6. Math 235 Fall 2023, Worksheet 6: Invariants and processes

On this worksheet, we will discuss some techniques for analyzing *processes*. In particular, we will present *invariants*, *monovariants* (aka *semi-invariants*) and *working backwards*. As with many other techniques, these are very simple ideas, but can be used in many elegant and creative ways to obtain fairly surprising results.

These ideas appear in several texts on problem-solving: For example, invariants and monovariants appear in [GelAnd17, §1.5], [Engel98, Chapter 1], [Zeitz17, §3.4], [StaRik08, Chapter 10] and [Grinbe20, Chapter 8], whereas working backwards is discussed in [Engel98, §14.3 and elsewhere]. In particular, I recommend [Grinbe20, Chapter 8] as a source of extra examples (as I have been deliberately avoiding a repetition of the examples from [Grinbe20, Chapter 8] in the present worksheet).

As before, $\mathbb{N}$ means the set $\{0, 1, 2, \ldots\}$.

6.1. Processes

6.1.1. What are they?

Let me first informally define the concepts I will be using. For more examples, see [Grinbe20, Chapter 8].

Mathematicians frequently study *processes* – i.e., roughly speaking, objects that change with time. For example, imagine an integer that increases by 1 every minute. For another example, imagine a tuple of integers that is repeatedly extended by adding the last two of its entries together and inserting the sum at the end of the tuple. (The former process is a clock, while the latter is a way to compute the Fibonacci numbers, at least if you start with the tuple $(0, 1)$.) Processes do not have to be deterministic; for example, a process can consist of an integer that increases by 4 or 6 every minute, where the decision (whether to increase it by 4 or by 6) is made by whoever is performing the process. Thus, we have given three examples of processes; you will soon see many more.

We restrict ourselves to processes that have a precisely defined set of *states* (which the object can have at any given time) and a precisely defined set of *moves* (i.e., ways in which the object can change from one moment to the next; i.e., allowable transitions between states). The first of our three above examples is thus a process whose states are the integers, and whose moves are $i \mapsto i + 1$ (that is, "replace i by $i + 1$") for each integer i. Our second example is a process whose states are the tuples $(a_1, a_2, \ldots, a_k)$ of integers, and whose moves are $(a_1, a_2, \ldots, a_k) \mapsto (a_1, a_2, \ldots, a_k, a_{k-1} + a_k)$. Note that such moves are only allowed when $k \geq 2$, since otherwise $a_{k-1} + a_k$ does not make sense. Thus, if our state is a 0-tuple or a 1-tuple, then we cannot make a move. States from which no moves are possible are called *end states* or *final states*. Finally, our third example is a process whose states are the integers, and whose moves are $i \mapsto i + 4$ and $i \mapsto i + 6$ for each integer i.

6.1.2. Questions about processes

Whenever you have a process, you can ask several natural questions:

- Given two states a and b, is it possible to reach b from a by making (the right) moves?
- What states can be reached starting from a given state a ?
- Can the process go on forever, or will it necessarily reach a final state after a (finite) number of moves?
- Does the process exhibit periodic behavior (i.e., does it return to the starting state after some number of moves)?
- Is the process reversible? I.e., can we reconstruct a state from the next state?
- What things (quantities, statements) remain unchanged whenever we make a move (and thus never change during the process)? These things are called the *invariants* (or *conserved quantities*) of the process.
- What things change only in one direction (i.e., only get smaller or only get larger) whenever we make a move? These things are called the *monovariants* of the process.

These questions are related, and solving one can often help in solving another. For instance, our first example of a process above has an obvious monovariant (the number i itself, which gets larger and larger as we keep increasing it by 1), and this immediately entails that the process cannot exhibit periodic behavior. Likewise, our third example of a process has an easily-found invariant: the parity of the number i, that is, the remainder of i upon division by 2 (since this is clearly preserved if we increase i by 4 or by 6). Thus, all states reachable from 2 (using the moves of this process) are even integers (although not every even integer is reachable from 2). Hence, it is impossible to reach the state 7 from the state 2 by making moves of this process.

6.1.3. The cave-man's Euclidean algorithm

The examples so far were toy examples. A more interesting example of a process is the famous *Euclidean algorithm* for computing the greatest common divisor $\gcd(a,b)$ of two nonnegative integers a and b. In the most elementary version of this algorithm[1], we begin with the pair (a,b) and progressively "simplify" it by subtracting the smaller of its entries from the larger[2], as long as the smaller entry is

[1] This might not be the version that you are most familiar with. In fact, this is a rather slow version, since it uses subtraction instead of division with remainder. But I think it is best for illustrative purposes.

[2] If the two entries are equal, we can freely choose which one to subtract from which.

nonzero. In other words, this is a process whose states are the pairs $(a, b) \in \mathbb{N} \times \mathbb{N}$, and whose moves are

$$(a, b) \mapsto (a - b,\ b) \qquad \text{if } a \geq b > 0$$

and

$$(a, b) \mapsto (a,\ b - a) \qquad \text{if } b \geq a > 0.$$

For example, starting with $(10, 14)$, it proceeds as follows:

$$(10, 14) \mapsto (10, 4) \mapsto (6, 4) \mapsto (2, 4) \mapsto (2, 2) \mapsto (2, 0)$$

(we could also go to $(0, 2)$ from $(2, 2)$ instead). The state $(2, 0)$ reached here is an end state, since no moves from it are possible. What do we notice about this process?

- A monovariant of this process is the sum $a + b$ of the two entries of the state (a, b). Indeed, this sum decreases with every move (since $(a - b) + b$ and $a + (b - a)$ are both smaller than $a + b$).
- Hence, the process cannot keep running forever. Indeed, the sum $a + b$ is a nonnegative integer, but a nonnegative integer can only decrease a finite number of times until it becomes negative (or non-integer). Thus, after a finite number of moves, we reach an end state.
- An invariant of this process is the greatest common divisor $\gcd(a, b)$ of the two entries of the state (a, b). Indeed, this gcd[3] remains unchanged under every move, since $\gcd(a - b, b) = \gcd(a, b)$ and $\gcd(a, b - a) = \gcd(a, b)$.
- From these observations, we can predict how the process will end: If we start our process with a given pair $(a, b) \in \mathbb{N} \times \mathbb{N}$, then it will eventually (after finitely many moves) reach an end state. This end state will have either the form $(d, 0)$ or the form $(0, d)$ (since an end state must have a zero entry), and moreover the gcd of this end state will equal the gcd of the original state (since the gcd is an invariant). That is, $\gcd(d, 0)$ (or, respectively, $\gcd(0, d)$) will equal $\gcd(a, b)$. Since both $\gcd(d, 0)$ and $\gcd(0, d)$ equal d, this means that d will equal $\gcd(a, b)$. Thus, our process computes the gcd of a and b.

Note that we have used both a monovariant (to prove that the process terminates) and an invariant (to identify the end state to which it leads). A similar analysis can be made for the "grown-up" Euclidean algorithm that uses division with remainder instead of subtraction.

[3] I abbreviate the word "greatest common divisor" as "gcd" here and in the following.

6.2. Invariants

We now come to more interesting problems. The first is a famous puzzle (the "chameleon problem"):

Exercise 6.2.1. An island has a population of chameleons. Chameleons come in three colors (red, green and blue), but can change these colors according to the following rule: When two chameleons of different colors meet, they both change to the third color. (For example, if a red chameleon meets a blue one, they both turn green.) This is the only situation in which chameleons change colors. (Some fine print for the pedantic reader: More than two chameleons never meet simultaneously. Also, a chameleon cannot have more than one color at a time.)

At the beginning, the island has 13 red, 15 green and 17 blue chameleons. Can it happen that, after a while, all of the chameleons have acquired the same color?

(Tournament of Towns 6.10)

Solution idea. As a warmup, let us solve an easier version of this exercise: What if the island starts out with 1 red, 4 green and 5 blue chameleons instead? Then, the answer is "yes". For example, we can first have a red-green pair meet and change colors, so that we obtain 0 red, 3 green and 7 blue chameleons. Then, a green-blue pair meets and turns red, so we obtain 2 red, 2 green and 6 blue chameleons. Then, two red-green pairs meet (independently of each other), and ultimately all chameleons become blue.

Unfortunately, this kind of reasoning can only prove a positive answer, and it becomes increasingly cumbersome as the chameleon population grows.

So let us return to the exercise with a fresh mind. Obviously we are dealing with a process. Its states are the possible distributions of colors on the island. For the sake of convenience, we will encode such distributions as triples $(a, b, c) \in \mathbb{N}^3$, where a is the number of red chameleons, b is the number of green chameleons and c is the number of blue chameleons.

Hence, the states of our process are the triples $(a, b, c) \in \mathbb{N}^3$, and the moves are

$$\begin{aligned}
(a,b,c) &\mapsto (a-1, b-1, c+2) && \text{if } a \geq 1 \text{ and } b \geq 1; \\
(a,b,c) &\mapsto (a-1, b+2, c-1) && \text{if } a \geq 1 \text{ and } c \geq 1; \\
(a,b,c) &\mapsto (a+2, b-1, c-1) && \text{if } b \geq 1 \text{ and } c \geq 1.
\end{aligned}$$

(These moves correspond to a red-green meeting, a red-blue meeting and a green-blue meeting, respectively.)

The end states are precisely the states in which all chameleons have the same color. Thus, we are wondering if we can reach an end state starting from the state $(13, 15, 17)$. Our warmup above shows that an end state can be reached from the state $(1, 4, 5)$ via the moves

$$(1,4,5) \mapsto (0,3,7) \mapsto (2,2,6) \mapsto (1,1,8) \mapsto (0,0,10).$$

The starting state $(13, 15, 17)$ does not succumb to such a simple move sequence, so we try to think more generally.

What invariants does our system have? Clearly, the total number of all chameleons on the island never changes. In other words, the sum $a + b + c$ of a state (a, b, c) is an invariant. This shows that any sequence of moves from $(13, 15, 17)$ to an end state must lead to an end state whose sum is $13 + 15 + 17 = 45$. But this does not rule out that such a sequence might exist.

However, there is another, subtler invariant that is helpful. Indeed, let us "work modulo 3", that is, consider the remainders of a, b, c upon division by 3. Recall that the remainder of a number n upon division by 3 is denoted by $n\%3$.

