Problem 55. Let S, T be posets (as categories). What is an adjunction in these categories?

Solution.

Suppose $F : S \to T$ is a functor. Then, $U : T \to S$ is its right adjoint if and only if $\mathbf{Hom}_T(Fs, t) \cong \mathbf{Hom}_S(s, Ut)$ for all $s \in S, t \in T$.

We can think of F as an order preserving function. That is, $\mathbf{Hom}_T(Fs, t)$ is equal to $\{*_{Fs,t}\}$ if $Fs \leq_T t$ and empty otherwise. Similarly, $\mathbf{Hom}_S(s, Ut)$ is $*_{s,Ut}$ iff $s \leq_S Ut$.

So, $\mathbf{Hom}_T(Fs, t) \cong \mathbf{Hom}_S(s, Ut)$ for all $s \in S, t \in T$ if and only if $\forall s, t, Fs \leq_T t \iff s \leq_S Ut$.

For the sake of completeness, we will check that this (trivial) isomorphism is natural. (I will only check naturality in s. t is almost identical.)Let the name of the isomorphism be α that sends $*_{Fs,t} \mapsto *_{s,Ut}$. Let $f :' s \to s$ be the statement $s' \leq s$. Then, by transitivity of the poset operation, we conclude that the following commutes (we trace the only possible morphism around the square)

$$\begin{tikzcd}
\mathbf{Hom}_T(Fs,t) \arrow[rr, "-\circ Ff"] \arrow[dd, "\alpha_s"'] & & \mathbf{Hom}_T(Fs',t) \arrow[dd, "\alpha_{s'}"] \\
& {*_{Fs,t} \to *_{Fs',t} \atop {\downarrow \qquad\quad \downarrow} \atop *_{s,Ut} \to *_{s',Ut}} & \\
\mathbf{Hom}_S(s,Ut) \arrow[rr, "-\circ f"'] & & \mathbf{Hom}_S(s',Ut)
\end{tikzcd}$$

This is a bit overkill, but confirms that $F \dashv U$. $\square$

Problem 58.

Show that the free functor from **Set** to **Mon** is left adjoint to the underlying set (forgetful) functor from **Mon** to **Set**.

Personal note: do it with hom-set and unit-counit.

Solution. We discussed the functor $F : \mathbf{Set} \to \mathbf{Mon}$ in class and gave its action on monoids. On set functions $f : A \to B$, we define $Ff : FA \to FB$ by $Ff((n; x_1, x_2, \dots, x_n)) = (n; f(x_1), f(x_2), \dots f(x_n))$. This is clearly a monoid homomorphism in FB, as $Ff((n; x) \circ (m, y)) = (n+m; f(x), f(y)) = (n; f(x)) \circ (m; f(y))$. It also clearly respects identities and function composition in **Set**.

We also discussed the function $U : \mathbf{Mon} \to \mathbf{Set}$ and gave its action on monoids. On monoid homomorphisms, $U(\phi : F \to G)$ is just ϕ on the underlying set. Every monoid homomorphism is just a set function with some fancy rules.

We'll do the unit-counit proof first. We first need a unit $\eta : 1_{\mathbf{Set}} \Rightarrow UF$ and counit $\varepsilon : FU \Rightarrow 1_{\mathbf{Mon}}$. η will have components for every set $\eta_A : A \to UFA$. We define $\eta_A(x) = (1; x)$. We also define $\varepsilon_M : FUM \to M$ with $\varepsilon((n; x_1, x_2, \dots x_n)) = x_1 x_2 \dots x_n$ where multiplication in the RHS is in M.

We'll take it one diagram at a time.

$$\begin{tikzcd} F \arrow[r, "F\eta"] \arrow[dr, "1_{[\mathbf{Set},\mathbf{Mon}]}"'] & FUF \arrow[d, "\varepsilon F"] \\ & F \end{tikzcd}$$

For this to commute, we need that the following commutes:

$$\begin{tikzcd} FA \arrow[r, "F\eta A := F(\eta_A)"] \arrow[dr, "1_{\mathbf{Mon}}"'] & FUFA \arrow[d, "\varepsilon FA = \varepsilon_{FA}"] \\ & FA \end{tikzcd}$$

Let $x \in A$
$\varepsilon_{FA} \circ F(\eta_A)(Fx) = \varepsilon_{FA}((1;x)) = x$. So, the diagram commutes.
On an individual element, we get

Problem 60. Is there a free-functor-shaped left adjoint to the forgetful functor from $\mathbf{Field}_p$ to **Set** for a fixed p?
Solution.

Problem 62. If $F \dashv U$, show that the counit of the adjunction is invertible iff U is fully faithful. Prove a dual statement about the unit.
Solution.

Problem 65. If $F \dashv U$ and $F' \dashv U'$, then $F'F \dashv UU'$.
Solution.

Problem 67. Construct a 2-category **Adj** with objects small categories, 1-cells from $\mathcal{A} \to \mathcal{B}$ given by pairs of adjunct functors $F : \mathcal{A} \to \mathcal{B}, U : \mathcal{B} \to \mathcal{A}$ with $F \dashv U$, and 2-cells $\alpha : F_1 \dashv U_1 \Rightarrow F_2 \dashv U_2$ given by $\alpha : F_1 \Rightarrow F_2$.
Solution.

Problem 68. Show the functor $(-)^{\mathrm{op}}$ is self-adjoint.
Solution.