4. Run your algorithm on the `hoos.cnf` instance, and list all solutions that satisfy the problem instance.

```
Enter file name: hoos.cnf
Number of satisfying solutions: 1
[True, True, False, False, False]
```

5. Run your algorithm on the `uf20-01.cnf` instance. How many solutions satisfy the problem instance?

```
Enter file name: uf20-01.cnf
Number of satisfying solutions: 8
[False, True, True, True, False, False, False, True, True, True, True, False, False, True, True, False, True, True, True, True]
[True, False, False, False, False, True, False, False, False, False, False, False, True, True, True, False, True, False, False, True]
[True, False, False, False, False, True, False, False, True, False, False, False, False, True, True, False, True, False, False, True]
[True, False, False, False, False, True, False, False, True, False, False, False, True, True, True, False, True, False, False, True]
[True, False, False, True, False, False, False, False, False, True, False, False, True, True, True, False, True, False, False, True]
[True, False, False, True, False, False, False, True, False, True, False, False, True, True, True, False, True, False, False, True]
[True, False, False, True, False, True, False, False, False, False, False, False, True, True, True, False, True, False, False, True]
[True, False, False, True, False, True, False, False, False, True, False, False, True, True, True, False, True, False, False, True]
```

6. It is impractical to do the same for a larger instance, say one contained in `uf100-430`. How could you estimate the time needed by your computer program to complete the task on such an instance? Justify your answer.

In this case, it would be very impractical to apply the same algorithm for a much larger instance. In this case, a uf100-430 refers to an instance with 100 variables and 430 clauses, so the program would have to check all the possibilities, that would be 2^{100} possibilities.
Given the fact that the program runs on a time complexity of $O(2^n * k)$, where **n** is the number of variables and **k** is the number of clauses, for this example, it becomes:

$O(2^{100} * 430)$

If we assume the program can check one million possibilities per second and each possibility involves checking all the 430 clauses, we can define the total number of assignments as:

$2^{100} \approx 1{,}27 * 10^{30}$

Now by calculating that time in seconds is:

$$\frac{1{,}27 \cdot 10^{30}}{10^{6}} \cdot 430$$

This is equal to $5{,}46 * 10^{26}$ seconds

In years, that's :

$$\frac{5{,}47 \cdot 10^{26}}{60 \cdot 60 \cdot 24 \cdot 365}$$

Which is around **$1{,}73 * 10^{19}$ years**

That's the time it would take for a problem as complex as this. Although the number of clauses heavily influences the time it takes, the main thing to lookout for here is the number of variables that make it an exponential problem. So, brute-forcing this would be heavily unadvised.

Ricardo Rosa A62461

Lab 1