Modulo 3, there is no difference between $x - 1$ and $x + 2$. Hence, all three possible moves "look like" $(a, b, c) \mapsto (a - 1, b - 1, c - 1)$ modulo 3, in the sense that all three entries of a state decrease by 1 modulo 3 when we apply a move. Hence, the difference $a - b$ between the numbers of red and of green chameleons does not change modulo 3 (that is, its remainder $(a - b)\,\%3$ does not change) when we make a move.[4]

Hence, the remainder $(a - b)\,\%3$ constructed from a state (a, b, c) is an invariant of our process. Thus, for any state (a', b', c') that is reachable from (a, b, c), we will have

$$\left(a' - b'\right)\%3 = (a - b)\,\%3. \tag{1}$$

Similarly, we can see that any such state will satisfy

$$\left(b' - c'\right)\%3 = (b - c)\,\%3 \qquad \text{and} \tag{2}$$

$$\left(c' - a'\right)\%3 = (c - a)\,\%3. \tag{3}$$

Now, let us assume that we can reach an end state starting from a state (a, b, c). This end state must have two zero entries (since all chameleons must have the same color), so it has either the form $(d, 0, 0)$ or the form $(0, d, 0)$ or the form $(0, 0, d)$ for some $d \in \mathbb{N}$. Consider this d. Let us assume that it has the form $(d, 0, 0)$ (the other two cases are analogous). Thus, the equality (2) (applied to $(a', b', c') = (d, 0, 0)$) yields $(0 - 0)\,\%3 = (b - c)\,\%3$. Hence, $(b - c)\,\%3 = (0 - 0)\,\%3 = 0\%3 = 0$, so that $b - c \equiv 0 \bmod 3$. In other words, $b \equiv c \bmod 3$. Thus, two of the three numbers a, b, c are congruent modulo 3. Similarly, we can obtain the same conclusion in the other two cases.

Thus, we have shown that if we can reach an end state starting from a state (a, b, c), then two of the three numbers a, b, c are congruent modulo 3. Taking the contrapositive, we conclude that if a state (a, b, c) has no two entries that are congruent modulo 3, then we cannot reach an end state from it. Hence, in particular, we cannot reach an end state from the state $(13, 15, 17)$ (since no two of the entries $13, 15, 17$ are congruent modulo 3). In other words, in Exercise 6.2.1,

[4] If you don't like the above handwaving, you can check this directly: For instance, when we make a move of the form $(a, b, c) \mapsto (a - 1, b + 2, c - 1)$, the difference $a - b$ decreases by 3 (since it becomes $(a - 1) - (b + 2) = a - b - 3$) and thus its remainder $(a - b)\,\%3$ does not change. Similarly, the other two kinds of moves don't change this remainder either.

the chameleons will never all end up having the same color, no matter how they organize their meetings. □

Here is another illustrative exercise:

Exercise 6.2.2. You arrive at Heaven's gate, but it is closed. Three glowing numbers appear on the wall in front of you: a (the number of times you have used the axiom of choice), b (the number of times you have denoted a function by the term for its value) and c (the number of theorems you have cited without knowing their proofs). Before you proceed, you must play a game: You can pick any of the three numbers and decrease it by 1, unless it already equals 0. When you do this, the product of the other two numbers is added to the time (in years) that you have to spend in Purgatory. When all three numbers become 0, you serve your punishment, after which you are set free.

How should you play to minimize your time in Purgatory?

(Tournament of Towns 28.2.1)

Solution idea. For the sake of brevity, we encode the three numbers a, b, c on the wall as a triple (a, b, c). Furthermore, we define the *punishment* to be the number of Purgatory-years accumulated so far during the game. At the beginning of the game, this number is 0, but every move increases it by a certain product. You want to minimize the final punishment (i.e., the punishment accumulated by the end of the game).

For a simple example, let us try to play this game with $a = 2$ and $b = 3$ and $c = 1$. Here is one way to do this:

$$\begin{aligned}
(2,3,1) &\mapsto (2,2,1) && (\text{punishment increases by } 2 \cdot 1 = 2) \\
&\mapsto (2,2,0) && (\text{punishment increases by } 2 \cdot 2 = 4) \\
&\mapsto (2,1,0) && (\text{punishment increases by } 2 \cdot 0 = 0) \\
&\mapsto (1,1,0) && (\text{punishment increases by } 1 \cdot 0 = 0) \\
&\mapsto (0,1,0) && (\text{punishment increases by } 1 \cdot 0 = 0) \\
&\mapsto (0,0,0) && (\text{punishment increases by } 0 \cdot 0 = 0)\,.
\end{aligned}$$

Thus, we finish the game with a punishment of $2 + 4 + 0 + 0 + 0 + 0 = 6$. Try another way (e.g., starting by decreasing the 1 or the 2 instead of the 3). Surprisingly perhaps, you will see that the final punishment is 6 again!

Try a few more examples to check whether this might be a coincidence. What you might notice is that the final punishment will not just be independent of how you play, but in fact always equal to abc.

This looks like something that an invariant could be useful to prove. However, care must be taken in defining the process. It is tempting to let the states of the process to be the triples of the numbers on the wall; but this leads to no useful invariants, since these numbers eventually all become 0. Moreover, such a definition

would forget about the punishment, but this is the most important feature of the problem!

Thus, we have to incorporate the punishment into the state. So we want the states to be 4-tuples (u, v, w, p), where u, v, w are the three numbers on the wall (originally a, b, c, ultimately $0, 0, 0$), and p is the punishment accumulated so far (originally 0). Thus, the states of our process are the 4-tuples $(u, v, w, p) \in \mathbb{N}^4$, and the moves are

$$\begin{aligned}
(u, v, w, p) &\mapsto (u-1, v, w, p+vw) && \text{if } u \geq 1; \\
(u, v, w, p) &\mapsto (u, v-1, w, p+uw) && \text{if } v \geq 1; \\
(u, v, w, p) &\mapsto (u, v, w-1, p+uv) && \text{if } w \geq 1.
\end{aligned}$$

The end states are the states of the form $(0, 0, 0, p)$. The initial state is $(a, b, c, 0)$. Clearly, the game ends after $a + b + c$ moves (why?), and we want to show that the end state is $(0, 0, 0, abc)$ (so that the total punishment incurred will be abc).

What invariants could our system have? We are looking for a quantity that is preserved under each move. For instance, the move $(u, v, w, p) \mapsto (u-1, v, w, p+vw)$ decreases the first entry u by 1 while increasing the last entry p by vw. In an invariant, these two changes should balance each other out. We can achieve this by multiplying u by vw. In other words, we take the sum $uvw + p$.

Thus, the quantity $uvw + p$ defined for a state (u, v, w, p) is unchanged under the move $(u, v, w, p) \mapsto (u-1, v, w, p+vw)$. Similarly, we can see that this quantity is unchanged under the other two types of moves. Hence, this quantity is an invariant.

For the initial state $(a, b, c, 0)$, this quantity is $abc + 0 = abc$. For the final state $(0, 0, 0, p)$, it is $0 \cdot 0 \cdot 0 + p = p$. Since this quantity cannot change from the initial to the final state (because it is unchanged under any move), we thus conclude that $p = abc$. In other words, the final punishment will always be abc, no matter how you play. □

6.3. Monovariants

Monovariants can be useful, too. We saw a simple example of their use in Subsubsection 6.1.3. Here are two trickier applications (see [Grinbe20, §8.2 and §A.2.10 (First solution to Exercise 3.7.10)] for other examples):

Exercise 6.3.1. Consider an n-tuple $(a_1, a_2, \ldots, a_n)$ of real numbers. We are allowed to make the following two kinds of moves:

- **S-moves:** We pick two adjacent entries a_i and a_{i+1} that are "out of order" (i.e., satisfy $a_i > a_{i+1}$), and swap them (i.e., replace them by a_{i+1} and a_i, respectively). This is called an *S-move*.
- **P-moves:** We pick two adjacent entries a_i and a_{i+1} that are "out of order" (i.e., satisfy $a_i > a_{i+1}$), and replace them by $a_{i+1} + 1$ and $a_i + 1$, respectively. This is called a *P-move*.

For instance, examples of S-moves are $(4,3,1,2) \mapsto (3,4,1,2)$ and $(4,3,1,2) \mapsto (4,1,3,2)$, while examples of P-moves are $(4,3,1,2) \mapsto (4,5,1,2)$ and $(4,3,1,2) \mapsto (4,2,4,2)$.

(a) Prove that if we keep applying S-moves to our n-tuple (sequentially), then we eventually end up with a weakly increasing n-tuple (i.e., the process terminates).

(b) Is the same true if we allow applying both P-moves and S-moves?

(c) Is the same true if we allow applying both P-moves and S-moves, but require that all entries of our n-tuple are integers?

Solution idea. Before we come to the solution, let me illustrate the different moves and what they can lead to.

Starting with the 4-tuple $(3,1,3,2)$, we can apply an S-move to its first two entries (which are out of order), and obtain the 4-tuple $(1,3,3,2)$. Subsequently, we apply a further S-move to the last two entries, obtaining $(1,3,2,3)$. Then, we apply one more S-move to the middle two entries, obtaining $(1,2,3,3)$, at which point the 4-tuple has become weakly increasing and thus we cannot apply any further moves. Thus, our process has stopped after three moves. However, if we also allow P-moves, then we can keep making moves a bit longer: For example, starting with a P-move (applied to the first two entries), we can obtain

$$(3,1,3,2) \overset{\text{P-move}}{\mapsto} (2,4,3,2) \overset{\text{S-move}}{\mapsto} (2,4,2,3) \overset{\text{P-move}}{\mapsto} (2,3,5,3) \overset{\text{S-move}}{\mapsto} (2,3,3,5).$$

Other choices of moves lead to different results. Still, the process terminates, no matter what we do. How can we prove this, and does this hold for every possible initial state?

(a) This is a classical result (underlying various sorting algorithms), see [Grinbe20, Exercise 8.2.1] for example. The following proof is essentially the same as in [Grinbe20, Exercise 8.2.1], but organized somewhat differently.

We consider the "S-process" – i.e., the process that successively applies S-moves to an n-tuple of real numbers. (Its states are the n-tuples of real numbers, and its moves are the S-moves.) We try to construct a monovariant that is a nonnegative integer and decreases with every S-move. If we can find such a monovariant, then it will automatically follow that the process will eventually terminate, since a nonnegative integer cannot decrease endlessly.

