HILL CLIMBING

- start at a random point \( X \)
- loop
  - if there is a “neighbor” that is better
    - set \( X \) equal to the neighbor
  - else
    - return \( X \) as the solution

For example, consider the two functions at the upper right.
Suppose we wish to find the value \( x \) where \( f(x) \) is at its minimum (or maximum).
This is called “minimizing” (or “maximizing”) – or simply “optimizing”:
  - The function on the left has ONE global minimum, and NO local minima.
  - The function on the right has TWO global minima, and several local minima.
So hill-climbing is guaranteed to solve the function on the left, but not necessarily the one on the right.

Requires defining a “NEIGHBOR”. This is problem-dependent. For the numeric functions above, one simplistic approach would be \( \text{NEIGHBOR}(X) = X \pm 0.001 \) (two neighbors). [note that in this case, however, the resulting solution may not be exactly correct]

Try to define “NEIGHBOR” functions for the following optimization problems:
- \( f(x,y) = x^2 + y^2 \)
- 8-queens problem

Advantages of hill-climbing: very simple, very fast, can be tailored to different problems.
Disadvantage of hill-climbing: susceptible to local minima, requires definition of “neighbor”.

An interesting variation on hill-climbing is “bit-climbing”:
- Devise a binary-encoding for \( X \)
- a “NEIGHBOR” is a single bit-flip
- the number of possible neighbors is equal to the bit-length of the encoding

Example:
Suppose you wish to solve a KNAPSACK problem using bit-climbing.
In a knapsack problem, there are several objects, each with a weight \( W \) and a value \( V \), and a “sack” with capacity \( C \). Find the collection of objects such that \( \Sigma(W) \leq C \), and \( \Sigma(V) \) is maximized.
A binary encoding for this problem would be to assign a BIT to each possible object.
A “0” means that object isn’t included, a “1” means that object is included.
Thus, if there are 10 possible objects, the string 0010010000 means to include the 3rd and 6th objects only.
For this scenario, there are always 10 possible neighbors (1010010000, 0110010000, 0000010000, etc.).

Start with a random bitstring. This represents a possible solution.
Evaluate the string by computing \( \Sigma(V) \) for the objects which have their bits set to “1”.
Also compute \( \Sigma(W) \) for each of the string’s 10 neighbors.
If one of the neighbors is more optimal than the original string, replace the string with that neighbor.
Repeat until a string is found with no better neighbors
Note that if the string has \( \Sigma(W) > C \), it cannot be a solution.
This method is still susceptible to local minima -- the resulting string may not be the global optimum.
- inspired by the “annealing” of metals
- optimal molecular crystal lattices achieved by heating the metal, then cooling it gradually
- requires definition of “NEIGHBOR”, and definition of “TEMPERATURE”.

Algorithm:

- switching to an inferior neighbor accepted with decreasing probability over time.
- temperature usually modeled as a value that starts at 1 (“high”) and gradually reduces to 0.
- temperature decrease can be computed as a percentage, such as TEMP’ = 0.99 (TEMP)
- Boltzmann distribution is commonly used: \( P(\Delta E, T) = \frac{1}{1+e^{E/CT}} \)  \( E=\text{energy}, T=\text{temp}, C=\text{constant} \)

An interesting variation is “neighbor annealing”:
- requires defining a “neighborhood” – neighbors are selected randomly from neighborhood.
- when temperature is high, neighborhood size is large (neighbors can be far away).
- as temperature decreases, neighborhood size gets smaller (neighbors are only close by).
- switch to the neighbor ONLY if it is better.

Example:
Consider the “sin bowl” function at the upper right of the previous page.
X can start as a random value between -50 and +50.
The definition of neighborhood can start as X ± 20·(temperature)
At each step, a random neighbor can be selected from the neighborhood.
If the neighbor is better than X, then set X equal to that neighbor.
At each time step, lower the temperature by setting temperature = 0.99 · (temperature)
When temperature reaches a low value, such as 0.0001, stop. X is the proposed solution.