CS21201 Discrete Structures

Tutorial 9

Recurrences

1. The triangular numbers are defined as $t_n = 1+2+3+\cdots+n = \frac{n(n+1)}{2}$ for $n \geqslant 0$. Define $a_n = \sum_{i=0}^{n} t_i$ for $n \geqslant 0$. Find a recurrence relation for a_n, and solve it.

Solution Here, the recurrence is, $a_n - a_{n-1} = t_n = \frac{n(n+1)}{2}$ with $a_0 = t_0 = 0$.

So, we can solve by expanding as follows:

$$\begin{aligned} a_n &= a_{n-1} + \frac{1}{2}n^2 + \frac{1}{2}n \\ &= a_{n-2} + \frac{1}{2}\left[n^2 + (n-1)^2\right] + \frac{1}{2}\left[n + (n-1)\right] \\ &= \cdots = a_0 + \frac{1}{2}\sum_{i=1}^{n} i^2 + \frac{1}{2}\sum_{i=1}^{n} i = \frac{n(n+1)(2n+1)}{12} + \frac{n(n+1)}{4} = \frac{n(n+1)(n+2)}{6} \end{aligned}$$

2. Let a_n, $n \geqslant 0$, be the count of strings over $\{0,1,2\}$ containing no consecutive 1's and no consecutive 2's. Find a recurrence relation for a_n, and solve it.

Solution Let b_n, c_n, d_n, $n \geqslant 1$, denote the counts of the strings of the desired form that start with $0,1,2$, respectively. Let us also take $b_0 = 1$, $c_0 = 0$ and $d_0 = 0$. We have the following equations involving these.

$$\begin{aligned} a_n &= b_n + c_n + d_n \text{ for all } n \geqslant 0, \\ b_n &= a_{n-1} \text{ for all } n \geqslant 1, \\ c_n &= b_{n-1} + d_{n-1} = a_{n-1} - c_{n-1} \text{ for all } n \geqslant 1, \\ d_n &= b_{n-1} + c_{n-1} = a_{n-1} - d_{n-1} \text{ for all } n \geqslant 1. \end{aligned}$$

Adding the last three equations gives

$$a_n = 3a_{n-1} - (c_{n-1} + d_{n-1}) = 3a_{n-1} - (a_{n-1} - b_{n-1}) = 2a_{n-1} + b_{n-1} = 2a_{n-1} + a_{n-2}$$

for all $n \geqslant 2$. The initial conditions are $a_0 = 1$, $a_1 = 3$. The characteristic equation of the sequence is $x^2 - 2x - 1 = 0$. The roots of the characteristic equation are $1+\sqrt{2}, 1-\sqrt{2}$.

So the general solution of this recurrence is of the form

$$a_n = \beta \cdot (1+\sqrt{2})^n + \gamma \cdot (1-\sqrt{2})^n.$$

The initial conditions give

$$\begin{aligned} a_0 &= 1 = \beta + \gamma, \\ a_1 &= 3 = \beta \cdot (1+\sqrt{2}) + \gamma \cdot (1-\sqrt{2}). \end{aligned}$$

Solving gives $\beta = \frac{1+\sqrt{2}}{2}, \gamma = \frac{1-\sqrt{2}}{2}$.

Therefore, the final solution is:

$$a_n = \frac{1}{2}\left[(1+\sqrt{2})^{n+1} + (1-\sqrt{2})^{n+1}\right] \text{ for all } n \geqslant 0.$$

3. Let D_n, $n \geqslant 1$, denote that number of derangements (permutations without fixed points) of $1,2,3,\ldots,n$.

(a) Prove that $D_n = (n-1)(D_{n-1} + D_{n-2})$ for all $n \geqslant 3$.

(b) Deduce that $D_n = nD_{n-1} + (-1)^n$ for all $n \geqslant 3$.

(c) Solve for D_n.

Solution (a) Let $x_1, x_2, \ldots, x_n$ be a derangement of $1,2,\ldots,n$. Then $x_n = i$ for some $i \in \{1,2,\ldots,n-1\}$. Fix an i, and consider the following two cases:

Case 1: $x_i = n$. Then, $x_1, x_2, \ldots, x_{i-1}, x_{i+1}, \ldots, x_{n-1}$ is a derangement of $1, 2, \ldots, i-1, i+1, \ldots, n-1$, and the number of such derangements is D_{n-2}.

