CSE 230 Problem Set 07

# Problem 23.1: Year Class

Consider the following class diagram:



Consider the following method definitions:



Classify the level of abstraction of the following class which stores a year as an int. Justify your answer. Hint: What happens when you subtract 2030 from today’s year?

I consider this to be **Opaque**. It is incredibly straightforward, but it would help to have negative years explained.

# Problem 23.2: Date-Time Class

Consider the following class diagram:



The Unix operating system represents time using the POSIX format. Here, time starts on the 1st of January 1970. Time is stored as a 32-bit integer, representing the number of seconds since that date. Classify the level of abstraction of the following class implementing POSIX date/time:



Classify the level of abstraction of the following class and justify your answer. Hint: What happens on the 19th of January 2038?

I classify this level of abstraction as **porous**, because once the user sees that the year resets to 1902, it will be obvious and exploitable how the class works. I would be compelled to tell the user that it only goes up to that year.

# Problem 23.3: File Name Class

Consider the following class diagram:



Consider the following method definitions: Note that the isGreater() method is used to sort files by their name so they are presented to the user in alphabetical order.



Classify the level of abstraction of the following class and justify your answer. Hint: What happens when I try to list “a.txt” and “B.txt” in the same directory?

I regard this as **Opaque**, as it can still function fine without added knowledge. It just has the quirk of grouping capitalized filenames separately.

# Problem 23.4: Chess Piece Class

Consider the following class diagram:



Classify the level of abstraction of a class designed to store a chess piece on a chess board. The member variable is a single character where ‘r’ corresponds to a white rook and ‘R’ corresponds to a black one. Note that the getValue() method returns the character corresponding to each chess piece.

I mark this as **Complete** because the user can easily find out which piece goes to which color. No help is needed, and no bugs are found.

# Problem 23.5: Angle Class

Classify the level of abstraction of a class that stores an angle. This allows the client to work equally with radians (where 2π is a complete loop around a circle) and degrees (where 360° is a complete loop).

I mark this as **complete**. The user will know when they’re using radians or degrees, and the ability to use both means this has short development time, high comprehension, and stability.