```
double prob, path_prob = 1, choice_prob = 1;
int sum = 0;

klee_make_symbolic(&prob, f"prob_sym_{i}");
for i in range(10):
        int d;
        klee_make_symbolic(&d, f"d_sym_{i}");

        d = bernoulli(prob);
        (d == 1) ? choice_prob = prob : choice_prob = (1 - prob);
        path_prob = path_prob * choice_prob;

        sum = sum + d;

klee_dump(path_prob)
klee_dump(sum) // E[heads in "n" runs]
```

Fig. 1. Program listing for n bernoulli trails experiment.

We consider a independent *bernoulli* trials here of flipping a fair coin "n" times.

$$choice\_prob = \begin{cases} p & \text{if } d \text{ value is 1 corresponding to getting a "heads"} \\ 1-p & \text{if } d \text{ value is 0 corresponding to getting a "tails"} \end{cases}$$

for example, on the ith run $\vec{d}$ can have a value.

$$\vec{d_i} = encode(< 0, 0, 0, 1, 1, 0, 1, 1, 1, 0 >)$$

Based on the value of the $\vec{d_i}$, we get w_i value using *choice_prob*.

$$w_i = (p)^{x_i} * (1-p)^{n-x_i} \tag{1}$$

where x_i denotes the number of *heads* in the ith randomized run and for n runs. $n = 10$ for the case in the above example.

We consider top "k" randomized runs now for the optimization query. The expression for optimization thus becomes

$$maximize(\sum_{i=1}^{k} w_i)$$

On substituting the value of w_i from (1).

$$maximize(\sum_{i=1}^{k} (p)^{x_i} * (1-p)^{n-x_i}) \tag{2}$$

After performing the optimization above, we get different values of $\vec{d_i}$. For i^{th} randomized run. we get a single one hot encoded $\vec{d}$ vector. We show below the encoding for a few i values.

$$\vec{d_1} = encode(< 0, 0, 0, 1, 1, 0, 1, 1, 1, 0 >)$$

$$\vec{d_3} = encode(< 0, 1, 0, 1, 1, 0, 0, 0, 1, 0 >)$$

$$Objective = maximize(\sum_{i=1}^{k} (p)^{x_i} * (1-p)^{n-x_i}) \mid \forall(i,j)[x_i \neq x_j] \qquad (3)$$

Fig. 2. Optimization Expression for k randomized paths

$$\vec{d_4} = encode(< 1, 0, 1, 0, 1, 0, 1, 1, 0, 1 >)$$

...

We run the optimization by renaming the k `pse` variables set appropriately and then impose the *distinct* clause so that we don't run the optimization on the same *randomized* runs again.

For best approximation of the expected value of heads in n trials,

$$EV(heads) = (\sum_{i=1}^{k} w_i * sum_i) \qquad (4)$$

where both w_i and sum_i can both be computed from the corresponding $\vec{d_i}$ expression we get from the *model* of the *optimization* query Eq 2