Daniel Hanna
Assignment 4

This project must demonstrate modularity and encapsulation, elegance and efficiency, & must also utilize the appropriate data structures as needed. Modularity and encapsulation are addressed by creating different classes (particularly based off of type), while avoiding the common-tendency of creating classes that are redundant in their functionality. For example, it was initially considered as to whether a “Explore Interface” class, “Homepage Interface” class, and “Help Form Interface” class should be created. However, it was determined that such an implementation would not only be redundant, but would also most likely raise difficulties when attempting implementation. Instead, these different interfaces are encapsulated under the “Interface” class.

When developing an application, it would not be logical for the boundary class to perform all processes and algorithmic actions. Therefore, modularity was further demonstrated by creating a guide and form class, which would hold data and calculate input (respectively). The result is a system which is elegant in nature by efficiently allocating processes amongst the said classes, instead of performing all tasks under one large entity, boundary, or controller.

Appropriate data structures are utilized based off of a functionality perspective. For example, a form question such as “Do you own a smartphone?” can be captured by the user and interpreted as a boolean data type, since the only reasonable responses to this question are “Yes” or “No”. User input as a whole may be captured and interpreted as a hash, since hash headings may hold the description of the form entry (for example: “Smartphone” => false).

These considerations amount to a system that works well and presents a reasonable and sound structure that will not only make logical sense, but will also be effective during implementation.