COMP 2804 — Solutions Assignment 2

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

Solution:

- Name: James Bond
- Student number: 007

Question 2: The Fibonacci numbers are defined as follows: $f_0 = 0$, $f_1 = 1$, and $f_n = f_{n-1} + f_{n-2}$ for $n \geq 2$. Prove each of the following three claims:

- For each $n \geq 0$, f_{3n} is even.

  Solution: The proof is by induction. If $n = 0$, then $f_{3n} = f_0 = 0$, which is even. Let $n \geq 0$ and assume that f_{3n} is even. Then

  $$\begin{aligned} f_{3(n+1)} &= f_{3n+3} \\ &= f_{3n+2} + f_{3n+1} \\ &= (f_{3n+1} + f_{3n}) + f_{3n+1} \\ &= 2 \cdot f_{3n+1} + f_{3n}. \end{aligned}$$

  Obviously, $2 \cdot f_{3n+1}$ is even. By our assumption, f_{3n} is even. Since the sum of two even integers is even, it follows that $f_{3(n+1)}$ is even.

- For each $n \geq 0$, f_{3n+1} is odd.

  Solution: The proof is by induction. If $n = 0$, then $f_{3n+1} = f_1 = 1$, which is odd. Let $n \geq 0$ and assume that f_{3n+1} is odd. Then

  $$\begin{aligned} f_{3(n+1)+1} &= f_{3n+4} \\ &= f_{3n+3} + f_{3n+2} \\ &= f_{3(n+1)} + (f_{3n+1} + f_{3n}). \end{aligned}$$

  We have proved above that both $f_{3(n+1)}$ and f_{3n} are even. By our assumption, f_{3n+1} is odd. Since the sum of two even integers and one odd integer is odd, it follows that $f_{3(n+1)+1}$ is odd.

- For each $n \geq 0$, f_{3n+2} is odd.

  Solution: The proof is by induction. If $n = 0$, then $f_{3n+2} = f_2 = f_1 + f_0 = 1 + 0 = 1$, which is odd. Let $n \geq 0$ and assume that f_{3n+2} is odd. Then

  $$\begin{aligned} f_{3(n+1)+2} &= f_{3n+5} \\ &= f_{3n+4} + f_{3n+3} \\ &= f_{3(n+1)+1} + f_{3(n+1)}. \end{aligned}$$

  We have proved above that $f_{3(n+1)+1}$ is odd and $f_{3(n+1)}$ is even. Since the sum of an even integer and an odd integer is odd, it follows that $f_{3(n+1)+2}$ is odd.

Question 3: Let $\varphi = \frac{1+\sqrt{5}}{2}$, $\psi = \frac{1-\sqrt{5}}{2}$, and $n \geq 0$ be an integer. We have seen in class that

$$\frac{\varphi^n - \psi^n}{\sqrt{5}} \tag{1}$$

is equal to the n-th Fibonacci number f_n. Since the Fibonacci numbers are obviously integers, the number in (1) is an integer as well.

Prove that the number in (1) is a rational number using only Newton's Binomial Theorem.

Solution: First recall from COMP 1805 that $\sqrt{5}$ is not a rational number.

Using Newton's Binomial Theorem, we get

$$\varphi^n = \frac{1}{2^n}\left(1+\sqrt{5}\right)^n = \frac{1}{2^n}\sum_{k=0}^{n}\binom{n}{k}\left(\sqrt{5}\right)^k$$

and

$$\psi^n = \frac{1}{2^n}\left(1-\sqrt{5}\right)^n = \frac{1}{2^n}\sum_{k=0}^{n}\binom{n}{k}\left(-\sqrt{5}\right)^k.$$

By combining them, we get

$$\varphi^n - \psi^n = \frac{1}{2^n}\sum_{k=0}^{n}\binom{n}{k}\left(\left(\sqrt{5}\right)^k - \left(-\sqrt{5}\right)^k\right).$$

For even k,

$$\left(\sqrt{5}\right)^k - \left(-\sqrt{5}\right)^k = 0,$$

whereas for odd k,

$$\left(\sqrt{5}\right)^k - \left(-\sqrt{5}\right)^k = 2\left(\sqrt{5}\right)^k = 2\sqrt{5}\cdot 5^{(k-1)/2}.$$

It follows that

$$\frac{\varphi^n - \psi^n}{\sqrt{5}} = \frac{2}{2^n}\sum_{k}\binom{n}{k}5^{(k-1)/2},$$

where the sum is over all odd integers k with $1 \leq k \leq n$. If k is odd, then $(k-1)/2$ is an integer and, therefore, $5^{(k-1)/2}$ is an integer as well. Therefore, for each odd k, the term