Case 2: $x_i \neq n$. Rename n as i occurring somewhere in the first $n-1$ positions (except the i-th one). But then, $x_1, x_2, \ldots, x_{n-1}$ is a derangement of $1, 2, \ldots, n-1$. The number of such derangements is D_{n-1}.

Now, varying i over all of the $n-1$ allowed values, we get:

$$D_n = (n-1)\big(D_{n-1} + D_{n-2}\big) \text{ for all } n \geqslant 3 \qquad (\text{Note that, } D_1 = 0, D_2 = 1).$$

(b) $D_n = (n-1)(D_{n-1} + D_{n-2}) \quad \Rightarrow \quad D_n - nD_{n-1} = (-1)\big[D_{n-1} - (n-1)D_{n-2}\big]$

We recursively simplify the righthand side of the equation above to get:

$$\begin{aligned} D_n - nD_{n-1} &= (-1)\big[D_{n-1} - (n-1)D_{n-2}\big] \\ &= (-1)^2\big[D_{n-2} - (n-2)D_{n-3}\big] \\ &= (-1)^3\big[D_{n-3} - (n-3)D_{n-4}\big] \\ &= \cdots = (-1)^{n-2}\big[D_2 - 2D_1\big] \end{aligned}$$

Since $D_1 = 0$ and $D_2 = 1$, we get: $D_n - nD_{n-1} = (-1)^{n-2} \quad \Rightarrow \quad D_n = nD_{n-1} + (-1)^n$.

(c) Diving both side of the given recurrence by $n!$, we get,

$$\frac{D_n}{n!} = \frac{D_{n-1}}{(n-1)!} + \frac{(-1)^n}{n!}$$

Now, if we recursively simplify the righthand side of the equation above, we get:

$$\begin{aligned} \frac{D_n}{n!} &= \frac{D_{n-1}}{(n-1)!} + \frac{(-1)^n}{n!} \\ &= \frac{D_{n-2}}{(n-2)!} + \frac{(-1)^{n-1}}{(n-1)!} + \frac{(-1)^n}{n!} \\ &= \frac{D_{n-3}}{(n-3)!} + \frac{(-1)^{n-2}}{(n-2)!} + \frac{(-1)^{n-1}}{(n-1)!} + \frac{(-1)^n}{n!} \\ &= \cdots = \frac{D_0}{0!} + \frac{(-1)^1}{1!} + \frac{(-1)^2}{2!} + \cdots + \frac{(-1)^{n-1}}{(n-1)!} + \frac{(-1)^n}{n!} = \sum_{k=0}^{n} \frac{(-1)^k}{k!} \end{aligned}$$

$$\therefore D_n = n!.\sum_{k=0}^{n} \frac{(-1)^k}{k!} = \,!n \quad (\text{called } \textit{subfactorial} \text{ of n}).$$

4. Let a_n, $n \geqslant 1$, satisfy $a_1 = 1$, and $a_n = \begin{cases} 2a_{n-1} & \text{if } n \text{ is odd} \\ 2a_{n-1} + 1 & \text{if } n \text{ is even} \end{cases}$ for $n \geqslant 2$. Develop a recurrence relation for a_n, that holds for both odd and even n, and solve it.

Solution For both cases, $a_n - a_{n-2} = 2(a_{n-1} - a_{n-3}) \Rightarrow a_n - 2a_{n-1} - a_{n-2} + 2a_{n-3} = 0$. Moreover, $a_2 = 3$ and $a_3 = 6$.

The characteristics equation is, $r^3 - 2r^2 - r + 2 = 0$. Solving this, we get the three roots as, $-1, 1, 2$.

So, the general recurrence form is, $a_n = \alpha.(-1)^n + \beta.1^n + \gamma.2^n$.

Now, $a_1 = -\alpha + \beta + 2\gamma = 1$, $a_2 = \alpha + \beta + 4\gamma = 3$ and $a_3 = -\alpha + \beta + 8\gamma = 6$.

Solving these, we get, $\alpha = \frac{1}{6}$, $\beta = -\frac{1}{2}$, and $\gamma = \frac{5}{6}$.

Therefore, $a_n = \frac{1}{6}.(-1)^n - \frac{1}{2}.1^n + \frac{5}{6}.2^n = \frac{1}{6}.\big[5.2^n + (-1)^n - 3\big]$.

