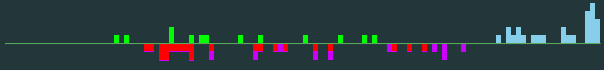
K: Khoikhoi Dralinpome

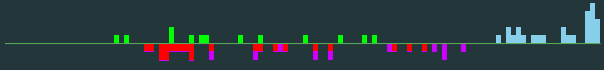
Problem author: Mike de Vries



Problem: Given a list of nouns, determine the largest dralinpome that can be made through concatenation of a subset.

K: Khoikhoi Dralinpome

Problem author: Mike de Vries

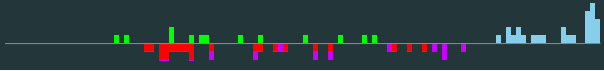


Problem: Given a list of nouns, determine the largest dralinpome that can be made through concatenation of a subset.

Observation 1: Dralinpome $\iff$ at most one character has an odd frequency.

K: Khoikhoi Dralinpome

Problem author: Mike de Vries



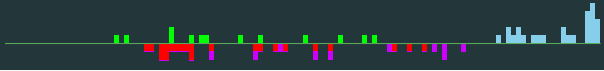
Problem: Given a list of nouns, determine the largest dralinpome that can be made through concatenation of a subset.

Observation 1: Dralinpome $\iff$ at most one character has an odd frequency.

Observation 2: The specific nouns only matter for the output. Besides that, all that matters is the length of a noun and which letters have odd frequency in that noun.

K: Khoikhoi Dralinpome

Problem author: Mike de Vries



Problem: Given a list of nouns, determine the largest dralinpome that can be made through concatenation of a subset.

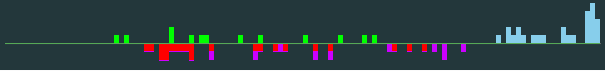
Observation 1: Dralinpome $\iff$ at most one character has an odd frequency.

Observation 2: The specific nouns only matter for the output. Besides that, all that matters is the length of a noun and which letters have odd frequency in that noun.

Idea: Precompute a 26-bit letter-parity mask for each noun. We can efficiently determine whether a subset of words forms a dralinpome by XORing these masks.

K: Khoikhoi Dralinpome

Problem author: Mike de Vries



Problem: Given a list of nouns, determine the largest dralinpome that can be made through concatenation of a subset.

Observation 1: Dralinpome $\iff$ at most one character has an odd frequency.

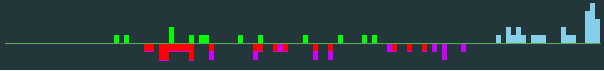
Observation 2: The specific nouns only matter for the output. Besides that, all that matters is the length of a noun and which letters have odd frequency in that noun.

Idea: Precompute a 26-bit letter-parity mask for each noun. We can efficiently determine whether a subset of words forms a dralinpome by XORing these masks.

Naive solution: Iterating over all 2^n subsets of nouns is too slow.

K: Khoikhoi Dralinpome

Problem author: Mike de Vries



Problem: Given a list of nouns, determine the largest dralinpome that can be made through concatenation of a subset.

Observation 1: Dralinpome $\iff$ at most one character has an odd frequency.

Observation 2: The specific nouns only matter for the output. Besides that, all that matters is the length of a noun and which letters have odd frequency in that noun.

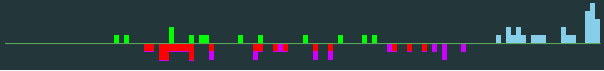
Idea: Precompute a 26-bit letter-parity mask for each noun. We can efficiently determine whether a subset of words forms a dralinpome by XORing these masks.

Naive solution: Iterating over all 2^n subsets of nouns is too slow.

Observation 3: We can use a meet-in-the-middle approach!

K: Khoikhoi Dralinpome

Problem author: Mike de Vries



Problem: Given a list of nouns, determine the largest dralinpome that can be made through concatenation of a subset.

Observation 1: Dralinpome $\iff$ at most one character has an odd frequency.

Observation 2: The specific nouns only matter for the output. Besides that, all that matters is the length of a noun and which letters have odd frequency in that noun.

Idea: Precompute a 26-bit letter-parity mask for each noun. We can efficiently determine whether a subset of words forms a dralinpome by XORing these masks.

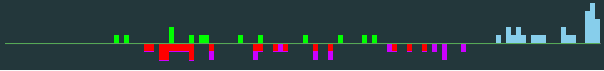
Naive solution: Iterating over all 2^n subsets of nouns is too slow.

Observation 3: We can use a meet-in-the-middle approach!

Solution: Compute the XORed letter-parity mask for each possible subset of the first and last $n/2$ nouns separately. On the second half, XOR with 0 and $1, 2, 4, \ldots, 2^{25}$, and see if any were found on the first pass. If so, joining these subsets forms a dralinpome. This finds every possible dralinpome; return the longest one.

K: Khoikhoi Dralinpome

Problem author: Mike de Vries



Problem: Given a list of nouns, determine the largest dralinpome that can be made through concatenation of a subset.

Observation 1: Dralinpome $\iff$ at most one character has an odd frequency.

Observation 2: The specific nouns only matter for the output. Besides that, all that matters is the length of a noun and which letters have odd frequency in that noun.

Idea: Precompute a 26-bit letter-parity mask for each noun. We can efficiently determine whether a subset of words forms a dralinpome by XORing these masks.

Naive solution: Iterating over all 2^n subsets of nouns is too slow.

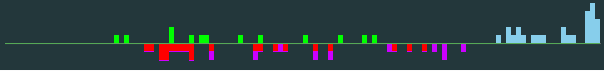
Observation 3: We can use a meet-in-the-middle approach!

Solution: Compute the XORed letter-parity mask for each possible subset of the first and last $n/2$ nouns separately. On the second half, XOR with 0 and $1, 2, 4, \ldots, 2^{25}$, and see if any were found on the first pass. If so, joining these subsets forms a dralinpome. This finds every possible dralinpome; return the longest one.

Running time: $\mathcal{O}((n + 26)2^{n/2})$.

K: Khoikhoi Dralinpome

Problem author: Mike de Vries



Problem: Given a list of nouns, determine the largest dralinpome that can be made through concatenation of a subset.

Observation 1: Dralinpome $\iff$ at most one character has an odd frequency.

Observation 2: The specific nouns only matter for the output. Besides that, all that matters is the length of a noun and which letters have odd frequency in that noun.

Idea: Precompute a 26-bit letter-parity mask for each noun. We can efficiently determine whether a subset of words forms a dralinpome by XORing these masks.

Naive solution: Iterating over all 2^n subsets of nouns is too slow.

Observation 3: We can use a meet-in-the-middle approach!

Solution: Compute the XORed letter-parity mask for each possible subset of the first and last $n/2$ nouns separately. On the second half, XOR with 0 and $1, 2, 4, \ldots, 2^{25}$, and see if any were found on the first pass. If so, joining these subsets forms a dralinpome. This finds every possible dralinpome; return the longest one.

Running time: $\mathcal{O}((n + 26)2^{n/2})$.

Statistics: 72 submissions, 13 accepted, 26 unknown