To define our monovariant, we first introduce some notation:

- If a is any n-tuple, and if $i \in \{1,2,\ldots,n\}$, then a_i shall denote the i-th entry of a. Thus, $a = (a_1, a_2, \ldots, a_n)$ for any n-tuple a.
- If a is any n-tuple, and if $i \in \{1,2,\ldots,n\}$, then $a_{>i}$ shall denote the $(n-i)$-tuple $(a_{i+1}, a_{i+2}, \ldots, a_n)$. In other words, $a_{>i}$ is the part of a to the right of

the i-th position. (In other words, $a_{>i}$ is what remains if we remove the first i entries from a.)

- If a is an n-tuple of real numbers, and if $i \in \{1,2,\ldots,n\}$, then $\ell_i(a)$ shall denote the number of entries of $a_{>i}$ that are smaller than a_i. In other words,

$$\ell_i(a) := \left(\text{the number of } j \in \{i+1, i+2, \ldots, n\} \text{ such that } a_j < a_i\right).$$

  For example, if $a = (4,2,4,1,3)$, then $\ell_1(a) = 3$ and $\ell_2(a) = 1$ and $\ell_3(a) = 2$ and $\ell_4(a) = 0$ and $\ell_5(a) = 0$.

  We note that $\ell_i(a)$ is always a nonnegative integer and is $\leq n - i$ (since the tuple $a_{>i}$ has only $n - i$ entries in total).

- If a is an n-tuple of real numbers, then we define the nonnegative integer

$$\ell(a) := \ell_1(a) + \ell_2(a) + \cdots + \ell_n(a). \tag{4}$$

  Thus, $\ell(a)$ is the number of all pairs (i,j) of integers such that $1 \leq i < j \leq n$ and $a_j < a_i$. (These pairs are known as the *inversions* of a.)

  For example, if $a = (4,2,4,1,3)$, then $\ell(a) = 3+1+2+0+0 = 6$.

  We observe that $\ell(a) = 0$ if and only if a is weakly increasing (why?).

Now, we shall show that this nonnegative integer $\ell(a)$ (where $a = (a_1, a_2, \ldots, a_n)$ is our state) is a monovariant of the S-process. To do so, we must prove that if an S-move transforms an n-tuple a into an n-tuple b, then $\ell(b) < \ell(a)$.

Let us prove this. Let an S-move transform an n-tuple a into an n-tuple b. We must prove that $\ell(b) < \ell(a)$. We will actually prove something more specific: namely, that $\ell(b) = \ell(a) - 1$.

Recall that $a = (a_1, a_2, \ldots, a_n)$. Since b is obtained from a by an S-move, we thus must have

$$b = \left(\underbrace{a_1, a_2, \ldots, a_{i-1}}_{\text{same entries as in } a}, a_{i+1}, a_i, \underbrace{a_{i+2}, a_{i+3}, \ldots, a_n}_{\text{same entries as in } a}\right) \tag{5}$$

for some $i \in \{1,2,\ldots,n-1\}$ satisfying $a_i > a_{i+1}$ (because an S-move swaps two consecutive entries a_i and a_{i+1} that satisfy $a_i > a_{i+1}$). Consider this i. Thus, $b_i = a_{i+1}$ and $b_{i+1} = a_i$.

Now, let us see how the numbers $\ell_k(a)$ differ from the respective numbers $\ell_k(b)$:

- For each $k \in \{1,2,\ldots,i-1\}$, we have

$$\ell_k(b) = \ell_k(a). \tag{6}$$

  [*Proof of (6):*[5] Let $k \in \{1,2,\ldots,i-1\}$. Then, the n-tuples a and b have the same k-th entries (since they differ only in their i-th and $(i+1)$-st positions).

[5] I am spelling out this argument in much more detail than you would need on any exam. You could just claim (6) to be obvious. The same applies to the equalities (7), (8) and (10), except that you should perhaps point out that $a_i > a_{i+1}$ is used in the proofs of (7) and (8).

In other words, $b_k = a_k$. Furthermore the entries of b after the k-th position are the same as the entries of a after the k-th position, up to a permutation (because the S-move only swaps the entries at the i-th and $(i+1)$-st positions, and both of these positions are to the right of the k-th position). In other words, the entries of $b_{>k}$ are the same as the entries of $a_{>k}$, up to a permutation.

However, the definition of $\ell_k(b)$ yields

$$\begin{aligned}
\ell_k(b) &= (\text{the number of entries of } b_{>k} \text{ that are smaller than } b_k) \\
&= (\text{the number of entries of } b_{>k} \text{ that are smaller than } a_k) \\
&\qquad (\text{since } b_k = a_k) \\
&= (\text{the number of entries of } a_{>k} \text{ that are smaller than } a_k) \\
&\qquad \left(\begin{array}{c} \text{since the entries of } b_{>k} \text{ are the same as} \\ \text{the entries of } a_{>k}\text{, up to a permutation} \end{array} \right) \\
&= \ell_k(a) \qquad (\text{by the definition of } \ell_k(a)).
\end{aligned}$$

This proves (6).]

- We have
$$\ell_i(b) = \ell_{i+1}(a). \tag{7}$$

[*Proof of (7):* The definition of $\ell_i(b)$ yields

$$\begin{aligned}
\ell_i(b) &= (\text{the number of entries of } b_{>i} \text{ that are smaller than } b_i) \\
&= (\text{the number of entries of } b_{>i} \text{ that are smaller than } a_{i+1}) \\
&\qquad (\text{since } b_i = a_{i+1}) \\
&= (\text{the number of entries among } a_i, a_{i+2}, a_{i+3}, \ldots, a_n \text{ that are smaller than } a_{i+1}) \\
&\qquad \left(\begin{array}{c} \text{since the entries of } b_{>i} \text{ are } a_i, a_{i+2}, a_{i+3}, \ldots, a_n \\ (\text{because (5) yields } b_{>i} = (a_i, a_{i+2}, a_{i+3}, \ldots, a_n)) \end{array} \right) \\
&= (\text{the number of entries among } a_{i+2}, a_{i+3}, \ldots, a_n \text{ that are smaller than } a_{i+1}) \\
&\qquad \left(\begin{array}{c} \text{here, we removed } a_i \text{ from the list, since } a_i \\ \text{is not smaller than } a_{i+1} \text{ (in fact, } a_i > a_{i+1}) \end{array} \right) \\
&= (\text{the number of entries of } a_{>i+1} \text{ that are smaller than } a_{i+1}) \\
&\qquad (\text{since } a_{i+2}, a_{i+3}, \ldots, a_n \text{ are the entries of } a_{>i+1}) \\
&= \ell_{i+1}(a) \qquad (\text{by the definition of } \ell_{i+1}(a)).
\end{aligned}$$

This proves (7).]

- We have
$$\ell_{i+1}(b) = \ell_i(a) - 1. \tag{8}$$

[*Proof of (8):* The definition of $\ell_{i+1}(b)$ yields

$$\begin{aligned}\ell_{i+1}(b) &= (\text{the number of entries of } b_{>i+1} \text{ that are smaller than } b_{i+1}) \\ &= (\text{the number of entries of } b_{>i+1} \text{ that are smaller than } a_i) \\ &\qquad\qquad (\text{since } b_{i+1} = a_i) \\ &= (\text{the number of entries among } a_{i+2}, a_{i+3}, \ldots, a_n \text{ that are smaller than } a_i)\end{aligned} \tag{9}$$

(since the entries of $b_{>i+1}$ are $a_{i+2}, a_{i+3}, \ldots, a_n$ (because from (5), we obtain $b_{>i+1} = (a_{i+2}, a_{i+3}, \ldots, a_n)$)). But the definition of $\ell_i(a)$ yields

$$\begin{aligned}\ell_i(a) &= (\text{the number of entries of } a_{>i} \text{ that are smaller than } a_i) \\ &= (\text{the number of entries among } a_{i+1}, a_{i+2}, a_{i+3}, \ldots, a_n \text{ that are smaller than } a_i) \\ &= 1 + (\text{the number of entries among } a_{i+2}, a_{i+3}, \ldots, a_n \text{ that are smaller than } a_i)\end{aligned}$$

(here, we removed the entry a_{i+1} from the list, but this changes the number by 1 since this entry is smaller than a_i (since $a_i > a_{i+1}$)). In view of (9), we can rewrite this as $\ell_i(a) = 1 + \ell_{i+1}(b)$. Hence, $\ell_{i+1}(b) = \ell_i(a) - 1$. This proves (8).]

- For each $k \in \{i+2, i+3, \ldots, n\}$, we have

$$\ell_k(b) = \ell_k(a). \tag{10}$$

[*Proof of (10):* This is very similar to (6).]

Now, (4) (applied to b instead of a) yields

$$\begin{aligned}
\ell(b) &= \ell_1(b) + \ell_2(b) + \cdots + \ell_n(b) \\
&= \left(\underbrace{\ell_1(b)}_{\substack{=\ell_1(a) \\ \text{(by (6))}}} + \underbrace{\ell_2(b)}_{\substack{=\ell_2(a) \\ \text{(by (6))}}} + \cdots + \underbrace{\ell_{i-1}(b)}_{\substack{=\ell_{i-1}(a) \\ \text{(by (6))}}} \right) + \underbrace{\ell_i(b)}_{\substack{=\ell_{i+1}(a) \\ \text{(by (7))}}} \\
&\qquad + \underbrace{\ell_{i+1}(b)}_{\substack{=\ell_i(a)-1 \\ \text{(by (8))}}} + \left(\underbrace{\ell_{i+2}(b)}_{\substack{=\ell_{i+2}(a) \\ \text{(by (10))}}} + \underbrace{\ell_{i+3}(b)}_{\substack{=\ell_{i+3}(a) \\ \text{(by (10))}}} + \cdots + \underbrace{\ell_n(b)}_{\substack{=\ell_n(a) \\ \text{(by (10))}}} \right) \\
&= (\ell_1(a) + \ell_2(a) + \cdots + \ell_{i-1}(a)) + \ell_{i+1}(a) \\
&\qquad + (\ell_i(a) - 1) + (\ell_{i+2}(a) + \ell_{i+3}(a) + \cdots + \ell_n(a)) \\
&= \underbrace{((\ell_1(a) + \ell_2(a) + \cdots + \ell_{i-1}(a)) + \ell_{i+1}(a) + \ell_i(a))}_{=\ell_1(a)+\ell_2(a)+\cdots+\ell_{i+1}(a)} \\
&\qquad + (\ell_{i+2}(a) + \ell_{i+3}(a) + \cdots + \ell_n(a)) - 1 \\
&= \underbrace{((\ell_1(a) + \ell_2(a) + \cdots + \ell_{i+1}(a)) + (\ell_{i+2}(a) + \ell_{i+3}(a) + \cdots + \ell_n(a)))}_{\substack{=\ell_1(a)+\ell_2(a)+\cdots+\ell_n(a) \\ =\ell(a) \\ \text{(by (4))}}} - 1 \\
&= \ell(a) - 1.
\end{aligned}$$

Hence, $\ell(b) = \ell(a) - 1 < \ell(a)$. This shows that $\ell(a)$ decreases with each S-move.

