Question 1: On the first page of your assignment, write your name and student number.

Solution:

- Name: Daniel Alfredsson
- Student number: 11

Question 2: The function \( f : \mathbb{N} \to \mathbb{N} \) is defined by

\[
\begin{align*}
f(0) &= 7, \\
f(n) &= 2^n - 7 + 2 \cdot f(n-1) \quad \text{if } n \geq 1.
\end{align*}
\]

- Determine \( f(n) \) for \( n = 0, 1, 2, 3, 4, 5 \).
- Prove that \( f(n) = n \cdot 2^n + 7 \) for all integers \( n \geq 0 \).

Solution: The value of \( f(0) \) is given to be 7. From the recurrence, we get

\[
\begin{align*}
f(1) &= 2^1 - 7 + 2 \cdot f(0) \\
     &= 2 - 7 + 2 \cdot 7 \\
     &= 9.
\end{align*}
\]

From the recurrence, we get

\[
\begin{align*}
f(2) &= 2^2 - 7 + 2 \cdot f(1) \\
     &= 4 - 7 + 2 \cdot 9 \\
     &= 15.
\end{align*}
\]

From the recurrence, we get

\[
\begin{align*}
f(3) &= 2^3 - 7 + 2 \cdot f(2) \\
     &= 8 - 7 + 2 \cdot 15 \\
     &= 31.
\end{align*}
\]

From the recurrence, we get

\[
\begin{align*}
f(4) &= 2^4 - 7 + 2 \cdot f(3) \\
     &= 16 - 7 + 2 \cdot 31 \\
     &= 71.
\end{align*}
\]
From the recurrence, we get
\[
\begin{align*}
 f(5) &= 2^5 - 7 + 2 \cdot f(4) \\
 &= 32 - 7 + 2 \cdot 71 \\
 &= 167.
\end{align*}
\]

Next we prove that
\[
f(n) = n \cdot 2^n + 7
\]
for all integers \(n \geq 0\).

The base case is when \(n = 0\). In this case, the left-hand side is \(f(0)\), which is 7, whereas the right-hand side is \(0 \cdot 2^0 + 7\), which is also 7. Thus, the claim is true if \(n = 0\).

For the induction step, let \(n \geq 1\) and assume that the claim is true for \(n - 1\). Thus, we assume that
\[
f(n - 1) = (n - 1) \cdot 2^{n-1} + 7.
\]

We have to prove that the claim is true for \(n\). In other words, we have to prove that
\[
f(n) = n \cdot 2^n + 7.
\]

Here we go:
\[
\begin{align*}
f(n) &= 2^n - 7 + 2 \cdot f(n - 1) \quad \text{(from the recurrence)} \\
&= 2^n - 7 + 2 \cdot ((n - 1) \cdot 2^{n-1} + 7) \quad \text{(from the assumption)} \\
&= 2^n - 7 + (n - 1) \cdot 2^n + 14 \\
&= n \cdot 2^n + 7.
\end{align*}
\]

This proves the induction step.

**Question 3:** The functions \(f : \mathbb{N} \to \mathbb{N}\), \(g : \mathbb{N}^2 \to \mathbb{N}\), and \(h : \mathbb{N} \to \mathbb{N}\) are recursively defined as follows:
\[
\begin{align*}
f(n) &= g(n, h(n)) \quad \text{if } n \geq 0, \\
g(m, 0) &= 0 \quad \text{if } m \geq 0, \\
g(m, n) &= g(m, n - 1) + m \quad \text{if } m \geq 0 \text{ and } n \geq 1, \\
h(0) &= 1, \\
h(n) &= 2 \cdot h(n - 1) \quad \text{if } n \geq 1.
\end{align*}
\]

Solve these recurrences for \(f\), i.e., express \(f(n)\) in terms of \(n\).

**Solution:** From the definitions, we see that the function \(f\) is defined in terms of the functions \(g\) and \(h\); the function \(g\) is defined in terms of the function \(g\) only; the function \(h\) is defined in terms of the function \(h\) only.

Based on this, we first solve the recurrence for \(g\), then we solve the recurrence for \(h\). At the end we will figure out what the function \(f\) does.
We start with the function $g$. If you stare long enough at the recurrence for $g$, then you will guess that $g(m, n)$ multiplies $m$ and $n$ by repeated addition. We verify that this guess is correct: We are going to prove that

$$g(m, n) = mn$$

for all $m \geq 0$ and $n \geq 0$.

