Problem Statement

From The Weekly Challenge - 144 Task #2: Ulam Sequence retrieved on 2021-12-22 at 17:34-05:

Submitted by: Mohammad S Anwar

You are given two positive numbers, $u$ and $v$.

Write a script to generate Ulam Sequence having at least 10 Ulam numbers where $u$ and $v$ are the first 2 Ulam numbers.

For more information about Ulam Sequence, please checkout the website.

The standard Ulam sequence (the (1, 2)-Ulam sequence) starts with $U_1 = 1$ and $U_2 = 2$. Then for $n > 2$, $Un$ is defined to be the smallest integer that is the sum of two distinct earlier terms in exactly one way and larger than all earlier terms.

Example 1
Input: $u = 36$, $v = 2$
Output: 1, 2, 3, 4, 6, 8, 11, 13, 16, 18

Example 2
Input: $u = 2$, $v = 3$
Output: 2, 3, 5, 7, 8, 9, 13, 14, 18, 19

Example 3
Input: $u = 2$, $v = 5$
Output: 2, 5, 7, 9, 11, 12, 13, 15, 19, 23

Discussion

See the commented program below.

I make no claim this solution is optimal.

Raku Solution

I like Raku much better than Perl. One reason: more expressive programming operators.

```raku
# Number of Ulam sequence numbers to generate.
my $n = 10;

# Read $u and $v---the first and second Ulam sequence numbers.
my $u = $*IN.get;
my $v = $*IN.get;
say "Input: \$u = $u, \$v = $v";

# Initialize the Ulam sequence.
my @ulam = ($u, $v);

while @ulam.elems < $n {
    # Compute @ulam[0]+@ulam[0], @ulam[1]+ulam[2], ...
    # Save all sums tha are greater than the last current Ulam sequence element.
    # And sort the list numerically.
    my @sum = (@ulam X+ @ulam).grep(* > @ulam[*-1]).sort(++);

    # For each @sum, tally the number of times it occurs.
    my %tally = ();
    %tally{$_}++ for @sum;
```
# Go through the %tally elements in numerically sorted order.
for %tally.keys.sort(++) {
    # If the tally value is two or three,
    # then add this tally value to the Ulam sequence,
    # and calculate the next Ulam sequence value.
    # The tally value occurs twice or three times because,
    # for example, when computing the forth (1,2)-Ulam sequence
    # element, 2 + 4 = 3 + 3 = 4 + 2 = 6.
    (%tally{$_} == 2|3) and @ulam.push($_), last;
}

say "Output: {@ulam.join(', ')}";