As we already said, this yields that the S-process must eventually stop, because a nonnegative integer cannot decrease endlessly (without becoming negative). Hence, Exercise 6.3.1 **(a)** is solved.

(b) No. The process can go on forever.

Proof. Start with the state $\left(0, \frac{1}{3}, \frac{-1}{3}\right)$, and apply P-moves and S-moves as follows:

$$\begin{aligned}
\left(0, \frac{1}{3}, \frac{-1}{3}\right) &\overset{\text{S-move}}{\mapsto} \left(0, \frac{-1}{3}, \frac{1}{3}\right) \overset{\text{P-move}}{\mapsto} \left(\frac{2}{3}, 1, \frac{1}{3}\right) \\
&\overset{\text{S-move}}{\mapsto} \left(\frac{2}{3}, \frac{1}{3}, 1\right) \overset{\text{P-move}}{\mapsto} \left(\frac{4}{3}, \frac{5}{3}, 1\right) \\
&\overset{\text{S-move}}{\mapsto} \left(\frac{4}{3}, 1, \frac{5}{3}\right) \overset{\text{P-move}}{\mapsto} \left(2, \frac{7}{3}, \frac{5}{3}\right) \mapsto \cdots .
\end{aligned}$$

(Each odd move is a S-move, and each even move is a P-move. This process continues indefinitely because every two successive move cause all entries of the state

to increase by $\frac{2}{3}$, and thus the order relations that make the moves possible are preserved.)

(c) Yes. The process will end.

Proof. The following nice proof was found by Jan Nienhaus (`https://mathoverflow.net/questions/456983/`), although I have modified it to fit better with the theme of this worksheet.

We consider the "SP-process" – i.e., the process that successively applies S-moves or P-moves to an n-tuple of integers. Its states are the n-tuples of integers[6], and its moves are the S-moves and the P-moves. We would like to find a monovariant of this SP-process, so that we can use the same argument as in part **(a)**.

It is tempting to reuse the monovariant $\ell(a) = \sum_{k=1}^{n} \ell_k(a)$ of the S-process that we constructed in our above solution to Exercise 6.3.1 **(a)**. Unfortunately, this does not work, since $\ell(a)$ is not a monovariant of the SP-process: For instance, a P-move transforms $(1,0,1)$ into $(1,2,1)$, but $\ell(1,2,1)$ is not smaller than $\ell(1,0,1)$ (in fact, $\ell(1,2,1) = 1$ and $\ell(1,0,1) = 1$ are equal).[7]

This does not mean that our above solution to Exercise 6.3.1 **(a)** is completely useless here. In that solution, we have also defined a number $\ell_i(a)$ for each n-tuple a of numbers and each $i \in \{1,2,\ldots,n\}$. We shall again use these numbers and all the other notations we introduced in that solution. Let us now define our new monovariant:

- If a is an n-tuple of real numbers, then we define the nonnegative integer

$$m(a) := \sum_{k=1}^{n} (n-k)! \cdot \ell_k(a). \tag{11}$$

  For example, if $a = (4,2,4,1,3)$, then $m(a) = 4! \cdot 3 + 3! \cdot 1 + 2! \cdot 2 + 1! \cdot 0 + 0! \cdot 0 = 82$.

Now, we shall show that this nonnegative integer $m(a)$ (where $a = (a_1, a_2, \ldots, a_n)$ is our state) is a monovariant of the SP-process. To do so, we must prove that if an S-move or a P-move transforms an n-tuple a into an n-tuple b, then $m(b) < m(a)$.

Let us prove this. Let an S-move or a P-move transform an n-tuple a into an n-tuple b. We must prove that $m(b) < m(a)$. (This time, we cannot prove $m(b) = m(a) - 1$, but we don't have to.)

Recall that $a = (a_1, a_2, \ldots, a_n)$. By assumption, we are in one of the following two cases:

Case 1: The n-tuple b is obtained from a by an S-move.

Case 2: The n-tuple b is obtained from a by a P-move.

[6] Yes, integers! Indeed, by the requirements of Exercise 6.3.1 **(c)**, we are starting with an n-tuple of integers, and of course we will never get any non-integers during the process (since neither S-moves nor P-moves can produce any non-integers).

[7] Sometimes, $\ell(a)$ even increases with a P-move.

Let us first consider Case 2 (we will handle Case 1 afterwards). In this case, b is obtained from a by a P-move. In other words,

$$b = \left(\underbrace{a_1, a_2, \ldots, a_{i-1}}_{\text{same entries as in } a},\ a_{i+1}+1,\ a_i+1,\ \underbrace{a_{i+2}, a_{i+3}, \ldots, a_n}_{\text{same entries as in } a} \right) \tag{12}$$

for some $i \in \{1, 2, \ldots, n-1\}$ satisfying $a_i > a_{i+1}$ (because a P-move replaces two consecutive entries a_i and a_{i+1} that satisfy $a_i > a_{i+1}$ by $a_{i+1}+1$ and a_i+1). Consider this i. Thus, $b_i = a_{i+1}+1$ and $b_{i+1} = a_i + 1$.

We note that $a_1, a_2, \ldots, a_n$ are integers (since our states are tuples of integers this time). Thus, in particular, a_i and a_{i+1} are integers. Hence, from $a_i > a_{i+1}$, we obtain $a_i \geq a_{i+1} + 1 = b_i$ (since $b_i = a_{i+1} + 1$). Hence, $b_i \leq a_i$.

Now, let us see how the numbers $\ell_k(a)$ differ from the respective numbers $\ell_k(b)$. This is less predictable than in part **(a)**, but we take what we can get:

- For each $k \in \{1, 2, \ldots, i-1\}$, we have

$$\ell_k(b) \leq \ell_k(a). \tag{13}$$

[*Proof of (13):* Let $k \in \{1, 2, \ldots, i-1\}$. Then, the n-tuples a and b have the same k-th entries (since they differ only in their i-th and $(i+1)$-st positions). In other words, $a_k = b_k$.

The P-move that transforms a into b consists in swapping the i-th and $(i+1)$-st entries (which are a_i and a_{i+1}) and incrementing them both by 1. Since $k < i$, these two entries are contained in $a_{>k}$ (before the move) and in $b_{>k}$ (after the move). Hence, the tuple $b_{>k}$ is obtained from $a_{>k}$ by swapping two entries and incrementing them by 1. Inverting this operation, we see that $a_{>k}$ can be recovered from $b_{>k}$ by decrementing two entries by 1 and swapping them. Clearly, this operation (decrementing and swapping two entries) cannot decrease the number of entries that are smaller than a_k (because if an entry of $b_{>k}$ is smaller than a_k, then it remains smaller than a_k when it is decremented by 1, and therefore becomes an entry of $a_{>k}$ smaller than a_k). Hence,

$$\begin{aligned} &(\text{the number of entries of } a_{>k} \text{ that are smaller than } a_k) \\ &\geq (\text{the number of entries of } b_{>k} \text{ that are smaller than } a_k). \end{aligned}$$

However, the definition of $\ell_k(a)$ yields

$$\begin{aligned} \ell_k(a) &= (\text{the number of entries of } a_{>k} \text{ that are smaller than } a_k) \\ &\geq (\text{the number of entries of } b_{>k} \text{ that are smaller than } a_k) \\ &= (\text{the number of entries of } b_{>k} \text{ that are smaller than } b_k) \\ &\qquad\qquad (\text{since } a_k = b_k) \\ &= \ell_k(b) \qquad (\text{by the definition of } \ell_k(b)). \end{aligned}$$

In other words, $\ell_k(b) \leq \ell_k(a)$. This proves (13).]

- We have
$$\ell_i(b) \leq \ell_i(a) - 1. \tag{14}$$
[*Proof of (14):* Let us compare the two tuples $a_{>i}$ and $b_{>i}$. According to (12), we have $b_{>i} = (a_i + 1, a_{i+2}, a_{i+3}, \ldots, a_n)$, while we obviously have $a_{>i} = (a_{i+1}, a_{i+2}, a_{i+3}, \ldots, a_n)$. Thus, the two tuples $b_{>i}$ and $a_{>i}$ differ only in their first entry, which is $a_i + 1$ in the former and a_{i+1} in the latter. What is important for us is that the first entry of $a_{>i}$ is smaller than a_i (since it is a_{i+1}, but we have $a_{i+1} < a_i$), but the first entry of $b_{>i}$ is not (since it is $a_i + 1$, which is greater than a_i). Hence, the tuple $a_{>i}$ has one more entry that is smaller than a_i than $b_{>i}$ (since these two tuples differ only in their first entry).

But the definition of $\ell_i(a)$ yields
$$\begin{aligned}\ell_i(a) &= (\text{the number of entries of } a_{>i} \text{ that are smaller than } a_i) \\ &= (\text{the number of entries of } b_{>i} \text{ that are smaller than } a_i) + 1 \end{aligned} \tag{15}$$
(since the tuple $a_{>i}$ has one more entry that is smaller than a_i than $b_{>i}$).

However, let us recall that $b_i \leq a_i$. Hence, each entry of $b_{>i}$ that is smaller than b_i must also be smaller than a_i. Thus,
$$\begin{aligned}&(\text{the number of entries of } b_{>i} \text{ that are smaller than } b_i) \\ &\leq (\text{the number of entries of } b_{>i} \text{ that are smaller than } a_i).\end{aligned}$$

Now, the definition of $\ell_i(b)$ yields
$$\begin{aligned}\ell_i(b) &= (\text{the number of entries of } b_{>i} \text{ that are smaller than } b_i) \\ &\leq (\text{the number of entries of } b_{>i} \text{ that are smaller than } a_i) \\ &= \ell_i(a) - 1 \qquad (\text{by (15)}).\end{aligned}$$

This proves (14).]