5. Solve the following recurrence relation, and deduce the closed-form expression for a_n.

$$a_n = a_{n-1} + 8a_{n-2} - 12a_{n-3} + 2^n \ (\text{for } n \geqslant 3), \quad \text{with } a_0 = 1,\ a_1 = 1,\ a_2 = \frac{83}{5}.$$

Solution Given the recurrence $a_n = a_{n-1} + 8a_{n-2} - 12a_{n-3} + 2^n$ (for $n \geqslant 2$), we find that the homogeneous part is

$$a_n = a_{n-1} + 8a_{n-2} - 12a_{n-3}.$$

So, the *characteristic equation* that results from the homogeneous part will be $x^3-x^2-8x+12=0$.

Solving for x, we get

$$x^3-x^2-8x+12=0 \quad\Rightarrow\quad (x-2)^2(x+3)=0 \quad\Rightarrow\quad x=2 \text{ (double root)},\ x=-3.$$

Therefore, the homogeneous solution is $a_n^{(h)}=(A+Bn)2^n+C(-3)^n$.

Now, the particular solution (with respect to 2^n) will be $a_n^{(p)}=Dn^2 2^n$ $\big($due to conflict term $(A+Bn)2^n$ in $a_n^{(h)}\big)$.

Solving for constant D with the help of the given recurrence, we get

$$\begin{aligned} Dn^2 2^n &= D(n-1)^2 2^{n-1}+8D(n-2)^2 2^{n-2}-12D(n-3)^2 2^{n-3}+2^n \\ \Rightarrow \quad Dn^2 &= \frac{D}{2}(n-1)^2+2D(n-2)^2-\frac{3}{2}D(n-3)^2+1. \end{aligned}$$

Comparing the constant terms (coefficients of n^0) in above equation, we find

$$0=\frac{D}{2}+8D-\frac{27}{2}+1 \quad\Rightarrow\quad D=\frac{1}{5}.$$

So, the general form of the solution is:

$$a_n=a_n^{(h)}+a_n^{(p)}=(A+Bn)2^n+C(-3)^n+\frac{1}{5}n^2 2^n.$$

Solving for the constants A,B,C, we get:

$$\begin{aligned} a_0=1=A+C \quad&\Rightarrow\quad C=1-A, \\ a_1=1=2(A+B)-3C+\frac{2}{5} \quad&\Rightarrow\quad 5A+2B=\frac{18}{5}, \\ a_2=\frac{83}{5}=4(A+2B)+9C+\frac{16}{5} \quad&\Rightarrow\quad -5A+8B=\frac{22}{5}. \end{aligned}$$

The above three equations produce: $A=\frac{2}{5}$, $B=\frac{4}{5}$, $C=\frac{3}{5}$.

Hence, the final solution is:

$$a_n = \left(\frac{2}{5}+\frac{4}{5}n\right)2^n+\frac{3}{5}(-3)^n+\frac{1}{5}n^2 2^n = \frac{1}{5}\left[(1+2n)2^{n+1}+(-1)^n 3^{n+1}+n^2 2^n\right].$$

6. Solve the recurrence relation $a_n=na_{n-1}+n(n-1)a_{n-2}+n!$ for $n\geqslant 2$, with $a_0=0$, $a_1=1$.

Solution Diving both side of the given recurrence by $n!$, we get,

$$\frac{a_n}{n!}=\frac{a_{n-1}}{(n-1)!}+\frac{a_{n-2}}{(n-2)!}+1$$

Let, $b_n=\frac{a_n}{n!}$. So, we have, $b_n=b_{n-1}+b_{n-2}+1$ with $b_0=0,\ b_1=1$.

Homogeneous Solution: $\quad b_n^{(h)}=\alpha_1.\left(\frac{1+\sqrt{5}}{2}\right)^n+\alpha_2.\left(\frac{1-\sqrt{5}}{2}\right)^n$ (similar as Fibonacci sequence).

Particular Solution: $\quad b_n^{(p)}=\beta.1^n=\beta$.

So, from the recurrence, we get, $\beta=\beta+\beta+1 \Rightarrow \beta=-1$.

