Symbols:

- NOT: `!`
- AND: `&&`
- OR: `||`
- IMPLY: `->`
- ABSOURD: `0 (false)`
- TAUTOLOGY: `1 (true)`

Duality Law (De Morgan):

A formula holds even by inverting `1s` with `0s` and `&&`s with `||`s (and viceversa). For the same purpose it's possible to invert variables with their negate form.

Examples with steps:

`P && 1 => !P || 0`

- invert `1` with `0` => `P && 0`
- invert `P` => `!P && 0`
- invert `&&` with `||` => `!P || 0`

`!(P && Q) => !P || !Q`

- invert `P` and `Q` => `!P && !Q`
- invert `&&` with `||` => `!P || !Q`

`!(!P || !Q) => P && Q`

- invert `!P` and `!Q` => `P || Q`
- invert `||` with `&&` => `P && Q`

Normal Form Definition:

DNF: *disjunction of conjunctions (or literals)*

```
F = P1 || (P2 && P3) || !P4 || (!P5 && P6 && P7) || ...
```

CNF: *conjunction of disjunctions (or literals)*

```
F = P1 && (P2 || P3) && !P4 && (!P5 || P6 || P7) && ...
```

Reduction Methods for CNF and DNF:

Method 1 - using tautology laws:

1. delete connectives and absourds (like `->` or `0`) with equivalent `&&` or `||`

   ex. `P -> Q => !P && Q` ; `0 => !A && A` ; `1 => !A || A`

2. use duality (De Morgan) to lead negative forms to atomic expressions

   ex. `!(A || B) => !A && !B`

3. use distributivity to reach the right normal form (remeber that normal form is a conjunction of disjunctions or viceversa)

Method 2 - using truth table:

1. Compute truth table of the given formula
2. For DNF concatenate 1 rows with `||`
3. For CNF use duality (De Morgan) to exchange `&&` with `||`, and exchange `1` rows with `0` rows (and viceversa). Than apply DNF rules.

Note: In any case, for passing from a CNF to a DNF it's possible to use duality (De Morgan law)

Example Method 1 *(tautology laws)*:

`A || (A -> B)` = `A || (!A && B)` = `(A) || (!A) || (B) {DNF}` = `(A || !A || B) {CNF}`

We easily find out that the formula is a tautology: `A || !A => 1` always true

DNF = `P1 || P2 || P3`, where `P1 = A` , `P2 = !A` , `P3 = B`

CNF = `P1`, where `P1 = (A || !A || B)`

Example Method 2 *(truth table)*:

DNF:

```
| A | B | A->B  |
-----------------
| 0 | 0 |   1   |  -> !A && !B (because A=0 and B=0 give us 1, so take !A=1 and !B=1)
| 0 | 1 |   1   |  -> !A && B (because A=0 and B=1 give us 1, so take !A=1)
| 1 | 0 |   0   |  (skip this row because is a 0)
| 1 | 1 |   1   |  -> A && B (because A=1 and B=1 give us 1)
```

result = `(!A && !B) || (!A && B) || (A && B)`

CNF:

It is usefull to add the nagate column of the formula too invert `0` and `1` rows

```
| A | B | A->B  | !(A->B) | (added negate column of formula)
---------------------------
| 0 | 0 |   1   |    0    | (skip this row because is a 0 on negate formula)
| 0 | 1 |   1   |    0    | (skip this row because is a 0 on negate formula)
| 1 | 0 |   0   |    1    | -> A || !B => !A && B (using duality / De Morgan)
| 1 | 1 |   1   |    0    | (skip this row because is a 0 on negate formula)
```

result = `(!A && B)`

To be even faster it's possible to invert all the table if it doesn't cointain too much variables. After that apply DNF rules but with `&&` operator

```
| A | B | A->B  |      | !A  | !B  | !(A->B) |
-----------------      -----------------------
| 0 | 0 |   1   |      |  1  |  1  |    0    |
| 0 | 1 |   1   |  =>  |  1  |  0  |    0    |
| 1 | 0 |   0   |      |  0  |  1  |    1    | -> consider only this row: !A && B
| 1 | 1 |   1   |      |  0  |  0  |    0    |
```

result = `(!A && B)`

Advantages of Normal Forms:

DNF: form that represents truth table, it is possible to see immediately if the formula is satisfiable

CNF: form that represents tautologies, if each part of CNF is a tautology, so it will be the entire formula