- We cannot find an upper bound for $\ell_{i+1}(b)$ in terms of $\ell_{i+1}(a)$, but we can at least state the obvious: We have
$$\ell_{i+1}(b) < n - i. \tag{16}$$
[*Proof of (16):* The definition of $\ell_{i+1}(b)$ yields
$$\begin{aligned}\ell_{i+1}(b) &= (\text{the number of entries of } b_{>i+1} \text{ that are smaller than } b_{i+1}) \\ &\leq (\text{the number of all entries of } b_{>i+1}) \\ &= n - \underbrace{(i+1)}_{>i} \qquad (\text{since } b_{>i+1} \text{ is an } (n-(i+1))\text{-tuple}) \\ &< n - i.\end{aligned}$$

This proves (16).]

- For each $k \in \{i+2, i+3, \ldots, n\}$, we have

$$\ell_k(b) = \ell_k(a). \tag{17}$$

[*Proof of (17):* This is very similar to (6).]

Now, the definition of $m(b)$ yields

$$\begin{aligned}
m(b) &= \sum_{k=1}^{n} (n-k)! \cdot \ell_k(b) \\
&= \sum_{k=1}^{i-1} (n-k)! \cdot \underbrace{\ell_k(b)}_{\substack{\leq \ell_k(a) \\ \text{(by (13))}}} + (n-i)! \cdot \underbrace{\ell_i(b)}_{\substack{\leq \ell_i(a)-1 \\ \text{(by (14))}}} + \left(\underbrace{n-(i+1)}_{=n-i-1} \right)! \cdot \underbrace{\ell_{i+1}(b)}_{\substack{<n-i \\ \text{(by (16))}}} \\
&\qquad + \sum_{k=i+2}^{n} (n-k)! \cdot \underbrace{\ell_k(b)}_{\substack{=\ell_k(a) \\ \text{(by (17))}}} \\
&< \sum_{k=1}^{i-1} (n-k)! \cdot \ell_k(a) + (n-i)! \cdot (\ell_i(a) - 1) + \underbrace{(n-i-1)! \cdot (n-i)}_{\substack{=(n-i)! \\ \text{(since } (p-1)! \cdot p = p! \\ \text{for any integer } p>0)}} \\
&\qquad + \sum_{k=i+2}^{n} (n-k)! \cdot \ell_k(a) \\
&= \sum_{k=1}^{i-1} (n-k)! \cdot \ell_k(a) + \underbrace{(n-i)! \cdot (\ell_i(a) - 1) + (n-i)!}_{\substack{=(n-i)! \cdot \ell_i(a) \\ \leq (n-i)! \cdot \ell_i(a) + (n-(i+1))! \cdot \ell_{i+1}(a) \\ \text{(since } (n-(i+1))! \cdot \ell_{i+1}(a) \text{ is nonnegative)}} \\
&\qquad + \sum_{k=i+2}^{n} (n-k)! \cdot \ell_k(a) \\
&\leq \sum_{k=1}^{i-1} (n-k)! \cdot \ell_k(a) + (n-i)! \cdot \ell_i(a) + (n-(i+1))! \cdot \ell_{i+1}(a) \\
&\qquad + \sum_{k=i+2}^{n} (n-k)! \cdot \ell_k(a) \\
&= \sum_{k=1}^{n} (n-k)! \cdot \ell_k(a) = m(a)
\end{aligned}$$

(by the definition of $m(a)$). This concludes the proof of $m(b) < m(a)$ in Case 2.

Let us now consider Case 1. In this case, b is obtained from a by an S-move. Thus, as in our solution to part **(a)**, we see that

$$b = \left(\underbrace{a_1, a_2, \ldots, a_{i-1}}_{\text{same entries as in } a}, a_{i+1}, a_i, \underbrace{a_{i+2}, a_{i+3}, \ldots, a_n}_{\text{same entries as in } a} \right)$$

for some $i \in \{1, 2, \ldots, n-1\}$ satisfying $a_i > a_{i+1}$. Consider this i.

We have already analyzed how the numbers $\ell_k(b)$ relate to the numbers $\ell_k(a)$ in our above solution to part **(a)**. We could try to reuse the results of that analysis (i.e., the four equalities (6), (7), (8) and (10)) again, but there is an easier way: We instead argue that the inequalities (13), (14), (16) and (17) still hold (even though we are no longer in Case 2!). Indeed, these inequalities can be proved similarly to how they were proved in Case 2, with some minor changes (mostly simplifications, because an S-move is like a P-move but without the incrementation part). Thus, the inequality $m(b) < m(a)$ can be proved just as in Case 2.

We have now proved $m(b) < m(a)$ in both Cases 1 and 2. Hence, $m(b) < m(a)$ always holds. Therefore, $m(a)$ is a monovariant, which shows that the SP-process will eventually terminate. Part **(c)** of the exercise is solved. □

The following exercise is one of the most famous problems of the International Mathematical Olympiad, known to the contest community as "the pentagon game" (see [WegRei07], [Erikss92] and [Erikss93, Chapter 3] for further discussion and generalizations):

Exercise 6.3.2. At each vertex of a regular pentagon, an integer is written. Assume that the sum of all these five integers is positive.

We can make the following move: If x, y, z are the integers at three consecutive vertices of our pentagon (in this order), and if $y < 0$, then we can replace these integers x, y, z by $x+y$, $-y$, $z+y$, respectively (see Figure 1 for an illustration).

We perform this move repeatedly. Prove that this process will eventually come to an end (i.e., at some point, there will be no negative integers left on any vertex).

(International Mathematical Olympiad 1986/3)

Solution idea. We encode the integers at the vertices of our pentagon as a 5-tuple, listing them in clockwise order (starting from some vertex we choose at the onset).

Thus, we are dealing with a process whose states are the 5-tuples $(a, b, c, d, e) \in \mathbb{Z}^5$, and whose moves are

$$\begin{aligned}
(a,b,c,d,e) &\mapsto (a+b, -b, c+b, d, e) && \text{if } b < 0; \\
(a,b,c,d,e) &\mapsto (a, b+c, -c, d+c, e) && \text{if } c < 0; \\
(a,b,c,d,e) &\mapsto (a, b, c+d, -d, e+d) && \text{if } d < 0; \\
(a,b,c,d,e) &\mapsto (a+e, b, c, d+e, -e) && \text{if } e < 0; \\
(a,b,c,d,e) &\mapsto (-a, b+a, c, d, e+a) && \text{if } a < 0.
\end{aligned}$$

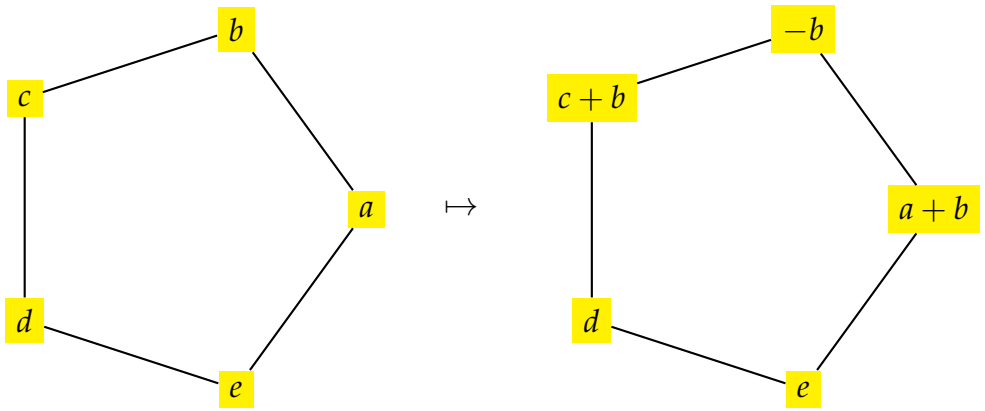

Figure 1: A move in Exercise 6.3.2. The three integers x, y, z chosen for this move are a, b, c here. This move can only be made if $b < 0$.

(Because of the cyclic symmetry, I could have just written down the first kind of move and said "the others are obtained from it by cyclic rotation", but I wanted to be precise.)

We want to show that the process must terminate (i.e., come to an end) after finitely many moves. One way to prove such claims (as we already saw in Subsubsection 6.1.3) is by finding a monovariant that is a nonnegative integer and decreases with each move. Since a nonnegative integer cannot decrease endlessly, this will yield that the process must come to an end.

Of course, this is easier said than done! Invariants are easy to find: For instance, the sum $a+b+c+d+e$ of all entries of a state (a,b,c,d,e) is an invariant (since, e.g., a move of the form $(a,b,c,d,e) \mapsto (a+b,-b,c+b,d,e)$ changes it to $(a+b)+(-b)+(c+b)+d+e$, which simplifies back to $a+b+c+d+e$). But finding a proper monovariant is harder. Here are two monovariants (one is enough, but I want to show different approaches for pedagogical purposes):

1. The quantity

$$f(a,b,c,d,e) := (a-c)^2 + (b-d)^2 + (c-e)^2 + (d-a)^2 + (e-b)^2$$

defined for each state (a,b,c,d,e) is a monovariant that decreases with each move. To prove this, we just have to check five inequalities (one for each type of move). For instance, if we make a move of the form

$$(a,b,c,d,e) \mapsto (a+b,-b,c+b,d,e) \qquad \text{with } b<0,$$

then $f(a,b,c,d,e)$ is replaced by $f(a+b,-b,c+b,d,e)$. We must therefore show that $f(a+b,-b,c+b,d,e) < f(a,b,c,d,e)$. In order to show this, we

calculate the difference:

$$\begin{aligned}
&f(a+b,-b,c+b,d,e) - f(a,b,c,d,e) \\
&= \Big(((a+b)-(c+b))^2 + ((-b)-d)^2 + ((c+b)-e)^2 + (d-(a+b))^2 + (e-(-b))^2\Big) \\
&\qquad - \Big((a-c)^2 + (b-d)^2 + (c-e)^2 + (d-a)^2 + (e-b)^2\Big) \\
&= 2b(a+b+c+d+e) \qquad \text{(after a long but straightforward computation)}.
\end{aligned}$$

Now, recall that the sum $a+b+c+d+e$ is an invariant (as we already saw), and thus equals its value at the beginning of the process; but the latter value is >0 (since we assumed that the sum of all the five integers is positive at the beginning of the process). Hence, $a+b+c+d+e>0$, so that

$$f(a+b,-b,c+b,d,e) - f(a,b,c,d,e) = 2\underbrace{b}_{<0}\underbrace{(a+b+c+d+e)}_{>0} < 0$$

and therefore $f(a+b,-b,c+b,d,e) < f(a,b,c,d,e)$. Thus, we have shown that the quantity $f(a,b,c,d,e)$ decreases whenever we make a move of the form

$$(a,b,c,d,e) \mapsto (a+b,-b,c+b,d,e) \qquad \text{with } b<0.$$

Similarly, we can see that it decreases with any move of the other four forms (indeed, because of the cyclic symmetry in the definition of $f(a,b,c,d,e)$, the proofs for the other four forms will be completely analogous and need not be written out). Hence, $f(a,b,c,d,e)$ decreases with each move. But $f(a,b,c,d,e)$ is a nonnegative integer (because it is defined as a sum of five squares of integers), and thus cannot decrease endlessly. Hence, the process must come to an end, and the problem is solved.