Final Solution (general form): $\quad b_n=b_n^{(h)}+b_n^{(p)}=\alpha_1.\left(\frac{1+\sqrt{5}}{2}\right)^n+\alpha_2.\left(\frac{1-\sqrt{5}}{2}\right)^n-1.$

Using the base conditions, $b_0=0,\ b_1=1$, we can solve for $\alpha_1=\frac{3+\sqrt{5}}{2\sqrt{5}}$ and $\alpha_2=-\frac{3-\sqrt{5}}{2\sqrt{5}}$.

Finally, $b_n=\left(\frac{3+\sqrt{5}}{2\sqrt{5}}\right).\left(\frac{1+\sqrt{5}}{2}\right)^n-\left(\frac{3-\sqrt{5}}{2\sqrt{5}}\right).\left(\frac{1-\sqrt{5}}{2}\right)^n-1=\frac{1}{\sqrt{5}}.\left[\left(\frac{1+\sqrt{5}}{2}\right)^{n+2}-\left(\frac{1-\sqrt{5}}{2}\right)^{n+2}\right]-1.$

$$\therefore a_n=n!.b_n=\frac{n!}{\sqrt{5}}.\left[\left(\frac{1+\sqrt{5}}{2}\right)^{n+2}-\left(\frac{1-\sqrt{5}}{2}\right)^{n+2}-\sqrt{5}\right] \quad \text{for } n\geqslant 0$$

7. Solve the recurrence relation: $a_0 = 1$, $a_1 = 2$, and $a_n = \dfrac{2a_{n-1}^3}{a_{n-2}^2}$ for $n \geqslant 2$.

Solution Taking log (base 2) of the given recurrence, we get,

$$\log a_n = \log 2 + 3\log a_{n-1} - 2\log a_{n-2}$$

Let, $b_n = \log a_n$. So, we have, $b_n = 3b_{n-1} - 2b_{n-2} + 1$ with $b_0 = 0,\ b_1 = 1$.

Homogeneous Solution: The homogeneous part gives the characteristics equation as, $r^2 - 3r + 2 = 0$. Since the roots of this equation are 1 and 2, we can write $b_n^{(h)} = \alpha_1.1^n + \alpha_2.2^n = \alpha_1 + \alpha_2.2^n$.

Particular Solution: $b_n^{(p)} = \beta.n.1^n = \beta.n$.

Now, from the recurrence, we get, $\beta.n = 3.\beta.(n-1) - 2.\beta.(n-2) + 1 \ \Rightarrow\ \beta = -1$.

Final Solution (general form): $b_n = b_n^{(h)} + b_n^{(p)} = \alpha_1 + \alpha_2.2^n - n$.

Using the base conditions, $b_0 = 0,\ b_1 = 1$, we can solve for $\alpha_1 = -2$ and $\alpha_2 = 2$.

Finally, $b_n = -2 + 2.2^n - n = 2^{n+1} - n - 2$.

$$\therefore a_n = 2^{b_n} = 2^{2^{n+1}-n-2} \quad \text{for } n \geqslant 0$$

8. How many lines are printed by the call $f(n)$ for an integer $n \geqslant 0$?

```
void f ( int n )
{
    int m;
    printf("Hi\n");
    m = n - 1;
    while (m >= 0) { f(m); m -= 2; }
}
```

Solution Suppose, L_n be the number of lines printed by the call $f(n)$. So,

$$L_n = \begin{cases} 1 + \sum_{k=0}^{(n-1)/2} L_{2k}, & \text{if } n \text{ is odd} \\ 1 + \sum_{k=0}^{(n-2)/2} L_{2k+1}, & \text{if } n \text{ is even} \end{cases} \qquad \text{for } n \geqslant 0$$

Here, for both cases we derive, $L_n - L_{n-2} = L_{n-1}$ with $L_0 = 1,\ L_1 = 2$.

This is the same recurrence as we got in Fibonnaci series computation having $L_n = F_{n+1}$. Therefore,

$$L_n = \frac{1}{\sqrt{5}}\Big[\Big(\frac{1+\sqrt{5}}{2}\Big)^{n+1} + \Big(\frac{1-\sqrt{5}}{2}\Big)^{n+1}\Big].$$

