1. Write each of the following Boolean expressions as a polynomial in $\mathbb{F}[x,y,z]$.

   (a) $x \wedge y$

   (b) $x \vee y$

   (c) $\overline{x}$

   (d) $\overline{x} \vee (y \wedge \overline{z})$

   It may help to write the *truth table* of each. Remember that $\wedge$ is AND and $\vee$ is OR.

2. Consider the following simple model of the *lac* operon:

$$\begin{aligned} f_M &= \overline{R} & f_R &= \overline{A} \\ f_P &= M & f_A &= L \wedge B \\ f_B &= M & f_L &= P \end{aligned}$$

   For this problem, make the convention that $(x_1, x_2, x_3, x_4, x_5, x_6) = (M, P, B, R, A, L)$.

   (a) (Optional) Sketch the signed wiring diagram of this Boolean network.

   (b) (Optional) Justify each function in a single sentence. What other assumptions are made in this model? (E.g., presence or absense of external lactose and glucose?)

   (c) Write each function as a polynomial over $\mathbb{F}_2 = \{0,1\}$. Then, write out a system of equations whose solutions are the fixed points of the Boolean network.

   (d) Find the fixed points by computing a Gröbner bases with a software package.

3. Consider a Boolean function $f\colon \mathbb{F}_2^3 \to \mathbb{F}_2$ with the following partial truth table

xyz	111	000	110
$f(x,y,z)$	0	0	1

   which we can express as the following input-ouput pairs of data:

$$\mathcal{D} = \big\{(\mathbf{s}_1, t_1), (\mathbf{s}_2, t_2), (\mathbf{s}_3, t_3)\big\} = \big\{(111, 0), (000, 0), (110, 1)\big\}.$$

   (a) Compute the pseudomonomial $p(\mathbf{s}_i, \mathbf{s}_j)$, for each distinct pair.

   (b) Use a computer algebra package like Macaulay2 to compute the primary decomposistion of the pseudomonomial ideal

$$J_{\Delta^c_{\mathcal{D}}} = \big\langle p(\mathbf{s}_i, \mathbf{s}_j) \mid t_i < t_j \big\rangle.$$

   (c) Find the signed min sets of f.

   (d) For each signed min set, find a Boolean function that fits the data with that as its wiring diagram.

4. In Macaulay2, the following commands produce a shortcut for converting Boolean functions into polynomials:

```
RingElement | RingElement :=(x,y)->x+y+x*y;
RingElement & RingElement :=(x,y)->x*y;
```

   For example, Problem 1(d) can be easily solved with the command: `(1+x) | (y & (1+z))`.
   Determine how to do this in Singular, for those familiar with with that software.