We fix an integer $m \geq 0$. Now we are going to use induction on $n$.

The base case is when $n = 0$. In this case, the left-hand side is $g(m, 0)$, which is 0, whereas the right-hand side is $m \cdot 0$, which is also 0. Thus, the claim is true if $n = 0$.

For the induction step, let $n \geq 1$ and assume that the claim is true for $n - 1$. Thus, we assume that

$$g(m, n - 1) = m(n - 1).$$

We will show that $g(m, n) = mn$:

\[
g(m, n) = g(m, n - 1) + m \quad \text{(from the recurrence)} \\
= m(n - 1) + m \quad \text{(from the assumption)} \\
= mn.
\]

This proves the induction step.

We next go to the function $h$. If you stare long enough at the recurrence for $h$, then you will guess that $h(n) = 2^n$. We verify that this guess is correct: We are going to prove that

$$h(n) = 2^n$$

for all $n \geq 0$.

The base case is when $n = 0$. In this case, the left-hand side is $h(0)$, which is 1, whereas the right-hand side is $2^0$, which is also 1. Thus, the claim is true if $n = 0$.

For the induction step, let $n \geq 1$ and assume that the claim is true for $n - 1$. Thus, we assume that

$$h(n - 1) = 2^{n-1}.$$ 

We will show that $h(n) = 2^n$:

\[
h(n) = 2 \cdot h(n - 1) \quad \text{(from the recurrence)} \\
= 2 \cdot 2^{n-1} \quad \text{(from the assumption)} \\
= 2^n.
\]

This proves the induction step.

Now we can determine $f(n)$:

\[
f(n) = g(n, h(n)) \quad \text{(from the definition)} \\
= n \cdot h(n) \quad \text{($g$ multiplies)} \\
= n \cdot 2^n. \quad \text{($h(n) = 2^n$)}
\]
Question 4: The sequence of numbers $a_n$, for $n \geq 0$, is recursively defined as follows:

\[
\begin{align*}
a_0 &= 0, \\
a_1 &= 1, \\
a_n &= 2 \cdot a_{n-1} + a_{n-2} \text{ if } n \geq 2.
\end{align*}
\]

- Determine $a_n$ for $n = 0, 1, 2, 3, 4, 5$.

- Prove that

\[
a_n = \frac{(1 + \sqrt{2})^n - (1 - \sqrt{2})^n}{2\sqrt{2}} \quad (1)
\]

for all integers $n \geq 0$.

**Hint:** What are the solutions of the equation $x^2 = 2x + 1$? Using these solutions will simplify the proof.

- Since the numbers $a_n$, for $n \geq 0$, are obviously integers, the fraction on the right-hand side of (1) is an integer as well.

Prove that the fraction on the right-hand side of (1) is an integer using only Newton’s Binomial Theorem.

Solution: We are given that $a_0 = 0$ and $a_1 = 1$. From the recurrence, we get

\[
\begin{align*}
a_2 &= 2 \cdot a_1 + a_0 \\
&= 2 \cdot 1 + 0 \\
&= 2. \\
a_3 &= 2 \cdot a_2 + a_1 \\
&= 2 \cdot 2 + 1 \\
&= 5. \\
a_4 &= 2 \cdot a_3 + a_2 \\
&= 2 \cdot 5 + 2 \\
&= 12. \\
a_5 &= 2 \cdot a_4 + a_3 \\
&= 2 \cdot 12 + 5 \\
&= 29.
\end{align*}
\]

Next we are going to prove that for all $n \geq 0$,

\[
a_n = \frac{(1 + \sqrt{2})^n - (1 - \sqrt{2})^n}{2\sqrt{2}}.
\]
The hint says that we should determine the solutions of the quadratic equation \( x^2 = 2x + 1 \).

This equation has two solutions

\[
\alpha = 1 + \sqrt{2} \quad \text{and} \quad \beta = 1 - \sqrt{2}.
\]

Thus, we have to prove that for all \( n \geq 0 \),

\[
a_n = \frac{\alpha^n - \beta^n}{2\sqrt{2}}.
\]

We will prove this by induction. By the way, using \( \alpha \) and \( \beta \) simplifies the algebra!