9. Solve for the following divide-and-conquer recurrence: $T(n) = 2T(n/2) + \dfrac{n}{\log n}$ with $T(1) = 1$.

Solution Dividing both the sides of the given recurrence by n, we obtain:

$$\frac{T(n)}{n} = \frac{T(n/2)}{n/2} + \frac{1}{\log n}$$

Assuming $n = 2^k$ and $S(k) = \frac{T(n)}{n} = \frac{T(2^k)}{2^k}$, we can rewrite the above as:

$$S(k) = S(k-1) + \frac{1}{k} \quad \text{with } S(0) = 1$$

Now, if we recursively simplify the righthand side of the equation above, we get:

$$\begin{aligned}
S(k) &= S(k-1)+\frac{1}{k} \\
&= S(k-2)+\frac{1}{k-1}+\frac{1}{k} \\
&= S(k-3)+\frac{1}{k-2}+\frac{1}{k-1}+\frac{1}{k} \\
&= \cdots = S(0)+\sum_{i=1}^{k}\frac{1}{i} \\
\therefore T(n) &= nS(k) = n\Big[1+\sum_{i=1}^{\log n}\frac{1}{i}\Big].
\end{aligned}$$

10. Solve the following recurrence relation, and deduce the closed-form expression for $T(n)$.

$$T(n)=\begin{cases}\sqrt{n}T(\sqrt{n})+n(\log_2 n)^d, & \text{if } n>2\\ 2 \quad , & \text{if } n=2\end{cases} \quad (d\geqslant 0).$$

Solution Given that $T(n)=\sqrt{n}T(\sqrt{n})+n\log_2^d n$ (where $d\geqslant 0$) and $T(2)=2$, we have:

$$\begin{aligned}
\frac{T(n)}{n} &= \frac{T(\sqrt{n})}{\sqrt{n}}+\log_2^d n && \ldots\Big[\textit{dividing both sides by } n\Big]\\
\Rightarrow \quad S(n) &= S(\sqrt{n})+\log_2^d n && \ldots\Big[\textit{assuming } S(n)=\frac{T(n)}{n}\Big]\\
\Rightarrow \quad S(2^{2^k}) &= S(2^{2^{(k-1)}})+(2^k)^d && \ldots\Big[\textit{substituting } n=2^{2^k}\Big]\\
\Rightarrow \quad R(k) &= R(k-1)+(2^k)^d && \ldots\Big[\textit{let } R(k)=S(2^{2^k})\Big]\\
\Rightarrow \quad R(k) &= R(0)+(2^d)^1+(2^d)^2+\cdots+(2^d)^{k-1}+(2^d)^k && \ldots\Big[\textit{because } (2^k)^d=2^{kd}=(2^d)^k\Big]\\
\Rightarrow \quad R(k) &= 1+\sum_{i=1}^{k}(2^d)^i && \ldots\Big[S(2)=\frac{T(2)}{2}=1,\ \textit{implying } R(0)=S(2^{2^0})=1\Big]\\
\Rightarrow \quad R(k) &= \begin{cases}\frac{(2^d)^{(k+1)}-1}{2^d-1}, & \text{if } d>0\\ 1+k, & \text{if } d=0\end{cases}
\end{aligned}$$

$$\therefore\ R(k)=S(2^{2^k})=\begin{cases}\frac{(2^d)^{(k+1)}-1}{2^d-1}, & \text{if } d>0\\ 1+k, & \text{if } d=0\end{cases}, \quad \text{where } n=2^{2^k} \text{ and } S(n)=\frac{T(n)}{n}.$$

Finally,

$$S(n)=\begin{cases}\frac{2^d\log_2^d n-1}{2^d-1}, & \text{if } d>0\\ 1+\log_2\log_2 n, & \text{if } d=0\end{cases} \quad\Rightarrow\quad T(n)=\begin{cases}\frac{2^d n\log_2^d n-n}{2^d-1}, & \text{if } d>0\\ n+n\log_2\log_2 n, & \text{if } d=0\end{cases}$$

11. Deduce the running times of divide-and-conquer algorithms in the big-Θ notation if their running times satisfy the following recurrence relations.

(a) $T(n)=T(2n/3)+T(n/3)+n\log n$. **(b)** $T(n)=T(n/5)+T(7n/10)+n$.