2. Another monovariant is the quantity

$$\begin{aligned}
&g(a,b,c,d,e) \\
&:= |a|+|b|+|c|+|d|+|e| \\
&\qquad + |a+b|+|b+c|+|c+d|+|d+e|+|e+a| \\
&\qquad + |a+b+c|+|b+c+d|+|c+d+e|+|d+e+a|+|e+a+b| \\
&\qquad + |a+b+c+d|+|b+c+d+e|+|c+d+e+a| \\
&\qquad + |d+e+a+b|+|e+a+b+c|.
\end{aligned}$$

This complicated-looking sum actually does something quite simple: It adds the absolute values of all "contiguous partial sums" of our five integers, where a "contiguous partial sum" means a sum of the integers at a few consecutive vertices of the pentagon (for example, $d+e+a$ is a contiguous partial sum, whereas $a+c+e$ is not).

Clearly, the quantity $g(a,b,c,d,e)$ is a nonnegative integer, but why is it a monovariant? This is a bit trickier to prove than the analogous property

of $f(a,b,c,d,e)$, since absolute values don't lend themselves to mechanical simplification like squares do. But it is easy once you view it from the right perspective: If we make a move of the form

$$(a,b,c,d,e) \mapsto (a+b,-b,c+b,d,e) \qquad \text{with } b<0,$$

then most of the "contiguous partial sums" remain unchanged except for the following six types:

Type 1: the ones that contain a but not b or c (for example, $e+a$);

Type 2: the ones that contain a and b but not c (for example, $e+a+b$);

Type 3: the ones that contain c but not b or a (for example, $c+d$);

Type 4: the ones that contain c and b but not a (for example, $b+c+d$);

Type 5: the one that contains c and a but not b (there is only one such sum, namely $c+d+e+a$);

Type 6: the one that contains only b (that is, b itself).

But even these types of sums change in a fairly predictable way: Sums of Type 1 are turned into sums of Type 2 (for example, $e+a$ becomes $e+a+b$ since we replace a by $a+b$), whereas sums of Type 2 are turned into sums of Type 1 (for example, $e+a+b$ becomes $e+(a+b)+(-b)=e+a$). Likewise, sums of Type 3 are turned into sums of Type 4, and vice versa. Thus, as we add the absolute values of all these sums together, the contributions of the sums of Types 1, 2, 3 and 4 remain the same (even as the specific sums trade places), and the only real changes come from the sums of Types 5 and 6. These sums are $c+d+e+a$ and b before the move, and they become $(c+b)+d+e+(a+b)$ and $-b$ after move. Thus, in order to show that the total sum $g(a,b,c,d,e)$ decreases, we only need to check that

$$|(c+b)+d+e+(a+b)|+|-b| < |c+d+e+a|+|b|.$$

But this is not hard: The addends $|-b|$ and $|b|$ on the two sides are equal, so we can cancel them and it remains to show the inequality

$$|(c+b)+d+e+(a+b)| < |c+d+e+a|.$$

Set $s := a+b+c+d+e$. Then, we can rewrite this inequality as $|s+b| < |s-b|$, since

$$(c+b)+d+e+(a+b) = \underbrace{a+b+c+d+e}_{=s}+b = s+b \qquad \text{and}$$
$$c+d+e+a = \underbrace{a+b+c+d+e}_{=s}-b = s-b.$$

But as we know, the quantity $s = a + b + c + d + e$ is an invariant and thus is positive (since we assumed that the sum of all the five integers is positive at the beginning of the process). Thus, we know that $s > 0$ and $b < 0$. From this, we can easily conclude that $|s + b| < |s - b|$ [8]. As explained above, this shows that $g(a, b, c, d, e)$ decreases under our move. Analogous arguments work for the other four types of moves, and thus we have shown that $g(a, b, c, d, e)$ is a monovariant that decreases with each move.

Since $g(a, b, c, d, e)$ is a nonnegative integer and thus cannot decrease endlessly, we thus conclude that the process must terminate. This solves the problem again.

Remarks:

1. The exercise can be generalized to hexagons, heptagons and n-gons in general. Our second monovariant $g(a, b, c, d, e)$ can easily be adapted to this generalization (again, take the sum of the absolute values of all "contiguous partial sums"), but our first monovariant $f(a, b, c, d, e)$ cannot (at least not easily). Thus, our second solution, while longer, has the advantage of generalizability.

2. In the exercise, it is assumed that the sum of all five integers is positive. This assumption is important. In fact, if the sum of all five integers is negative, then the process cannot terminate! The reason for this is simple: The sum of all five integers is an invariant, so it always stays negative (if it starts out negative). Thus, after any number of moves, there is at least one negative number on our pentagon (since otherwise, their sum could not be negative); but this means that we can make yet another move. Hence, the process goes on forever.

   What happens if the sum of the five integers is zero? See Exercise 6.6.6 for that.

□

Monovariants don't have to be numbers. They can also be logical statements that become "truer" with every move, in the sense that if they are true before the move, then they remain true after the move (but not necessarily vice versa). Here is a somewhat artificial but nice example:

Exercise 6.3.3. You have three heaps of stones: one containing 51 stones, one containing 49 stones, and one containing 5 stones. The *size* of a heap is defined to be the number of stones in this heap.

[8] *Proof.* There are many ways to check this formally, but here is the geometric argument: The number b is negative, so that it lies left of 0 on the number line. Therefore, $-b$ lies right of 0 on the number line. However, s also lies right of 0 on the number line (since $s > 0$). Hence, s is closer to $-b$ than to b. In other words, $|s - (-b)| < |s - b|$. In other words, $|s + b| < |s - b|$.

> In a single move, you can combine two heaps into one; you can also break up any even-sized heap into two equal-sized heaps (i.e., a heap with $2n$ stones becomes two heaps with n stones each).
>
> Is it possible to achieve (by a sequence of such moves) 105 heaps, each containing one stone?
>
> (Tournament of Towns 22.7.2)

Solution idea. The process described has an obvious invariant: the total number of stones. But this is not too helpful.

Let us look at the first possible move. Since there are no even-sized heaps initially, this first move must be a "combine two heaps" move. Thus, it results either in two heaps of sizes 100 and 5, or in two heaps of sizes 51 and 54, or in two heaps of sizes 56 and 49.

What do these three resulting states have in common? A bit of thought reveals one pattern: In each of these states, the sizes of the heaps have an odd divisor larger than 1 in common. For instance, 100 and 5 have the odd divisor 5 in common; 51 and 54 have the odd divisor 3 in common; 56 and 49 have the odd divisor 7 in common.

We shall call a state *good* if it has the property that the sizes of all heaps have an odd divisor larger than 1 in common. Thus, the initial state (with three heaps of sizes 51, 49 and 5) is not good, but we have just shown that any move will transform it into a good state.

Moreover, applying a move to a good state will produce another good state, because both possible types of moves ("combine two heaps" and "break an even-sized heap into two equal-sized heaps") preserve odd divisors.[9]

Thus, the goodness of a state is a monovariant: Once you obtain a good state, any further moves will preserve its goodness. In particular, this means that a non-good state cannot be obtained from a good state.

Hence, the only non-good state that can be obtained from our initial state (with three heaps of sizes 51, 49 and 5) is the initial state itself (since any other state would require making at least one move, but after that move we already have a good state in front of us, and any further moves will preserve its goodness). Hence, in particular, the desired "flat" state (i.e., the state with 105 heaps, each containing one stone) cannot be obtained, because it is not good and not the initial state. □

6.4. Working backwards

Problem-solving is metaphorically (sometimes also literally) a form of way-finding: You start from an origin, and you want to get to a destination. It is natural to

[9] *Proof.* Let d be an odd positive integer. If you combine two heaps whose sizes a and b are both divisible by d, then the size $a+b$ of the resulting heap will again be divisible by d. If you break up an even-sized heap whose size $2n$ is divisible by d into two heaps of size n each, then the size n of these resulting heaps will again be divisible by d (since d is odd, so that $d \mid 2n$ entails $d \mid n$).

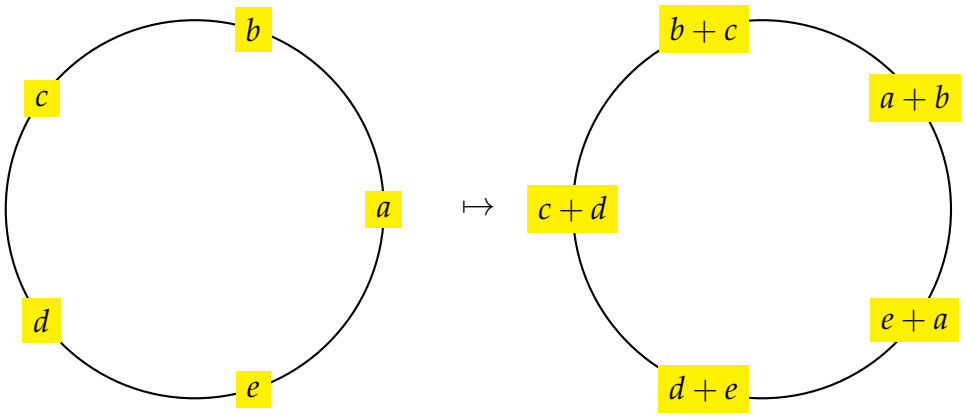

Figure 2: The daily ritual in Exercise 6.4.1.

attempt this by looking around the origin and picking a direction or trail that appears hopeful. This is called *working forwards*. In contrast, *working backwards* means exploring the destination and trying to understand how it could be reached, i.e., what the last steps of a path to this destination might be.

