Conditional probability

P("Some event" | "Given something we know/assume is true")

Law of Total Probability

If we don't know the second part, we can decompose: Assume it's true. Then correct for the incertitude. And integrate that over all possible values.

$$P(event) = \sum_{all\ possibilities\ of\ X} P(\text{ event}|\text{ assume X })P(X)$$

See https://en.wikipedia.org/wiki/Conditional_probability

At least One

P("at least one ") = 1 - P("exactly zero")

P("exactly zero")

= P("exactly 0 at step 1" AND "exactly 0 at step 2" AND ... 0 at step k)

= P("exactly 0 at step 1") * P("exactly 0 at step 2") * ... P(0 at step k)

(And is multiplication, Or is addition)

The idea:

- Instead of dealing with sequences of possible output we deal with probabilities after the nth step.
- So instead of going deep first search, we do breathe first search.
- Unfortunately, the pulls are not independent. They depend on previous pulls.
- Fortunately, that dependence exists only via a compact state.
    - Mostly via the remaining budget. (Maybe also add is the 4* pool available yes/no)
- Not all budgets are as likely, we keep track of the probability of each.

- The base case is easy. There's a valid set of sleep style given a budget at step N.
  The probability is 1/size(valid set) if the target is in the valid set. Or 0 if the target is not.

- For the next case, we'll call the base case again and again for different possible budgets that are remaining. Use the Law of Total Probability to correctly assemble those probabilities togethers.

- At each step we find the probability that we pick target at this step. (we allow repeat) Then we can flip to not pick it, combine (multiply) the "not pick" for a global "exactly zero" then flip again for "at least once."

Easy case, step 1

P("exactly 0" | budget) = 1 – P("select this" | budget)

>> Just count the number of available Pokémon below budget.
Then P("select this") = 0 is self over budget. Or 1/ count below(budget).

We can trim the list of all SPO to just SPO bellow initial budget.

Medium Case, step 2

P("exactly 0" | budget after step1) = 1 – P("select this" | budget after step1)

P("select this" | budget after step1) =

$$\sum_{bx} P(\text{ “select this” }|\text{budget} = \text{bx}) P(bx|step2)$$

>> Loop all the valid sleep style for step 1.
>> Compute the remaining budget (parent Budget - SPO)
>> Keep track of the different budgets and their counts
Then P(budget X) = count self / count all

Once we have the different budgets, we can apply the P("select this") of step 1 for each of them.

If a budget is below SPO, P("select this") will be zero for the remaining of the algorithm so we can just drop that budget. However, count all must still account for it.

Hard Case, step 3+

For each of the budgets at the end of step 2.

This is now a "parent Budget" for the loop described in step 2.
We'll end up with ((parent Budget - SPOi), parent Probability*count/count all)

The probability of the children budget is now the probability amongst the children multiplied by the probability of the parent.

It's likely that multiple parents will results in the same children budget. It's possible to merge those children by adding their probabilities.

Step2 is really the same as step3+ but with a single parent budget.