Challenge — Divisible Subsequence[1]

The core of the problem is proving the lemma below. The algorithm easily follows from the proof.

Lemma 1. *Let $a_1, a_2, \ldots, a_n \in \mathbb{Z}$. Then there exists a consecutive subsequence $a_i, a_{i+1} \ldots, a_j$ with $1 \leq i \leq j \leq n$ such that*

$$n \mid (a_i + a_{i+1} + \ldots + a_j).$$

Proof. As suggested in the hint, we consider the sums $S_i = \sum_{k=1}^{i} a_k$ for $1 \leq i \leq n$. If $R_n(S_l) = 0$ for some $1 \leq l \leq n$, then we are done (we can choose $i = 1$ and $j = l$). Now we assume $R_n(S_l) \neq 0$ for all $1 \leq l \leq n$. Then the remainder of each sum mod n is in $\{1, \ldots, n-1\}$, giving $n-1$ possible remainders. Since we have n sums, by the pigeonhole principle there must exist two sums S_i, S_j with $i < j$ that have the same remainder. We then have

$$\begin{aligned} R_n(S_j) = R_n(S_i) &\iff S_j \equiv_n S_i \\ &\iff S_j - S_i \equiv_n 0 \\ &\iff n \mid (S_j - S_i) = (a_{i+1} + a_{i+2} + \ldots + a_j), \end{aligned}$$

giving us the consecutive subsequence $a_{i+1}, a_{i+2}, \ldots, a_j$ whose sum is divisible by n. $\square$

The idea of the algorithm is then the following. Using prefix sums we can compute the value of a sum S_i in $\mathcal{O}(1)$. We then check first if any of the sums are already divisible by n and if not, we find two sums with the same remainder (which are guaranteed to exist as shown in the above proof). All of this can be done in $\mathcal{O}(n)$ as follows:

Algorithm DivisibleSubsequence($a_1, a_2, \ldots, a_n$)

```
p[1] = a_1
for i = 2, ..., n do
    p[i] = p[i − 1] + a_i                                   ▷ Compute the sum S_i.
r[i] = −1 for all 1 ≤ i < n
for j = 1, ..., n do
    d = R_n(p[j])
    if d = 0 then
        return a_1, ..., a_j                                 ▷ Case n | S_j.
    else if r[d] = −1 then
        r[d] = j    ▷ Store R_n(S_j). We will have our answer when we find the next sum
with the same remainder.
    else
        return a_{r[d]+1}, ..., a_j
```

The correctness of the algorithm follows from the proof of Lemma 1.

[1] This problem was adapted from Task 5 of the HS18 exam.