Both of these approaches are useful. When searching for a proof, it is helpful to analyze both the claims that can be derived from the assumptions (working forwards) and the possible claims that could lead to the desired conclusion (working backwards). When trying to construct an object with some desired properties, it can be useful to ponder how such an object could arise and what it could look like.

A much more down-to-earth manifestation of the "working backwards" principle appears when analyzing a process. Here, you often ask yourself whether a given state B can be obtained from a given state A (for instance, Exercise 6.2.1 and Exercise 6.3.3 ask such questions). Working forwards means trying to move from state A, whereas working backwards means trying to move into state B. We saw in Exercise 6.3.3 how the former can be helpful. The following exercises illustrate the usefulness of the latter.

Exercise 6.4.1. Five distinct points are chosen on a circle. At each of these five points, an integer is written.

Every day, you perform the following ritual: For every two adjacent integers on the circle, you write the sum of these two integers into the position midway between these two integers on the circle. Having written these five sums, you erase the original five integers (see Figure 2 for an illustration). In other words, every day, you replace the five integers a, b, c, d, e by $a+b,\ b+c,\ c+d,\ d+e,\ e+a$.

On the first day, the five integers are $1,\ 2,\ 3,\ 4,\ 5$. Is it possible that one day, all five integers on the circle will be even?

Solution idea. The answer is "no".

Proof. Assume the contrary. Thus, there exists an $i \in \mathbb{N}$ such that all five integers on day i are even. Pick the **smallest** such i. This i cannot be 1 (since the integers on day 1 are 1, 2, 3, 4, 5, which are not all even). Hence, day $i-1$ exists.

Let a, b, c, d, e be the five integers on day $i-1$. Thus, the five integers on day i are $a+b,\ b+c,\ c+d,\ d+e,\ e+a$. Since the latter five integers are all even, we thus conclude that $a+b,\ b+c,\ c+d,\ d+e,\ e+a$ are all even. Hence, the five integers a, b, c, d, e all have the same parity – i.e., they are either all even or all odd.

Can they be all even? No, because that would mean that all five integers on day $i-1$ are even, but this would contradict the fact that i was the **smallest** day on which all five integers are even. Hence, the five integers a, b, c, d, e must all be odd.

In other words, all five integers on day $i-1$ are odd. Therefore, $i-1$ cannot be 1 (since the integers on day 1 are 1, 2, 3, 4, 5, which are not all odd). Thus, day $i-2$ exists.

Let a', b', c', d', e' be the five integers on day $i-2$. Thus, the five integers on day $i-1$ are $a'+b',\ b'+c',\ c'+d',\ d'+e',\ e'+a'$. But we know that the five integers on day $i-1$ are a, b, c, d, e (since this is how a, b, c, d, e were defined). Comparing these facts, we conclude that the integers a, b, c, d, e are precisely the integers $a'+b',\ b'+c',\ c'+d',\ d'+e',\ e'+a'$ (although possibly in a different order). Hence,

$$\begin{aligned} a+b+c+d+e &= \left(a'+b'\right)+\left(b'+c'\right)+\left(c'+d'\right)+\left(d'+e'\right)+\left(e'+a'\right) \\ &= 2\left(a'+b'+c'+d'+e'\right). \end{aligned}$$

Therefore, $a+b+c+d+e$ is even. But $a+b+c+d+e$ is a sum of 5 odd integers (since a, b, c, d, e are odd), and thus must be odd. This contradicts the preceding sentence. This contradiction shows that our assumption was wrong, and the exercise is solved.[10]

[*Remark:* We could also rephrase part of the above argument as follows: On any day starting with day 2, the five integers on the circle have an even sum, because they have the form $a+b,\ b+c,\ c+d,\ d+e,\ e+a$ where a, b, c, d, e are the five integers from the day before. Thus, we cannot have five odd integers on the circle, except on the very first day. Actually, we can say something more concrete: The sum $a+b+c+d+e$ gets multiplied by 2 every day, so that it is divisible by 2^{i-1} on day i.] □

Exercise 6.4.2. The three numbers 1, 2 and 4 are written on a blackboard. Every hour, you are allowed to pick two of the three numbers – say, u and v – and replace each of them by $u+v$. (Thus, if the three numbers are $a,\ b,\ c$, then you can turn them into $a+b,\ a+b,\ c$ or into $a+c,\ b,\ a+c$ or into $a,\ b+c,\ b+c$.)

[10] Here is another way to obtain this contradiction: The five sums $a'+b',\ b'+c',\ c'+d',\ d'+e',\ e'+a'$ are odd (since they are the five integers on day $i-1$, but we already know that these five integers are odd). This entails that the parities of the six integers $a',\ b',\ c',\ d',\ e',\ a'$ alternate: i.e., if a' is odd, then b' is even, thus c' is odd, thus d' is even, thus e' is odd, thus a' is even; and similarly, if a' is even, then b' is odd, etc.. In either case, we obtain conflicting parities for a', which gives us the contradiction we were looking for.

Is it possible that after some time, you have three equal numbers written on the blackboard?

Solution idea. The answer is "no".

Proof. Assume the contrary. Thus, three equal numbers are obtained after some time.

We encode the three numbers on the blackboard as a triple $(a,b,c) \in \{1,2,3,\ldots\}^3$ of positive integers. Thus, we are dealing with a process whose states are triples $(a,b,c) \in \{1,2,3,\ldots\}^3$, and whose moves are

$$\begin{aligned} &(a,b,c) \mapsto (a+b,\ a+b,\ c) && \text{and} \\ &(a,b,c) \mapsto (a+c,\ b,\ a+c) && \text{and} \\ &(a,b,c) \mapsto (a,\ b+c,\ b+c)\,. \end{aligned}$$

We start in the state $(1,2,4)$, and want to show that we can never reach a state with three equal numbers, i.e., a state of the form (x,x,x).

We reason backwards: How could a state of the form (x,x,x) be reached? What could be a last move that leads to this state? The moves that lead to this state are moves of the form

$$\begin{aligned} &(u,v,x) \mapsto (x,x,x)\,, \\ &(u,x,v) \mapsto (x,x,x)\,, && \text{and} \\ &(x,u,v) \mapsto (x,x,x)\,, \end{aligned}$$

where u and v are two positive integers satisfying $u+v=x$. Thus, the state preceding (x,x,x) must have one of the forms (u,v,x), (u,x,v) and (x,u,v) for two such u and v. This is good to know, but not as informative as we might hope for, since there are lots of states of these forms.

However, we can simplify our life by also working forwards a little bit. Namely, we observe the following: Each move results in a state that has two equal entries! For example, the move $(a,b,c) \mapsto (a+b,\ a+b,\ c)$ results in a state whose first two entries are equal.

To restate this in a more convenient form, we say that a state is *good* if it has (at least) two equal entries. Each move thus results in a good state. Thus, our process will only ever produce good states (except for the initial state $(1,2,4)$, which is not good).

Hence, when reasoning backwards, we can restrict ourselves to **good** states only (except for the initial state). So we ask ourselves not what states can produce (x,x,x), but only what **good** states can produce (x,x,x). This is much easier: The only **good** states that can produce (x,x,x) in a single move are the states

$$\left(\frac{x}{2},\frac{x}{2},x\right), \qquad \left(\frac{x}{2},x,\frac{x}{2}\right), \qquad \left(x,\frac{x}{2},\frac{x}{2}\right)$$

(if x is even; otherwise, there are no such states at all).

Thus, if we ever reach the state (x,x,x), then we must reach it from one of these three states (or else directly by a single move from the initial state $(1,2,4)$, but this is easily seen to be impossible[11]). We WLOG assume that we reach it from $\left(\frac{x}{2},\frac{x}{2},x\right)$ (since the other two options are the same up to permutation of the entries, but the three numbers play symmetric roles and thus can be permuted without changing the reasoning). Thus, we reach (x,x,x) from $\left(\frac{x}{2},\frac{x}{2},x\right)$.

Now, let us think one step further backwards: How can we reach the state $\left(\frac{x}{2},\frac{x}{2},x\right)$? The only way to reach this state from a good state is by the move $\left(\frac{x}{4},\frac{x}{4},x\right) \mapsto \left(\frac{x}{2},\frac{x}{2},x\right)$ (because the move must produce the two equal entries $\frac{x}{2}$ and $\frac{x}{2}$ by summing two smaller entries, and these two smaller entries must be equal in order for the state to be good; thus, these two smaller entries must be $\frac{x}{4}$ and $\frac{x}{4}$). Similarly, the only way to reach the state $\left(\frac{x}{4},\frac{x}{4},x\right)$ from a good state is by the move $\left(\frac{x}{8},\frac{x}{8},x\right) \mapsto \left(\frac{x}{4},\frac{x}{4},x\right)$. Continuing this reasoning further, we see that the only good states from which we can reach (x,x,x) are the states of the form $\left(\frac{x}{2^i},\frac{x}{2^i},x\right)$ for $i \in \mathbb{N}$ (as well as the cyclically rotated versions of these states). We shall refer to the latter states as the *hopeful states*. They have the property that the ratio of any two entries is a (positive or negative) power of 2.

Thus, if (x,x,x) can be obtained from $(1,2,4)$ by our process, then every good state obtained along the way must be a hopeful state. In particular, the state obtained after the very first move must be a hopeful state (since it is a good state). But this state is either $(3,3,4)$ or $(5,2,5)$ or $(1,6,6)$ (since these are the three possible states that can be obtained from $(1,2,4)$ by a single move), and neither of these three options is a hopeful state (since the ratios $\frac{4}{3}$, $\frac{5}{2}$ and $\frac{6}{1}$ are not powers of 2). Hence, we obtain a contradiction.

This shows that (x,x,x) cannot be obtained from $(1,2,4)$ by our process. This completes the proof. □

Note that Exercise 6.4.2 is the beginning of an exciting story: We can play a similar game with four, five or n numbers. The answer depends on the parity of n:

- If $n > 1$ is odd, then we cannot transform the n integers $1, \underbrace{2,2,\ldots,2}_{n-1 \text{ times}}$ into n equal integers by repeatedly replacing two integers x and y by $x+y$ and $x+y$.
- If n is even, then we can transform any n integers (or, more generally, any n

[11] Indeed, the three possible moves from the initial state $(1,2,4)$ produce the three states $(3,3,4)$, $(5,2,5)$ and $(1,6,6)$, neither of which has the form (x,x,x).

complex numbers) into n equal numbers by repeatedly replacing two numbers x and y by $x+y$ and $x+y$.

