Carl’s Notes on Dwave (from https://docs.dwavesys.com/docs/latest/):

- Two formulations we look at for objective functions are found in the Ising model and in QUBO problems. Conversion between these two formulations is trivial.
  - QUBOs and the Ising model are basically just similar representations of the same objective function.
- Quadratic unconstrained binary optimization problems—QUBOs—are unconstrained in that there are no constraints on the variables other than those expressed in Q.
- QPU architecture - the layout of the QPU is important to translation the QUBO or Ising objective function into a format the Dwave system can solve
  - Although Ocean software automates this mapping, you should understand it if you are directly programming the QPU because it has implications for the problem-graph size and solution quality.
- The process of mapping the logical qubits to physical qubits is known as minor embedding.
- Scaling the objective value for the QUBO to get a larger energy gap between the ground states and the excited states doesn’t affect the results/number of occurrences of each state.
  - This result is caused by a feature of the D-Wave system known as auto-scaling. Each QPU has an allowed range of values for the biases and strengths of a and b. Unless we explicitly disable auto-scaling, the D-Wave software adjusts the a and b values of a problem to take the entire (a,b) range available before sending it to the QPU. As a result, by the time these two problems are run, they present the same (a,b) values to the QPU, and therefore the returned solutions are effectively the same. When the energies and objective values are reported at the end of the runs, we are using the pre-scaling values.
- Because the yield of a working graph is typically less than the total number of qubits and couplers that are fabricated and physically present in a QPU, we must first verify that the qubits we selected are active. Likewise for the selected couplers. If any are inactive, we can use a different unit cell. For example, to check the qubits in the first unit cell:
  - >>> print(sampler_manual.nodelist[0:8])
  - [0, 1, 2, 3, 4, 5, 6, 7]
- Leap’s quantum-classical hybrid solvers are intended to solve arbitrary application problems formulated as binary quadratic models (BQM) or discrete[1] quadratic models.
  - discrete quadratic model (DQM) solvers solve problems with variables that have more than two values; for example, variables that represent colors or DNA bases.

- Interesting Quantum Annealing Controls that might be of use
  - Anneal Schedule Variations
    - In the standard application of quantum annealing in D-Wave systems, qubits evolve according to a predetermined schedule, in which energy changes smoothly as a function of scaled time. As in previous releases, you can change the default duration using the annealing_time parameter when submitting a problem.

        - Some types of research, however, may benefit from more fine-grained adjustments to the default anneal schedule. To provide this level of control, anneal schedule features enable you to change the shape of the energy waveform by providing points at which to pause or quench (i.e., abruptly terminate) the anneal process; see Figure 90. This level of control helps investigate what is happening partway through the annealing process.
        - Unlike the anneal offsets feature—which allows you to control the annealing path of individual qubits separately—anneal schedule changes apply to all qubits in the working graph.
    - Reverse Annealing
        - Reverse annealing, a technique that makes it possible to refine known good local solutions thereby increasing performance for certain applications.[1] Reverse annealing involves (1) annealing backward from a known classical state to a mid-anneal state of quantum superposition, (2) searching for optimum solutions at this mid-anneal point while in the presence of an increased transverse field (quantum state), and then (3) proceeding forward to a new classical state at the end of the anneal.
    - Anneal Offsets
        - In the standard application of quantum annealing in D-Wave systems, all qubits evolve simultaneously, experiencing equal changes to tunneling energy and making an equal contribution to the classical energy function.
        - In some situations, however, it is beneficial to offset the annealing paths of the qubits, so that some are annealed slightly before others. This technique can improve both optimization and sampling performance for certain types of problems.
    - Time-Dependent Gain in Hamiltonian Biases (H-Gain)
        - This feature increases user control of the Hamiltonian that represents the D-Wave system’s quantum anneal by introducing a time-dependent gain on its linear coefficients (biases), hi.

Different type of Dwave solvers
- QPU (Hardware) Solvers
- QPU-Like Solvers
- Leap’s Hybrid Solvers
- Ising Heuristic Solvers

Notes on Problem Solving
- Discrete optimization, also known as combinatorial optimization, is the optimization of an objective function defined over a set of discrete values such as Booleans.
- We can convert a Constraint Satisfaction Problem into a Maximum Independent Set problem

- Even if a penalty can be derived through conversion to a linear equality or inequality, it may still be useful to apply the penalty-deriving machinery discussed in the Nonlinear Constraints section to construct a penalty with fewer ancillary variables.
    - Commonly used to reduce 3-SAT to MAX-2-SAT