The first base case is when \( n = 0 \). In this case, the left-hand side is \( a_0 \), which is 0, whereas the right-hand side is

\[
\frac{\alpha^0 - \beta^0}{2\sqrt{2}} = \frac{1 - 1}{2\sqrt{2}} = 0,
\]

which is also 0. Thus, the claim is true if \( n = 0 \).

The second base case is when \( n = 1 \). In this case, the left-hand side is \( a_1 \), which is 1, whereas the right-hand side is

\[
\frac{\alpha^1 - \beta^1}{2\sqrt{2}} = \frac{(1 + \sqrt{2}) - (1 - \sqrt{2})}{2\sqrt{2}} = \frac{2\sqrt{2}}{2\sqrt{2}} = 1,
\]

which is also 1. Thus, the claim is true if \( n = 1 \).

For the induction step, let \( n \geq 2 \), and assume the claim is true for \( n - 1 \) and \( n - 2 \). Thus, we assume that

\[
a_{n-1} = \frac{\alpha^{n-1} - \beta^{n-1}}{2\sqrt{2}}
\]

and

\[
a_{n-2} = \frac{\alpha^{n-2} - \beta^{n-2}}{2\sqrt{2}}.
\]

We get

\[
a_n = 2 \cdot a_{n-1} + a_{n-2} \quad \text{(from the recurrence)}
\]

\[
= 2 \cdot \left( \frac{\alpha^{n-1} - \beta^{n-1}}{2\sqrt{2}} \right) + \frac{\alpha^{n-2} - \beta^{n-2}}{2\sqrt{2}} \quad \text{(from the assumptions)}
\]

\[
= \frac{\alpha^{n-2}(2\alpha + 1) - \beta^{n-2}(2\beta + 1)}{2\sqrt{2}}
\]

\[
= \frac{\alpha^{n-2}\alpha^2 - \beta^{n-2}\beta^2}{2\sqrt{2}} \quad \text{(} 2\alpha + 1 = \alpha^2, 2\beta + 1 = \beta^2 \text{)}
\]

\[
= \frac{\alpha^n - \beta^n}{2\sqrt{2}}.
\]

This proves the induction step.
Finally, we are going to use Newton to prove that

\[ \frac{(1 + \sqrt{2})^n - (1 - \sqrt{2})^n}{2\sqrt{2}} \]

is an integer. For \( n = 0 \) and \( n = 1 \), we have already seen that this is the case.

Assume that \( n \geq 2 \). Newton tells us that

\[
(1 + \sqrt{2})^n - (1 - \sqrt{2})^n = \sum_{k=0}^{n} \binom{n}{k} (\sqrt{2})^k - \sum_{k=0}^{n} \binom{n}{k} (-\sqrt{2})^k
\]

\[ = \sum_{k=0}^{n} \binom{n}{k} \left((\sqrt{2})^k - (-\sqrt{2})^k\right). \tag{2} \]

If \( k \) is even, then

\[ (\sqrt{2})^k - (-\sqrt{2})^k = (\sqrt{2})^k - (\sqrt{2})^k = 0. \]

If \( k \) is odd, then

\[ (\sqrt{2})^k - (-\sqrt{2})^k = (\sqrt{2})^k + (\sqrt{2})^k = 2\sqrt{2} \cdot (\sqrt{2})^{k-1}. \]

Since \( k \) is odd, we have \( k \geq 1 \) and \( k - 1 \) is even. Therefore, \( (\sqrt{2})^{k-1} \) is an integer.

We conclude: For any \( k \), the \( k \)-th term in (2) is either 0 or an integer multiple of \( 2\sqrt{2} \). It follows that

\[ (1 + \sqrt{2})^n - (1 - \sqrt{2})^n \]

is an integer multiple of \( 2\sqrt{2} \). This implies that

\[ \frac{(1 + \sqrt{2})^n - (1 - \sqrt{2})^n}{2\sqrt{2}} \]

is an integer.

