Chapter 18: Table-Driven Methods

- A table driven method is a scheme that allows you to look up information in a table rather than using logic statements (if and case) to figure it out
- Virtually anything you can select with a logic statement can be selected with tables instead

General Considerations in Using Table-Driven Methods

- Used in appropriate circumstance, table driven code is
  - Simpler than complicated logic
  - Easier to modify
  - More efficient

Two Issues in Using Table-Driven Methods

- First, need to address how to loop up entries in the table
  - Direct access
  - Indexed access
  - Stair-step access
- Second, you need address what you should store in the table
  - Some cases, the result of a table lookup is data
  - Other cases, the lookup is an action

Direct Access Tables

- Like directly accessing the value
  - Either by index? Or idk, kinda vague

Days in Month Example

- Suppose need to determine number of days per month
  - Need if branch for each month
- OR store each month number of days in a table then use the months number to access the days
  - I already do this all the time lel

Fudging Lookup Keys

- In the examples above, could use the data to key into the table directly
- You always want to be able to key into a table directly
  - But sometimes data is in weird ranges to need to fudge
- How to fudge
  - Duplicate data in the table to accommodate ranges
  - Transform the key to make it work directly
    - Min(key, bound), max(key, bound)
    - Isolate the transformation into its own routine

Indexed Access Tables

- Sometimes a simple mathematical transformation isn't enough to jump from "age" to a table key
- In using indexes
    - Use primary data to lookup a key in an index table
    - Then use the value from the index table to look up the main data youre interested in
- Advantages
    - First, if each of the entries in the main lookup table is large, it takes a lot less space to create an index array with wasted space than a main lookup table with lots of wasted space
    - Second, even if you don't save space in an index, sometimes its cheaper to manipulate entries in an index than a main table
    - Finally, much easier to maintain than data embedded in code

Stair-Stepped Access Tables

- Isnt as direct as an index structure, but doesn't waste as much space
- General idea is entries in a table are valid for ranges of data, instead of distinct points
    - A
    - B
    - C
    - D
    - F
- To use
    - Put the upper end of each range into a table
    - Write a loop to check the score against the upper end of each range
- Works well with irregular data on irregular intervals
- Flexible and modifiable
- Things to note
    - Watch the endpoints and make sure you've covered the case at the top end of the stair-step range
    - Be careful about mistaking < for <=
    - Consider using binary search instead of sequential search
    - Consider using indexed access instead of stair step
        - The searching in stair-step can add up
    - Put the stair-step into its own routine