Solution (a) This recurrence is not in the standard form described earlier, but we can still solve it using recursion trees. Now nodes in the same level of the recursion tree have different values, and different leaves are at different levels. However, the nodes in any complete level (that is, above any of the leaves) sum to $\approx n\log n$. Moreover, every leaf in the recursion tree has depth between $\log_3 n$ and $\log_{3/2} n$. To derive an upper bound, we overestimate $T(n)$ by ignoring the base cases and extending the tree downward to the level of the deepest leaf. Similarly, to derive a lower bound, we overestimate $T(n)$ by counting only nodes in the tree up to the level of the shallowest leaf. These observations give us the upper and lower bounds.

$$\begin{aligned}
(c_1.n\log n)(\log_3 n) &\leqslant T(n) \leqslant (c_2.n\log n)(\log_{3/2} n)\\
(c_1.n\log n)\Big(\frac{\log n}{\log 3}\Big) &\leqslant T(n) \leqslant (c_2.n\log n)\Big(\frac{\log n}{\log 3-1}\Big)
\end{aligned}$$

Since these bounds differ by only a constant factor, we have $T(n) = \Theta(n \log^2 n)$.

(b) Again, we have a lopsided recursion tree. If we look only at complete levels of the tree, we find that the level sums form a descending geometric series,

$$T(n) = n + \frac{9n}{10} + \frac{81n}{100} + \cdots \quad \Rightarrow \quad n \leqslant T(n) \leqslant 10n$$

We can get an upper bound by ignoring the base cases entirely and growing the tree out to infinity, and we can get a lower bound by only counting nodes in complete levels. Either way, the geometric series is dominated by its largest term, so $T(n) = \Theta(n)$.

Additional Exercises

12. Consider a linear recurrence relation with constant coefficients having characteristic equation $(x-r)^\mu$ for some $\mu \in \mathbb{N}$, and with a non-homogeneous part $f(n) = n^t r^n$. Using the theory of generating functions, prove that the particular solution for this recurrence relation is of the form

$$n^\mu (u_t n^t + u_{t-1} n^{t-1} + \cdot + u_2 t^2 + u_1 t + u_0) r^n.$$

13. Foosia and Barland play a long series of ODI matches. In the first game, Foosia bats first. After that, the team that wins a match must bat first in the next match. For each team, the probability of win is p if it bats first. Assume that $0 < p < 1$. Find the probability p_n that Foosia wins the n-th match. What is $\lim_{n\to\infty} p_n$?

14. Pell numbers are defined as $P_0 = 0$, $P_1 = 2$, $P_n = 2P_{n-1} + P_{n-2}$ for $n \geqslant 2$.

(a) Deduce a closed-form formula for P_n.

(b) Prove that $\begin{pmatrix} P_{n+1} & P_n \\ P_n & P_{n-1} \end{pmatrix} = \begin{pmatrix} 2 & 1 \\ 1 & 0 \end{pmatrix}^n$ for all $n \geqslant 1$.

(c) Prove that $\lim_{n\to\infty} \dfrac{P_{n-1} + P_n}{P_n} = \sqrt{2}$.

(d) Prove that if P_n is prime, then n is also prime.

15. The Pell–Lucas numbers are defined as $Q_0 = Q_1 = 2$, $Q_n = 2Q_{n-1} + Q_{n-2}$ for $n \geqslant 2$.

(a) Deduce a closed-form formula for Q_n.

(b) Prove that $Q_n = P_{2n}/P_n$ for all $n \geqslant 1$.

16. Let $a_0 = 1$, and $a_n = \frac{5}{2}a_{n-1} - a_{n-2}$ for all $n \geqslant 2$. Find a_1 such that the sequence a_n converges.

17. A set of natural numbers is called *selfish* if it contains its size as a member. Let s_n denote the number of selfish subsets of $\{1, 2, 3, \ldots, n\}$ for $n \geqslant 1$. Develop a recurrence relation for s_n, and solve it.

18. Let us call a selfish set A minimal if no proper subset of A is selfish. Let S_n denote the number of minimal selfish subsets of $\{1, 2, 3, \ldots, n\}$. Develop a recurrence relation for S_n, and solve it.

19. Let a_n, $n \geqslant 0$, denote the number of binary strings of length n, not containing the pattern 101. Develop a recurrence relation for a_n, and solve it.

20. Solve the recurrence relation: $a_0 = 1$, $a_1 = 2$, $a_n = a_{n-1} + 2a_{n-2} + n^2 + 2^n$ for $n \geqslant 2$.

21. Solve the recurrence relation: $a_0 = 1$, $a_1 = 2$, $a_n = 4a_{n-2} + 2^n + n3^n$ for $n \geqslant 2$.

22. Solve the recurrence relation: $a_0 = 0$, $a_1 = 1$, $a_2 = 2$, $a_n = a_{n-1} + a_{n-2} - a_{n-3} + n^2 + n + (-1)^n$ for $n \geqslant 3$.