**Question 5:** Let \( n \) be a positive integer and consider a \( 1 \times n \) board \( B_n \) consisting of \( n \) cells, each one having sides of length one. The top part of the figure below shows \( B_9 \).

\[ \begin{array}{cccccccccccc}
\text{ } & \text{ } & \text{ } & \text{ } & \text{ } & \text{ } & \text{ } & \text{ } & \text{ } & \text{ } & \text{ } & \text{ } \\
\text{ } & \text{ } & \text{ } & \text{ } & \text{ } & \text{ } & \text{ } & \text{ } & \text{ } & \text{ } & \text{ } & \text{ } \\
\text{ } & \text{ } & \text{ } & \text{ } & \text{ } & \text{ } & \text{ } & \text{ } & \text{ } & \text{ } & \text{ } & \text{ } \\
\text{ } & \text{ } & \text{ } & \text{ } & \text{ } & \text{ } & \text{ } & \text{ } & \text{ } & \text{ } & \text{ } & \text{ } \\
\text{ } & \text{ } & \text{ } & \text{ } & \text{ } & \text{ } & \text{ } & \text{ } & \text{ } & \text{ } & \text{ } & \text{ } \\
\text{ } & \text{ } & \text{ } & \text{ } & \text{ } & \text{ } & \text{ } & \text{ } & \text{ } & \text{ } & \text{ } & \text{ } \\
\text{ } & \text{ } & \text{ } & \text{ } & \text{ } & \text{ } & \text{ } & \text{ } & \text{ } & \text{ } & \text{ } & \text{ } \\
\text{ } & \text{ } & \text{ } & \text{ } & \text{ } & \text{ } & \text{ } & \text{ } & \text{ } & \text{ } & \text{ } & \text{ } \\
\end{array} \]

\[ R \quad B \quad G \]

You have an unlimited supply of bricks, which are of the following types (see the bottom part of the figure above):
There are red (R) and blue (B) bricks, both of which are 1 × 1 cells. We refer to these bricks as squares.

- There are green (G) bricks, which are 1 × 2 cells. We refer to these as dominoes.

A tiling of the board $B_n$ is a placement of bricks on the board such that
- the bricks exactly cover $B_n$ and
- no two bricks overlap.

In a tiling, a color can be used more than once and some colors may not be used at all. The figure below shows an example of a tiling of $B_9$.

```
G B B R B G R
```

Let $T_n$ be the number of different tilings of the board $B_n$.

- Determine $T_1$, $T_2$, and $T_3$.
- For any integer $n \geq 1$, express $T_n$ in terms of numbers that appear in this assignment.

**Solution:** For $n = 1$, we have the board $B_1$ consisting of one cell. There are two ways to tile this board: $R$ and $B$. Thus, $T_1 = 2$.

For $n = 2$, we have the board $B_2$ consisting of two cells. There are five ways to tile this board: $RR$, $RB$, $BR$, $BB$, and $G$. Thus, $T_2 = 5$.

For $n = 3$, we have the board $B_3$ consisting of three cells. There are twelve ways to tile this board:
- $RRR$, $BBB$,
- $RRB$, $RBR$, $BRR$, $RBB$, $BRB$, $BBR$,
- $GR$, $GB$, $RG$, $BG$.

Thus, $T_3 = 12$.

For the second part of the question, we are going to derive a recurrence for $T_n$. Assume $n \geq 3$. Any tiling of the board $B_n$ is of one of the following three types:
- The leftmost brick is a red square. Any such tiling is of the form $R$ followed by an arbitrary tiling of the board $B_{n-1}$. The number of such tilings is equal to $T_{n-1}$.
- The leftmost brick is a blue square. Any such tiling is of the form $B$ followed by an arbitrary tiling of the board $B_{n-1}$. The number of such tilings is equal to $T_{n-1}$.
The leftmost brick is a green domino. Any such tiling is of the form $G$ followed by an arbitrary tiling of the board $B_{n-2}$. The number of such tilings is equal to $T_{n-2}$.

Since these three types are pairwise disjoint, we conclude that, for $n \geq 3$,

$$T_n = 2 \cdot T_{n-1} + T_{n-2}.$$ 

The base cases are given by $T_1 = 2$ and $T_2 = 5$.

This is the same recurrence as in Question 4, but it has different base cases. We compare the numbers $a_n$ and $T_n$:

<table>
<thead>
<tr>
<th>$a_0$</th>
<th>$a_1$</th>
<th>$a_2$</th>
<th>$a_3$</th>
<th>$a_4$</th>
<th>$a_5$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td>5</td>
<td>12</td>
<td>29</td>
</tr>
<tr>
<td></td>
<td>$T_1$</td>
<td>$T_2$</td>
<td>$T_3$</td>
<td>$T_4$</td>
<td></td>
</tr>
</tbody>
</table>

From this table, we see that the $T_n$’s are a shifted version of the $a_n$’s. That is, for each $n \geq 1$, we have

$$T_n = a_{n+1}.$$ 

If you want to be formal, you prove this by induction. But in this case, it is obvious and, therefore, no formal proof is needed.

