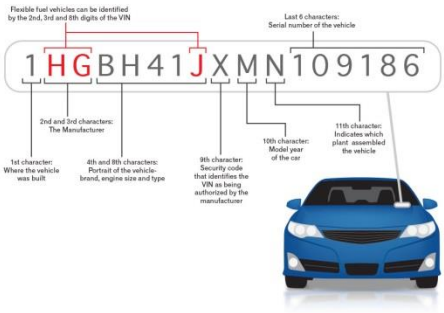
Hash Table Lab

"My hashCode is my bond."

After finishing each part of the lab, copy your entire project and work on the copy for the next part!

Part 4: Create a *VIN* class and *Car* class; implement *hashCode* & *equals* methods.



- Create a *VIN* class, which holds a property for a vehicle ID number string.
  - Expectations when you write your own class:
    - Provide a no-arg constructor, a member-wise constructor, & any other constructor that will make using the class easier (think about typical data that will be provided).
    - Override *equals*, *hashCode*, & *toString*
  - Follow the rules for *hashCode* & *equals*!
    - For any 2 instances of type *VIN* that have the same field values, *equals* must return *true*, and *hashCode* must return the same value.
      - If `a.equals(b) == true`, `a.hashCode()` must be same as `b.hashCode()`
    - For any 2 instances of type *VIN* that have different field values, *equals* must return *false*, and *hashCode* should return a different value.
    - Use the same fields that you use to compute *equals* to compute *hashCode*
    - Calculate hashCode when called – don't precalculate
- Create a *Car* class, which holds string properties for VIN, year, make, & country of origin.
  - The Car class is a convenience structure - you do not have to implement *hashCode* or *equals*.
  - It might be convenient to override *toString*
- Using a copy of the previous lab sections:
  - Modify the file reading code to read *VIN* & *Car* data.
  - Make minor mods to code involving data output.
  - Run the table building, searching, and timing code as before (it should be unchanged)
  - Create Excel spreadsheets & charts as before
- As before, turn in your spreadsheet with charts to Canvas