Tracy Mann

CSE210

Bro. Parrish

Section A8

Abstraction to the best of my understanding is the process of making something abstract or more generic. In order to do so, you must bring in the focus on exactly what this object is in all of its different forms if it has any. If the object can act without using the descriptor or method at all then it should also not be defined in this object type. On the contrast, Any descriptor or method that only applies to a more specific type of this same object should also not be defined in this object type. In an essence, this process defined the scope of the object and helps keep the object doing only what the object is supposed to do and no more or less.

The great advantage of this helps to simplify the design of the object. Making the object less error prone and easier to support. One example I remember that demonstrates abstraction is if you were to define an object called pen, used to represent a physical pen. The pen would have descriptors of body, point or tip, ink or dye, and possibly clip or even an optional tip retractor feature. The methods could include draw, refill, store, recover, and possibly retract or deploy the tip. This pen would not typically have an erase method or a target surface color descriptor.

In the assignment, I used abstraction for the journal to break out that a journal is a collection of entries. The entry is comprised of a date, a prompt and a response. The Prompt in my example was tracked to count the number of times that prompt was used and when it was last used. This information in the prompt helped determine a balanced distribution for random selection of a prompt. Entry itself does not even need to know about or use any of those details. The same as, Prompt does not need to know or use any details on the date or response of the Entry. As show below, I extracted the constructors and encapsulation of the accessors for the descriptors and demonstrate only the abstraction layer. The note about the csv and json functions is that they only convert the value of the object to a string in that format, not actually write the file and vice versa for reading and importing.

public class Entry

{

    private string \_date;

    private Prompt \_prompt;

    private string \_response;

…

    public void Display(Encryption encryption){…}

    public string GetCSV(Encryption encryption, Boolean encrypted){…}

    public void ParseCSV(string input, Encryption encryption, Boolean encrypted) {…}

    protected string JSON {}

}

public class Prompt

{

    private string \_value;

    private string \_timesUsed;

    private string \_lastUsed;

…

    public void Display(Encryption encryption) {…}

    protected string JSON {…}

}

I also did my best to break out the File read and write functions to their own helper class as well as a helper class for database utilization. Also, though my classes needed to use my encryption object they let the encryption object handle all the encryption processes. The last part of abstraction I did was to make a distinction of the encrypted prompt/entry against the unencrypted prompt/entry as similar but separate classes.

This allowed me to focus on each part of the assignment localized to the object I was working on at the time.