**Question 6:** In this question, we use the notation of Question 5. Let $n \geq 1$ be an integer and consider the $1 \times (2n + 1)$ board $B_{2n+1}$. We number the cells of this board, from left to right, as $1, 2, 3, \ldots, 2n + 1$.

- Determine the number of tilings of the board $B_{2n+1}$ in which the rightmost square is at position 1.

- Let $k$ be an integer with $1 \leq k \leq n$. Determine the number of tilings of the board $B_{2n+1}$ in which the rightmost square is at position $2k + 1$.

- Use the results of the above two parts to prove that

$$T_{2n+1} = 2 + 2 \sum_{k=1}^{n} T_{2k}.$$ 

**Solution:** For the first part of the question, the rightmost square is at position 1. There are two choices for this square: It is either $R$ or $B$. The positions $2, 3, \ldots, 2n + 1$ (there are an even number of them) must be tiled using dominoes ($G$); there is one way to do this. Thus, the answer to this part is 2.

For the second part, the rightmost square is at position $2k + 1$: 
• There are two choices for the square at position \(2k + 1\): \(R\) or \(B\).

• The positions \(2k + 2, \ldots, 2n + 1\) (there are an even number of them) must be tiled using dominoes \((G)\); there is one way to do this. (Note: If \(k = n\), this part is empty. Still, there is one way to tile an empty board.)

• The positions \(1, 2, \ldots, 2k\) contain an arbitrary tiling of the board \(B_{2k}\). There are \(T_{2k}\) many such tilings.

By the Product Rule, the answer to this part of the question is \(2 \cdot T_{2k}\).

For the third part: By definition, the number of tilings of the board \(B_{2n+1}\) is equal to \(T_{2n+1}\). We are going to divide all these tilings into groups, based on the location of the rightmost square.

• Since the board \(B_{2n+1}\) has an odd length, any tiling must contain at least one square.

• In any tiling, the rightmost square must be at an odd position.

The total number of tilings of \(B_{2n+1}\) is equal to

\[
\sum_{k=0}^{n} \text{number of tilings in which the rightmost square is at position } 2k + 1.
\]

From the first two parts of the question, this summation is equal to

\[
2 + \sum_{k=1}^{n} 2 \cdot T_{2k}.
\]

We conclude that

\[
T_{2n+1} = 2 + 2 \sum_{k=1}^{n} T_{2k}.
\]

**Question 7:** In this question, we use the notation of Question 5. Let \(n \geq 1\) be an integer and consider the \(1 \times n\) board \(B_n\).

• Consider strings consisting of characters, where each character is \(S\) or \(D\). Let \(k\) be an integer with \(0 \leq k \leq \lfloor n/2 \rfloor\). Determine the number of such strings of length \(n - k\), that contain exactly \(k\) many \(D\)’s.

**Hint:** This is a very easy question!
Let $k$ be an integer with $0 \leq k \leq \lfloor n/2 \rfloor$. Determine the number of tilings of the board $B_n$ that use exactly $k$ dominoes.

**Hint:** How many bricks are used for such a tiling? In the first part, imagine that $S$ stands for “square” and $D$ stands for “domino”.

- Use the results of the previous part to prove that

$$T_n = \sum_{k=0}^{\lfloor n/2 \rfloor} \binom{n-k}{k} \cdot 2^{n-2k}.$$ 

**Solution:** For the first part: This is the same as counting binary strings of length $n - k$ that have exactly $k$ many 1’s. The answer is

$$\binom{n-k}{k}.$$  

For the second part: We consider tilings of the board $B_n$ that use exactly $k$ dominoes ($G$).

- $B_n$ has length $n$.
- The total length of the $k$ dominoes is $2k$.
- The remaining length, which is $n - 2k$, must be covered by $n - 2k$ squares.
- Conclusion: Any such tiling uses $k$ dominoes and $n - 2k$ squares. In total, the tiling uses $k + (n - 2k) = n - k$ bricks.

To determine the number such tilings, we will use the Product Rule:

- Write down a string of length $n - k$ that contains $k$ many $D$’s and $n - 2k$ many $S$’s. There are $\binom{n-k}{k}$ ways to do this.
- Replace each $D$ by a dominoe ($G$). There is one way to do this.
- Replace each $S$ by either a red square ($R$) or a blue square ($B$). There are $2^{n-2k}$ ways to do this.

