**Primitive tandem Arrays** This exercise continues the use of the refinement method that was introduced in the last exercise.

A string $\alpha$ is called a **tandem array** if $\alpha$ is periodic, i.e., it can be written as $\beta^l$ for some string $\beta$ and some $l \geq 2$. When $l = 2$, the tandem array can also be called a **tandem repeat**. A tandem array $\alpha = \beta^l$ contained in a string $S$ is called **maximal** if there are no additional copies of $\beta$ before or after $\alpha$.

We are interested in identifying the maximal tandem arrays contained in a string. As discussed before, it is often best to focus on a structured subset of the strings of interest in order to limit the size of the output and to identify the most informative members. We focus here on a subset of the maximal tandem arrays that succinctly and implicitly encode all the maximal tandem arrays.

We use the pair $(\beta, l)$ to describe the tandem array $\beta^l$. Now consider the tandem array $\alpha = abababababababab$. It can be described by the pair $(abababab, 2)$, or by $(abab, 4)$, or by $(ab, 8)$. Which description is best? Since the first two pairs can be deduced from the last, we choose the later pair. This “choice” will now be precisely defined.

A string $\beta$ is said to be **primitive** if $\beta$ is not periodic. For example, the string $ab$ is primitive while $abab$ is not. The pair $(ababababababab, 8)$ is the preferred description of $ababababababab$ because string $ab$ is primitive. The preference for primitive strings extends naturally to the description of maximal tandem arrays that occur as substrings in larger strings. Given a string $S$, we use the triple $(i, \beta, l)$ to mean that a tandem array $(\beta, l)$ occurs in $S$ starting at position $i$. A triple $(i, \beta, l)$ is called a pm-triple if $\beta$ is primitive and $\beta^l$ is a maximal tandem array.

For example, the maximal tandem arrays in **mississippi** described by the pm-triples are $(2, iss, 2)$, $(3, s, 2)$, $(3, ssi, 2)$, $(6, s, 2)$, $(9, p, 2)$. Note that two or more pm-triples can have the same first number, since two different maximal tandem arrays can begin at the same position. For example the two maximal tandem arrays $ss$ and $ssisi$ both begin at position three of **mississippi**.

The pm-triples succinctly encode all the tandem arrays in a given string $S$. Crochemore [?] (with different terminology) used a successive refinement method to find all the pm-triples in $O(n \log n)$ time. This implies the very non-trivial fact that in any string of length $n$ there can be only $O(n \log n)$ pm-triples. The method in [?] finds the $E_k$ partition for each $k$. The following
Lemma 1: There is a tandem repeat of a $k$-length substring $\beta$ starting at position $i$ of $S$ if and only if the numbers $i$ and $i + k$ are both contained in a single class of $E_k$ and no numbers between $i$ and $i + k$ are in that class.

Problem: Prove Lemma 1. One direction is easy. The other direction is harder and it may be useful to use the following Lemma which you need not prove (unless you want to.)

Lemma 2: Let $\gamma$ and $\delta$ be two non-empty strings such that $\gamma\delta = \delta\gamma$. Then $\delta = \rho^i$ and $\gamma = \rho^j$ for some string $\rho$ and positive integers $i$ and $j$.

Lemma 2 says that if a string is the same before and after a circular shift (so that it can be written both as $\gamma\delta$ and $\delta\gamma$, for some strings $\gamma$ and $\delta$) then $\gamma$ and $\delta$ can both be written as concatenations of some single string $\rho$.

For example, let $\delta = abab$ and $\gamma = ababab$, so $\delta\gamma = ababababab = \gamma\delta$. Then $\rho = ab$, $\delta = \rho^2$ and $\gamma = \rho^3$.

Lemma 1 makes it easy to identify pm-triples. Assume that the indices in each class of $E_k$ are sorted in increasing order. Lemma 1 implies that $(i, \beta, j)$ is a pm-triple, where $\beta$ is a $k$-length substring, if and only if some single class of $E_k$ contains a maximal series of numbers $i, i + k, i + 2k, ..., i + jk$, such that each consecutive pair of numbers differs by $k$.

Problem: Explain this in detail.

By modifying the reverse refinement method developed in the last homework, show how to compute all the $E_k$ partitions for all $k$ (not just the powers of two), in $O(n^2)$ time.

Problem: Give implementation details to maintain the indices of each class sorted in increasing order. Extend that method, using Lemma 1, to obtain an $O(n^2)$-time algorithm to find all the pm-triples in a string $S$.

In order to find all the pm-triples in $O(n \log n)$ time, Crochemore [?] used one additional idea, which will be part of HW 4.