23. Solve the recurrence relation $na_n = (n+1)a_{n-1} + 2n$ for $n \geqslant 1$, with the initial condition $a_0 = 0$.

24. Solve the recurrence relation $a_0 = \frac{2}{3}$, and $a_n = 2a_{n-1}^2 - 1$ for $n \geqslant 1$.

25. A sequence a_n is defined recursively as $a_1 = 1$, $a_2 = 2$, $a_3 = 24$, and $a_n = \dfrac{6a_{n-1}^2 a_{n-3} - 8a_{n-1}a_{n-2}^2}{a_{n-2}a_{n-3}}$ for $n \geqslant 4$. Prove that a_n is an integer for all $n \in \mathbb{N}$.

26. Consider the recurrence relation $a_n = a_{n-1} + 3a_{n-2} - a_{n-3}$ for $n \geqslant 3$. Find a matrix A such that $\begin{pmatrix} a_3 \\ a_4 \\ a_5 \end{pmatrix} = A \begin{pmatrix} a_0 \\ a_1 \\ a_2 \end{pmatrix}$. Express $\begin{pmatrix} a_{3n} \\ a_{3n+1} \\ a_{3n+2} \end{pmatrix}$ in terms of A and a_0, a_1, a_2, for all $n \in \mathbb{N}_0$.

27. So far, we have solved recurrence relations. In this exercise, we reverse this process, that is, from a sequence, we generate a recurrence relation, of which the given sequence is a solution. We concentrate only on linear recurrence relations with constant coefficients (homogeneous/non-homogeneous). Solve the following parts for the given sequences with the orders of the recurrence relations as specified.

(a) $(2+\sqrt{3})^n + (2-\sqrt{3})^n$, order two.
(b) $2^n + 3^n$, order two.
(c) $2^n + 3^n$, order one.
(d) $2^n + n3^n$, order two.
(e) $2^n + n3^n$, order one.
(f) $2^n + n3^n + n^2 4^n$, order three.
(g) $2^n + n3^n + n^2 4^n$, order two.
(h) $2^n + n3^n + n^2 4^n$, order one.

28. Let $A(x) = 1 + \dfrac{1}{\sqrt{1-2x}}$ be the (ordinary) generating function of a sequence a_n, $n \geqslant 0$. Develop a recurrence relation for the sequence.

29. Let a_n denote the number of strings w of length n over the alphabet $\{A, C, G, T\}$ such that the number of T in w is a multiple of 3. Find a closed-form expression for a_n.

30. How many lines are printed by the call $g(n, 0, 0)$ for an integer $n \geqslant 0$?

```
void g ( int n, int i, int flag )
{
    if (i == n) { printf("Hola\n"); return; }
    g(n, i+1, flag); g(n, i+1, flag); g(n, i+1, flag);
    if (flag == 0) g(n, i+1, 1);
}
```

31. **(a)** How many strings of length n over the alphabet $\{A, C, G, T\}$ are there, in which T never appears after A? Notice that there is no restriction on the appearances of T before the first occurrence of A.

(b) Modify the function `g()` of the last exercise so as to print precisely the strings of Part (a).

32. How many strings of length n over the alphabet $\{A, C, G, T\}$ are there, in which the pattern TT (two consecutive T's) never appears after A? Note that TT may appear before the first occurrence of A, and that single isolated T's may appear after A.

33. Find big-Θ estimates for the following positive-real-valued increasing functions $f(n)$.

(a) $f(n) = 125 f(n/4) + 2n^3$ whenever $n = 4^t$ for $t \geqslant 1$.
(b) $f(n) = 125 f(n/5) + 2n^3$ whenever $n = 5^t$ for $t \geqslant 1$.
(c) $f(n) = 125 f(n/6) + 2n^3$ whenever $n = 6^t$ for $t \geqslant 1$.

34. Let the running time of a recursive algorithm satisfy the recurrence

$$T(n) = aT(n/b) + cn^d \log^e n$$

for some $e \in \mathbb{N}$. Let $t = \log_b a$. Deduce the running time $T(n)$ in the big-Θ notation for the three cases: (i) $t < d$, (ii) $t > d$, and (iii) $t = d$.