By the Product Rule, the number of tilings of the board $B_n$ that use exactly $k$ dominoes is equal to

$$\binom{n-k}{k} \cdot 2^{n-2k}.$$ 

For the third part: By definition, the number of tilings of the board $B_n$ is equal to $T_n$. We are going to divide all these tilings into groups, based on the number of dominoes. Denote
the number of dominoes by \( k \). Obviously, \( k \geq 0 \). What is the largest possible value for \( k \):
The total length of the \( k \) dominoes is \( 2k \), which must be at most \( n \). In other words, \( 2k \leq n \),
which is equivalent to \( k \leq \lfloor n/2 \rfloor \).

This gives:

\[
T_n = \text{number of tilings of } B_n \\
= \sum_{k=0}^{\lfloor n/2 \rfloor} \text{number of tilings of } B_n \text{ that use } k \text{ dominoes} \\
= \sum_{k=0}^{\lfloor n/2 \rfloor} \binom{n-k}{k} \cdot 2^{n-2k}.
\]

**Question 8:** The few of you who come to class will remember that Elisa Kazan\(^1\) loves to
drink cider. On Saturday night, Elisa goes to her neighborhood pub and runs the following
recursive algorithm, which takes as input an integer \( n \geq 1 \):

**Algorithm** `ElisaDrinksCider(n)`:

```
if n = 1
    then drink one pint of cider
else if n is even
    then `ElisaDrinksCider(n/2)`;
        drink one pint of cider;
        `ElisaDrinksCider(n/2)`
else drink one pint of cider;
    `ElisaDrinksCider(n - 1)`;
        drink one pint of cider
endif
endif
```

For any integer \( n \geq 1 \), let \( P(n) \) be the number of pints of cider that Elisa drinks when
running algorithm `ElisaDrinksCider(n)`. Determine the value of \( P(n) \).

**Solution:** Since the algorithm is recursive, we are going to derive a recurrence for the
function \( P(n) \):

- If \( n = 1 \), Elisa drinks one pint; thus, \( P(1) = 1 \).

- If \( n \geq 2 \) and \( n \) is even: Elisa drinks \( P(n/2) \) pints, followed by one pint, followed by
  \( P(n/2) \) pints. Thus,

\[
P(n) = 1 + 2 \cdot P(n/2).
\]

\(^1\)President of the Carleton Computer Science Society
• If \( n \geq 3 \) and \( n \) is odd: Elisa drinks one pint, followed by \( P(n - 1) \) pints, followed by one pint. Thus,
\[
P(n) = 2 + P(n - 1).
\]
By looking at \( P(n) \) for some small values of \( n \), you will guess that, for \( n \geq 1 \),
\[
P(n) = 2n - 1.
\]
We verify using induction that this guess is correct.

The base case is when \( n = 1 \). In this case, the left-hand side is \( P(1) \), which is 1, whereas the right-hand side is \( 2 \cdot 1 - 1 \), which is also 1. Thus, the claim is true if \( n = 1 \).

For the induction step, let \( n \geq 2 \) and assume that the claim is true for all values that are strictly smaller than \( n \).

• Assume \( n \) is even. We know that
\[
P(n) = 1 + 2 \cdot P(n/2).
\]
Since \( n/2 < n \), the assumption implies that
\[
P(n/2) = 2 \cdot (n/2) - 1 = n - 1.
\]
This gives
\[
P(n) = 1 + 2 \cdot P(n/2) = 1 + 2(n - 1) = 2n - 1.
\]
• Assume \( n \) is odd. We know that
\[
P(n) = 2 + P(n - 1).
\]
Since \( n - 1 < n \), the assumption implies that
\[
P(n - 1) = 2(n - 1) - 1 = 2n - 3.
\]
This gives
\[
P(n) = 2 + P(n - 1) = 2 + (2n - 3) = 2n - 1.
\]

**Question 9:** Let \( n \geq 1 \) be an integer and consider a set \( S \) consisting of \( n \) points in \( \mathbb{R}^2 \). Each point \( p \) of \( S \) is given by its \( x \)- and \( y \)-coordinates \( p_x \) and \( p_y \), respectively. We assume that no two points of \( S \) have the same \( x \)-coordinate and no two points of \( S \) have the same \( y \)-coordinate.