See `https://mathoverflow.net/questions/452221/make-all-numbers-equal-game` for a discussion of this problem.

6.5. Class problems

The following problems are to be discussed during class.

Exercise 6.5.1. Let $a, b, c \in \mathbb{N}$. We generalize Exercise 6.2.1 by starting with a state in which there are a red, b green and c blue chameleons on the island.

We say that this state is *winnable* if and only if it is possible (by an appropriate arrangement of meetings between chameleons) to cause all chameleons to adopt the same color.

Prove that the state is winnable if and only if (at least) two of the integers a, b, c are congruent modulo 3.

Exercise 6.5.2. Start with the triple $(2,3,4)$ of numbers. In a single move, you are allowed to choose two entries x and y of the triple, and replace them with $\frac{3}{5}x - \frac{4}{5}y$ and $\frac{4}{5}x + \frac{3}{5}y$ (respectively). You are also allowed to permute the three entries.

(a) Can you reach the triple $(1,3,5)$ by such moves?

(b) What about the triple $(0,2,5)$?

(c) What about the triple $\left(\frac{1}{3}, \frac{2}{3}, \frac{16}{3}\right)$?

Exercise 6.5.3. Alla, Bella and Caella are sitting around a circular table (in this order). Alla opens a bag of nuts. She divides them evenly among Bella and Caella (as evenly as possible: i.e., Bella and Caella each get $\lfloor n/2 \rfloor$ nuts, where n is the number of nuts at Alla's disposal), and eats the remaining nut (if there is any left). Then, it is Bella's turn, and she does the same with the nuts she has: i.e., she divides them evenly among Caella and Alla, and eats the remaining nut (if there is any left). Then, Caella repeats the same procedure; after that, it is Alla's turn again, and so on.

Assume that Alla's bag has more than 3 nuts, and neither Bella nor Caella had any nuts initially. Prove the following:

(a) At least one nut gets eaten at some point.

(b) Not all nuts get eaten (i.e., at least one nut gets passed around endlessly).

(Tournament of Towns 27.5.4)

Exercise 6.5.4. Consider an $n \times n$-chessboard. Some of its n^2 squares are infected.

Every hour, the infection spreads by the following rule: If a square has at least two infected neighbors, then it gets infected as well. (The "neighbors" of a square are the squares that share an edge with it; thus, a square can have up to 4 neighbors.) Prove the following:

(a) If all diagonal squares are initially infected, then all squares will eventually get infected.

(b) If fewer than n squares are initially infected, then at least one square will never get infected.

Exercise 6.5.5. Consider n lightbulbs (arranged from left to right on a line), each of which is either on or off. Assume that $n \geq 1$.

If a lightbulb is off and has two neighbors (i.e., is neither at the left end nor at the right end of the line), then you can remove it from the line, but this causes the two neighboring lightbulbs to get flipped. (To *flip* a lightbulb means to turn it on if it was off, and to turn it off if it was on.)

Assume that all n lightbulbs are initially off. Prove that it is possible to remove $n-2$ lightbulbs by a sequence of such moves if and only if $n-1$ is not divisible by 3.

6.6. Homework exercises

Solve 4 of the 10 exercises below and upload your solutions on gradescope by November 15.

Exercise 6.6.1. Recall the Euclidean algorithm discussed in Subsubsection 6.1.3. Let u and v be two positive integers. Prove that if we start the Euclidean algorithm in the state (u, v), then it will terminate after at most $\max\{u, v\}$ moves.

[**Remark:** This is a sharp estimate, because if we start the Euclidean algorithm in the state $(n, 1)$, then it will take n moves to terminate:

$$(n,1) \mapsto (n-1,1) \mapsto (n-2,1) \mapsto \cdots \mapsto (1,1) \mapsto (1,0) \qquad (\text{or } \mapsto (0,1)).$$

On the other hand, the "grown-up" version of the Euclidean algorithm, which uses division with remainder instead of subtraction, terminates much faster: see, e.g., [Epasin83].]

Exercise 6.6.2. Consider the process whose states are the 4-tuples $(a, b, s, t) \in$

$\{1,2,3,\ldots\}^4$ of positive integers, and whose moves are

$$(a,b,s,t) \mapsto (a, b-a, s, t+s) \qquad \text{if } b > a;$$
$$(a,b,s,t) \mapsto (a-b, b, s+t, t) \qquad \text{if } a > b.$$

Its end states are clearly the 4-tuples (a,b,s,t) with $a = b$.

Now, let u and v be two positive integers. We run this process starting with the state (u,v,u,v). Prove the following:

(a) This process will eventually terminate.

(b) The end state at which it terminates is a 4-tuple (g,g,s,t) with $g = \gcd(u,v)$ and $\dfrac{s+t}{2} = \operatorname{lcm}(u,v)$.

[**Example:** If $u = 6$ and $v = 9$, then the process runs as follows:

$$(6,9,6,9) \mapsto (6,3,6,15) \mapsto (3,3,21,15),$$

and indeed we have $3 = \gcd(6,9)$ and $\dfrac{21+15}{2} = 18 = \operatorname{lcm}(6,9)$.]

Exercise 6.6.3. Consider the following variant of Exercise 6.2.1: Chameleons can come in four colors, and change colors only when three chameleons of pairwise different colors meet (in which case they all take on the fourth color). (No more than three chameleons meet at a time.)

Encode each state as a 4-tuple (a,b,c,d), where a is the number of chameleons of the first color, and so on. Say that a state (a,b,c,d) is *winnable* if an appropriate sequence of meetings can transform it into a state where all chameleons have the same color.

(a) Prove that if a state (a,b,c,d) is winnable, then (at least) three of the integers a,b,c,d are congruent modulo 4.

(b) Is the converse true?

Exercise 6.6.4. Consider the following variant of Exercise 6.2.1: We start with a red, b green and c blue chameleons. The rules of color change are the same as in Exercise 6.2.1, but the goal we want to achieve is not that all chameleons have the same color, but rather that the colors are equidistributed (i.e., there are equally many red, green and blue chameleons on the island).

Prove a simple necessary and sufficient criterion (in terms of a,b,c) for this goal to be achievable.

Exercise 6.6.5. You have a bank account with 2023 dollars. You can split a bank account into two, but this operation costs you 1 dollar in fees. (Thus, you can split a bank account with n dollars into two bank accounts with p and q dollars each, where $n = p + q + 1$.) Can you use a sequence of such operations to end up with all of your accounts containing exactly 2 dollars each?

Exercise 6.6.6. Consider Exercise 6.3.2, but now assume that the sum of all five integers is zero rather than positive. Assume furthermore that not all five integers are zero at the onset. Will the process go on forever, or will it terminate?

Exercise 6.6.7. Four real numbers a, b, c, d are written on a whiteboard. Every hour, a ghost flies by and replaces them by the numbers $a - b$, $b - c$, $c - d$, $d - a$. Prove that eventually, at least one of the numbers will be too large (in absolute value) to fit on the whiteboard, unless the original four numbers a, b, c, d were all equal.

Exercise 6.6.8. Consider a sequence $(a_0, a_1, a_2, \ldots)$ of non-zero reals that satisfies

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

Prove that there exists a real number α such that $a_{n+1} = \alpha a_n - a_{n-1}$ for all $n \geq 1$.

(54th Putnam contest 1993, Problem A2)

Exercise 6.6.9. Let m be a positive integer. Consider the ritual described in Exercise 6.4.1, but with 2^m points on the circle instead of 5 (and thus 2^m integers instead of 5).

Prove that after 2^m days, all the integers on the circle will be even.

Exercise 6.6.10. Consider an n-tuple $(a_1, a_2, \ldots, a_n)$ of real numbers. We are allowed to make the following three kinds of moves:

- **S-moves:** We pick two adjacent entries a_i and a_{i+1} that are "out of order" (i.e., satisfy $a_i > a_{i+1}$), and swap them (i.e., replace them by a_{i+1} and a_i, respectively). This is called an *S-move*.
- **L-moves:** We pick two adjacent entries a_i and a_{i+1} that are "out of order" (i.e., satisfy $a_i > a_{i+1}$), and replace them by a_{i+1} and a_{i+1} (that is, we replace the larger entry by the smaller one). This is called an *L-move*.
- **U-moves:** We pick two adjacent entries a_i and a_{i+1} that are "out of order" (i.e., satisfy $a_i > a_{i+1}$), and replace them by a_i and a_i (that is, we replace the smaller entry by the larger one). This is called a *U-move*.

For instance, $(4,3,1,2) \mapsto (3,4,1,2)$ is an S-move, $(4,3,1,2) \mapsto (3,3,1,2)$ is an L-move, and $(4,3,1,2) \mapsto (4,4,1,2)$ is a U-move.

(a) Prove that if we keep applying S-moves and L-moves to our n-tuple (sequentially), then we eventually end up with a weakly increasing n-tuple (i.e., the process terminates).

(b) Is the same true if we allow all three kinds of moves?

References

[Engel98] Arthur Engel, *Problem-Solving Strategies*, Springer 1998.

[Epasin83] P. W. Epasinghe, *Euclid's Algorithm and the Fibonacci Numbers*, Fibonacci Quarterly 23 (**1985**), issue 2, pp. 177–179.

[Erikss92] Kimmo Eriksson, *Convergence of Mozes's Game of Numbers*, Linear Algebra and its Applications **166** (1992), pp. 151–165.

[Erikss93] Kimmo Eriksson, *Strongly Convergent Games and Coxeter Groups*, PhD thesis at KTH, 1993.

[GelAnd17] Răzvan Gelca, Titu Andreescu, *Putnam and Beyond*, 2nd edition, Springer 2017.

[Grinbe20] Darij Grinberg, *Math 235: Mathematical Problem Solving*, 10 August 2021. `https://www.cip.ifi.lmu.de/~grinberg/t/20f/mps.pdf`

[StaRik08] Zvezdelina Stankova, Tom Rike, *A Decade of the Berkeley Math Circle: The American Experience, Volume I*, MSRI mathematical circles library **1**, AMS 2008.

[WegRei07] Elias Wegert, Christian Reiher, *Relaxation Procedures on Graphs*, Discrete Applied Mathematics **157** (2009), pp. 2207–2216. `https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=6ebf14e003ad7898838fb4b686c83dd6b3f1a36d`

[Zeitz17] Paul Zeitz, *The Art and Craft of Problem Solving*, 3rd edition, Wiley 2017.