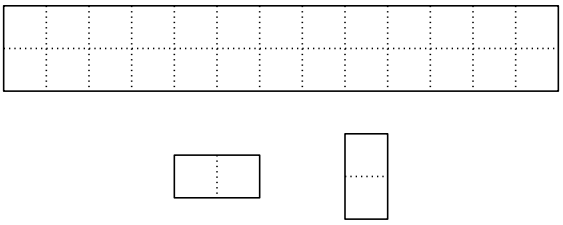
$$\frac{2}{2^n}\binom{n}{k}5^{(k-1)/2}$$

is a rational number (i.e., the quotient of two integers). Thus,

$$\frac{\varphi^n - \psi^n}{\sqrt{5}}$$

is the sum of rational numbers, which is again a rational number.

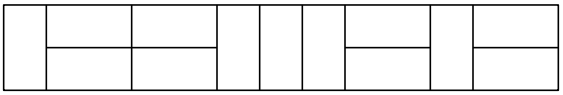
Question 4: Let $n \geq 1$ be an integer and consider a $2 \times n$ board B_n consisting of $2n$ cells, each one having sides of length one. The top part of the figure below shows B_{13}.



A *brick* is a horizontal or vertical board consisting of 2 cells; see the bottom part of the figure above. 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.

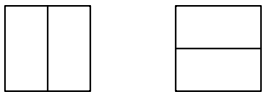
The figure below shows a tiling of B_{13}.



For $n \geq 1$, let a_n be the number of different tilings of the board B_n. Determine the value of a_n, i.e., express a_n in terms of numbers that we have seen in class. Justify your answer.

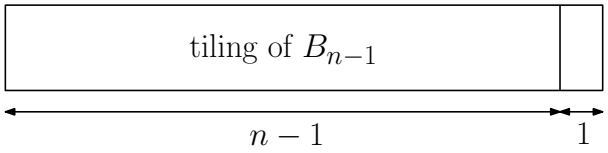
Solution: Since B_1 is a board consisting of two cells that are stacked on top of each other, there is only one way to tile B_1. Thus, $a_1 = 1$.

The figure below shows that the board B_2 can be tiled in two ways. Thus, $a_2 = 2$.



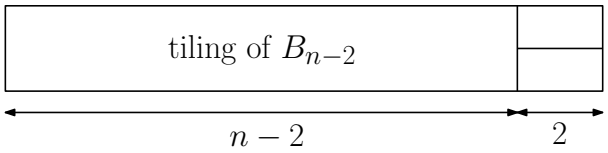
Let $n \geq 3$. Consider all a_n many possible tilings of the board B_n. These tilings can be divided into two groups:

- The first group contains all tilings whose rightmost brick is vertical.



  By deleting the rightmost brick from each tiling in this group, we obtain all tilings of B_{n-1}. Thus, there are a_{n-1} many tilings of B_n in this group.

- The second group contains all tilings that have two horizontal bricks at the far right.



By deleting the two rightmost bricks from each tiling in this group, we obtain all tilings of B_{n-2}. Thus, there are a_{n-2} many tilings of B_n in this group.

We conclude that we have divided all a_n tilings of B_n into two groups, one of size a_{n-1} and the other of size a_{n-2}. It follows that

$$a_n = a_{n-1} + a_{n-2} \text{ for } n \geq 3.$$

This is the Fibonacci recurrence, except that the base cases are different. The Fibonacci sequence starts with

$$f_0 = 0, f_1 = 1, f_2 = 1, f_3 = 2, f_4 = 3, f_5 = 5, f_6 = 8,$$

whereas the a_n-sequence starts with

$$a_1 = 1, a_2 = 2, a_3 = 3, a_4 = 5, a_5 = 8.$$

We conclude that

$$a_n = f_{n-1} \text{ for } n \geq 1.$$

Question 5: We consider strings of n characters, each character being a, b, c, or d, that contain an even number of as. (Recall that 0 is even.) Let E_n be the number of such strings. Prove that for any integer $n \geq 1$,

$$E_{n+1} = 2 \cdot E_n + 4^n.$$

Solution: We say that a string is *valid* if each of its characters is a, b, c, or d, and the string contains an even number of as.

Consider all valid strings of length $n+1$. There are E_{n+1} many such strings. We divide them into four groups:

- Strings that start with b. If we remove the first character from each such string, we get all valid strings of length n. Therefore, there are E_n many strings in this group.
- Strings that start with c. By the same argument, there are E_n many strings in this group.
- Strings that start with d. By the same argument, there are E_n many strings in this group.

- Strings that start with a. If we remove the first character from each such string, we get all strings of length n that have an *odd* number of as. How many of these are there?
  - The total number of strings of length n is 4^n.
  - The number of strings of length n with an even number of as is E_n.
  - Therefore, the number of strings of length n with an odd number of as is $4^n - E_n$.