A point \( p \) of \( S \) is called **maximal** in \( S \) if there is no point in \( S \) that is to the north-east of \( p \), i.e.,
\[
\{ q \in S : q_x > p_x \text{ and } q_y > p_y \} = \emptyset.
\]
The figure below shows an example, in which the ●-points are maximal and the ×-points are not maximal. Observe that, in general, there is more than one maximal element in \( S \).
Describe a recursive algorithm \textsc{MaxElem}(S, n) that has the same structure as algorithms \textsc{MergeSort} and \textsc{ClosestPair} that we have seen in class, and does the following:

\textbf{Input:} A set \( S \) of \( n \geq 1 \) points in \( \mathbb{R}^2 \), in sorted order of their \( x \)-coordinates. You may assume that \( n \) is a power of two.

\textbf{Output:} All maximal elements of \( S \), in sorted order of their \( x \)-coordinates.

The running time of your algorithm must be \( O(n \log n) \). Explain why your algorithm runs in \( O(n \log n) \) time. You may use any result that was proven in class.

\textbf{Solution:} The algorithm will be recursive. The base case is when \( n = 1 \), i.e., the set \( S \) consists of only one point. Since this point is maximal in \( S \), the algorithm returns this point.

Assume that \( n \geq 2 \). Here is the main approach:

- Let \( \ell \) be a vertical line that divides the set \( S \) into two subsets, each of size \( n/2 \).

- Let \( S_1 \) be the set of all points of \( S \) that are to the left of the line \( \ell \). Run algorithm \textsc{MaxElem}(\( S_1, n/2 \)). This gives as output the set, say \( M_1 \), of all maximal elements in \( S_1 \). The set \( M_1 \) is returned in sorted \( x \)-order. These are the \( \bullet \)-points to the left of \( \ell \) in the figure below.

- Let \( S_2 \) be the set of all points of \( S \) that are to the right of the line \( \ell \). Run algorithm \textsc{MaxElem}(\( S_2, n/2 \)). This gives as output the set, say \( M_2 \), of all maximal elements in \( S_2 \). The set \( M_2 \) is returned in sorted \( x \)-order. These are the \( \bullet \)-points to the right of \( \ell \) in the figure below.
• Each point of $M_2$ is maximal in the set $S_2$. Since $S_1$ is to the left of $\ell$, each point of $M_2$ is maximal in the entire set $S$. Thus, the points of $M_2$ belong to the output.

• Each point of $M_1$ is maximal in the set $S_1$, but not necessarily in the entire set $S$. Let $q$ be the leftmost point of $M_2$. Then a point $p$ of $M_1$ is maximal in the entire set $S$ if and only $p$ is above $q$.

• From this, we can see how to obtain the final output: It consists of all points of $M_1$ that are above $q$, followed by all points of $M_2$.

This leads to the following algorithm in pseudocode:

<table>
<thead>
<tr>
<th>Algorithm MaxElem($S, n$):</th>
</tr>
</thead>
<tbody>
<tr>
<td>// $S$ is a set of $n$ points, sorted by $x$-coordinates</td>
</tr>
<tr>
<td>if $n = 1$</td>
</tr>
<tr>
<td>then return the only point of $S$</td>
</tr>
<tr>
<td>else $S_1 =$ first $n/2$ points of $S$; $S_2 =$ last $n/2$ points of $S$; $M_1 =$ MaxElem($S_1, n/2$); $M_2 =$ MaxElem($S_2, n/2$); $q =$ first point in $M_2$; $M =$ empty list; add to $M$ all points $p$ of $M_1$ for which $p_y &gt; q_y$; add all points of $M_2$ at the end of $M$; return $M$</td>
</tr>
<tr>
<td>endif</td>
</tr>
</tbody>
</table>

Let $T(n)$ be the running time of algorithm MaxElem on an input of size $n$. Then $T(1)$ is some constant. Assume that $n \geq 2$.

• There are two recursive calls, each on a set of size $n/2$. The total time for these recursive calls is $2 \cdot T(n/2)$.

• Besides the recursive calls, the algorithm spends $O(n)$ time, because it obtains $S_1$ and $S_2$ by traversing $S$, and it obtains $M$ by traversing $M_1$ and $M_2$.

Thus, the running time satisfies the recurrence

$$T(n) = O(n) + 2 \cdot T(n/2).$$

We have seen in class that this recurrence solves to $T(n) = O(n \log n)$. 

14