(\cdot)$ drops below $n/2$, we get that

$$T(n) \leq 2^tT(\lfloor n/2\rfloor) + (2^t-1)2^{\lceil\sqrt{n}\rceil}$$

The value of t can be at most $2^{\lceil\sqrt{n/2}\rceil}$, so

$$T(n) \leq 2^{\lceil\sqrt{n/2}\rceil}T(\lfloor n/2\rfloor) + 2^{\lceil\sqrt{n/2}\rceil}\cdot 2^{\lceil\sqrt{n}\rceil} \leq 2^{\lceil\sqrt{n}\rceil}T(\lfloor n/2\rfloor) + 2^{2\lceil\sqrt{n}\rceil}.$$

We can derive the inequality

$$T(n) + 1 \leq 2^{\lceil\sqrt{n}\rceil}T(\lfloor n/2\rfloor) + 2^{2\lceil\sqrt{n}\rceil} + 1 \leq 2\cdot 2^{2\sqrt{n}}(T(\lfloor n/2\rfloor)+1).$$

Iterating this formula down to $n = 2$, we get that

$$T(n) \leq 2\cdot 2^{2\sqrt{n}}\cdot 2^{2\sqrt{n/2}}\cdots 2^2 \leq 2\cdot 2^{2\sqrt{n}(1+1/\sqrt{2}+1/\sqrt{2}^2+\ldots)} = 2^{O(\sqrt{n})}.$$