We conclude that we have divided all valid strings of length $n+1$ into four groups; three of them have size E_n, and one of them has size $4^n - E_n$. It follows that

$$E_{n+1} = 3 \cdot E_n + (4^n - E_n) = 2 \cdot E_n + 4^n.$$

Question 6: Let a_n be the number of bitstrings that contain 000. Prove that for $n \geq 4$,

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

Solution: We say that a bitstring is *valid* if it contains 000.

Consider all valid strings of length n. There are a_n many such strings. We divide them into four groups:

- Strings that start with 1. If we remove the first bit from each such string, we get all valid strings of length $n-1$. Therefore, there are a_{n-1} many strings in this group.
- Strings that start with 01. If we remove the first two bits from each such string, we get all valid strings of length $n-2$. Therefore, there are a_{n-2} many strings in this group.
- Strings that start with 001. If we remove the first three bits from each such string, we get all valid strings of length $n-3$. Therefore, there are a_{n-3} many strings in this group.
- Strings that start with 000. Note that all these strings contain the required substring 000 at the far left. Therefore, if we remove the first three bits from each such string, we get all bitstrings of length $n-3$. Therefore, there are 2^{n-3} many strings in this group.

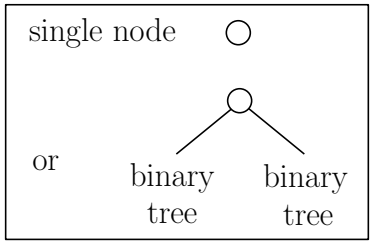
We conclude that we have divided all valid strings of length n into four groups; one of size a_{n-1}, one of size a_{n-2}, one of size a_{n-3}, and one of size 2^{n-3}. It follows that

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

Question 7: A binary tree is

- either one single node

- or a node whose left subtree is a binary tree and whose right subtree is a binary tree.



Prove that any binary tree with n leaves has exactly $2n - 1$ nodes.

Solution: The proof is by induction on the number n of leaves in the binary tree.

The base case is when $n = 1$. In this case, the tree consists of one single node (which is a leaf). Thus, the tree has exactly 1 node, which is equal to $2n - 1$.

Let $n \geq 2$ and assume the claim is true for all binary trees with less than n leaves. Consider a binary tree T with n leaves. Since $n \geq 2$, T has a left subtree and a right subtree. Let m denote the number of leaves in the left subtree. Then the right subtree has $n - m$ leaves.

Since $m < n$, the induction hypothesis implies that the left subtree has $2m - 1$ nodes. Since $n - m < n$, the induction hypothesis implies that the right subtree has $2(n - m) - 1$ nodes. The number of nodes in T is equal to the sum of 1 (the root), the number of nodes in the left subtree, and the number of nodes in the right subtree. Therefore, the number of nodes in T is equal to

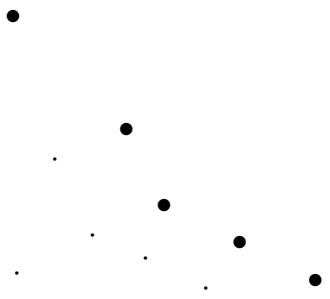
$$1 + (2m - 1) + (2(n - m) - 1) = 2n - 1.$$

Question 8: Let S be a set of n points in the plane. 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 MaxElem(S) that has the same basic structure as algorithm MergeSort that we have seen in class, and that does the following:

Input: A set S of n points in the plane, in sorted order from left to right.

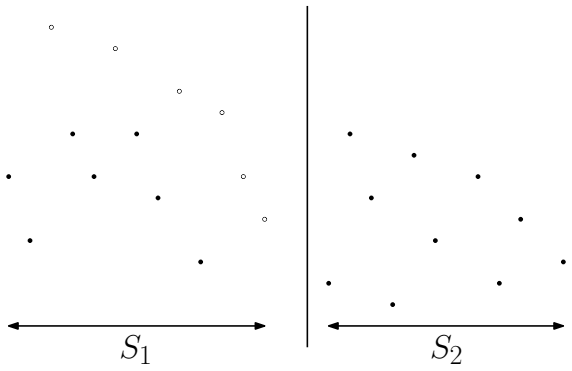
Output: All maximal elements of S, in sorted order from left to right.

The running time $T(n)$ of your algorithm must be $O(n \log n)$. Derive a recurrence for $T(n)$. (You do not have to solve the recurrence, because we have done that in class.) You may assume that n is a power of 2.

