Designing using itemizations and structures

September 25, 2015

Below are some problem statements whose solutions will require the use of itemizations or unions. Apply the design recipe from start to finish to produce a complete solution to each, writing code and comments in DrRacket. Work in pairs.

1. Design a function `next-day` that, given a day of the week (as a string), returns the next day of the week. So, for example, `(next-day "monday")` should return `"tuesday"`.

2. Design a function `collision?` that, given two coordinates in the x/y plane, returns `#true` if they are within a distance of 10, and `#false` otherwise.

3. Design a function `percentage->grade` that converts a percentage to a letter grade as follows: 90 to 100 is an A, 80 to 90 is a B, 70 to 80 is a C, 60 to 70 is a D, and below 60 is an F. Return a string indicating the letter grade.

4. Design a function `check-speed` for an airplane's automatic pilot that keeps the airspeed safe. If the plane drops below 80 knots, it will start to stall and needs to speedup. If it goes above 300 knots, it is going dangerously fast and needs to slow down. Return a string `"speed up"`, `"slow down"`, or `"ok"`, depending on the speed.