35. Let t be the number of one-bits in n. Suppose that the running time of a divide-and-conquer algorithm satisfies the recurrence $T(n) = 2T(n/2) + nt$. When n is a power of 2, we have $t = 1$, so $T(n) = 2T(n/2) + n$. Why does this not imply that $T(n) = \Theta(n \log n)$? Find a correct estimate for $T(n)$ in the big-O notation. (**Remark:** There exist algorithms whose running times depend on t. Example: Left-to-right exponentiation.)

36. Let the running time of a recursive algorithm satisfy the recurrence

$$T(n) = aT(\sqrt{n}) + h(n).$$

Deduce the running time $T(n)$ in the big-Θ notation for the cases: (i) $h(n) = n^d$ for some $d \in \mathbb{N}$, and (ii) $h(n) = \log^d n$ for some $d \in \mathbb{N}_0$.

37. [*Karatsuba multiplication*] You want to multiply two polynomials $a(x)$ and $b(x)$ of degree (or degree bound) $n-1$. Each of the input polynomials is stored in an array of n floating-point variables. The product $c(x) = a(x)b(x)$ is of degree (at most) $2n-2$, and can be stored in an array of size $2n-1$.

(a) Use the school-book multiplication method to compute $c(x)$ (use the convolution formula). Deduce the running time of this algorithm.

(b) Let $t = \lceil n/2 \rceil$. Divide the input polynomials as $a(x) = x^t a_{hi}(x) + a_{lo(x)}$ and $b(x) = x^t b_{hi}(x) + b_{lo(x)}$, where each part of a and b is a polynomial of degree $\leqslant t-1$. But then

$$c(x) = a_{hi}(x)b_{hi}(x)x^{2t} + \Big(a_{hi}(x)b_{lo}(x) + a_{lo}(x)b_{hi}(x)\Big)x^t + a_{lo}(x)b_{lo}(x).$$

The obvious recursive algorithm uses this formula to compute $c(x)$ by making four recursive calls on polynomials of degrees $\leqslant t-1$. Deduce the running time of this algorithm.

(c) Reduce the number of recursive calls to three (how?). Deduce the running time of this algorithm.

38. In the quick-sort algorithm, two recursive calls are made on arrays of sizes i and $n-i-1$ for some $i \in \{0, 1, 2, \ldots, n-1\}$ (assuming that there are no duplicates in the input array). Suppose that all these values of i are equally likely. Deduce the expected running time of quick sort under these assumptions.

39. Suppose that an algorithm, upon an input of size n, recursively solves two instances of size $n/2$ and three instances of size $n/4$. Let the "divide + combine" time be $h(n)$. Find the running times of the algorithm if

(a) $h(n) = 1$, **(b)** $h(n) = n$, **(c)** $h(n) = n^2$, **(d)** $h(n) = n^3$.

40. Use the method of recursion trees to derive the running times of the following algorithms.

(a) Majority finding: $T(n) = T(n/2) + cn$, where c is a positive constant.
(b) Stooge sort: $T(n) = 3T(2n/3) + c$, where c is a positive constant.

41. Deduce the running times of divide-and-conquer algorithms in the big-Θ notation if their running times satisfy the following recurrence relations.

(a) $T(n) = T(2n/3) + T(n/3) + 1$. **(b)** $T(n) = T(2n/3) + T(n/3) + n$.
(c) $T(n) = T(2n/3) + T(n/3) + n \log n$. **(d)** $T(n) = T(2n/3) + T(n/3) + n^2$.

42. Deduce the running times of divide-and-conquer algorithms in the big-Θ notation if their running times satisfy the following recurrence relations.

(a) $T(n) = T(n/5) + T(7n/10) + 1$. **(b)** $T(n) = T(n/5) + T(7n/10) + n$.
(c) $T(n) = T(n/5) + T(7n/10) + n \log n$. **(d)** $T(n) = T(n/5) + T(7n/10) + n^2$.

43. Consider the following variant of stooge sort for sorting an array A of size n.

1. Recursively sort the first $\lceil 3n/4 \rceil$ elements of A.
2. Recursively sort the last $\lceil 3n/4 \rceil$ elements of A.
3. Recursively sort the first $\lceil n/2 \rceil$ elements of A.

(a) Prove that this algorithm correctly sorts A.
(b) Derive the asymptotic running time of this algorithm.