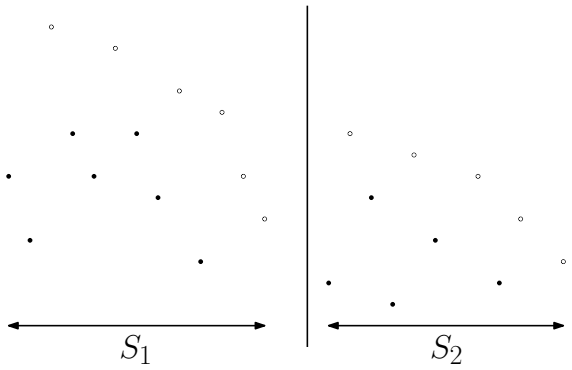
Solution: Since the algorithm is recursive, we need a base case. We take the base case to be when $n = 1$. In this case, the input consists of one single point, which is obviously maximal. So we just return this single point.

Let $n \geq 2$. Consider an input set S, in sorted order from left to right. We split S in the middle and run the algorithm recursively on each part:

- Let S_1 be the set of the first $n/2$ points in S. Run algorithm MaxElem(S_1) and let M_1 be the output.



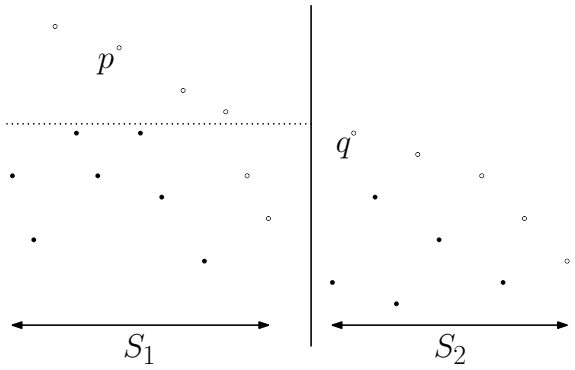
- S_2 is the set of the last $n/2$ points in S. Run algorithm MaxElem(S_2) and let M_2 be the output.



It remains to specify the merge step. Observe the following:

- M_2 contains all maximal elements in S_2. Since M_2 is to the right of S_1, each point in M_2 is maximal in the entire set S.
- M_1 contains all maximal elements in S_1. However, points in M_1 may not be maximal in the entire set S. Let p be an arbitrary element in M_1, and let q be the leftmost element in M_2. Then

$$p \text{ is maximal in } S \text{ if and only if } p_y > q_y.$$



From this, it follows that we can do the merge step in the following way:

- Take the leftmost point q in M_2.
- Walk along M_1 and add all points whose y-coordinate is larger than that of q to a list M.
- Add all points in M_2 to the list M.
- Return the list M. This list contains all maximal elements in the entire set S.

The running time $T(n)$ is the sum of the following:

- The time to split S into two sets; this can be done in $O(n)$ time by scanning S.
- The time for the recursive call MaxElem(S_1); this takes $T(n/2)$ time.
- The time for the recursive call MaxElem(S_2); this takes $T(n/2)$ time.
- The time for the merge step; this can be done in $O(n)$ time.

Thus, we arrive at the recurrence

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

which is the same as the merge-sort recurrence that we have seen in class.

Question 9: For an integer $n \geq 1$, draw n straight lines, such that no two of them are parallel and no three of them intersect in one single point. These lines divide the plane into regions (some of which are bounded and some of which are unbounded). Denote the number of these regions by R_n.

Derive a recurrence for the numbers R_n and use it to prove that for $n \geq 1$,

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

Solution: The base case is when $n = 1$. In this case, we have one line, which obviously divides the plane into two regions. Thus,

$$R_1 = 2.$$

Let $n \geq 2$ and consider n lines $L_1, L_2, \ldots, L_n$.

- We start by drawing the lines $L_1, L_2, \ldots, L_{n-1}$. At this moment, the number of regions is R_{n-1}.
- Now we add the line L_n. This line intersects each of the first $n-1$ lines. Thus, there are $n-1$ intersections between L_n and the lines we have already drawn. It follows that L_n goes through n regions (one more than the number of intersections), and each such region is cut into two. It follows that, when adding L_n, the number of regions increases by n.

Thus, we obtain the recurrence

$$R_n = R_{n-1} + n.$$

It remains to prove that

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

We do this by induction. If $n = 1$, then $R_1 = 2$ and $1 + n(n+1)/2 = 2$ as well, proving the base case.

Let $n \geq 2$, and assume the claim is true for $n-1$, i.e., assume that

$$R_{n-1} = 1 + (n-1)n/2.$$

We have to show that

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

This follows by applying the recurrence and the assumption:

$$\begin{aligned} R_n &= R_{n-1} + n \\ &= 1 + (n-1)n/2 + n \\ &= 1 + n(n+1)/2. \end{aligned}$$